A ENUMERAÇÃO DOS K TRAJECTOS MAIS CURTOS

Marta Margarida Braz Pascoal

marta@mat.uc.pt
Departamento de Matemática
Universidade de Coimbra
PORTUGAL

Novembro 1998

Abstract:

The K Shortest Paths Problem can be considered as a generalization of the classical Shortest Path Problem, since it is intended to determine not only the shortest path between two nodes, but also the second shortest path, ..., until the K-th shortest path between that pair of nodes. When it is intended to determine only loopless paths (that is, paths with no cycles) we have the K Shortest Loopless Paths Problem.

Although the first problem verifies the Optimality Principle, the second one does not and therefore the development of rather different algorithms for the resolution of both problems was necessary.

In this paper, after a brief analisys of these two problems, the notion of shortest paths tree is introduced. After a classification of some of the known algorithms for the K Shortest Paths Problem, and using the shortest paths tree concept, those two problems can be related in a manner which allows the development of new algorithms for its resolution. Finally, some of those algorithms are presented.

Keywords: path, loopless path, ranking

Resumo:

O Problema dos K Trajectos mais Curtos pode ser considerado como uma generalização do clássico Problema do Trajecto mais Curto, já que se pretende determinar não só o trajecto mais curto entre dois nós, mas também o segundo trajecto mais curto, ..., até ao K-ésimo trajecto mais curto entre o mesmo par de nós. No caso particular da determinação unicamente de trajectos sem ciclos, ou seja, de caminhos, temos o Problema dos K Caminhos mais Curtos.

Ao contrário do que se verifica para o primeiro problema, o segundo não satisfaz o Princípio de Optimalidade, o que tem originado o desenvolvimento de algoritmos substancialmente diferentes para a resolução dos dois problemas.

Neste artigo, após uma breve análise destes dois problemas, é introduzida a noção de árvore dos trajectos mais curtos. Após uma classificação de alguns dos algoritmos conhecidos para o Problema dos K Trajectos mais Curtos, e utilizando o conceito de árvore dos trajectos mais curtos, é possível relacionar de forma mais directa os dois problemas, permitindo o posterior desenvolvimento de novos algoritmos para a sua resolução, algoritmos esses que são também apresentados.

Palavras–chaves: trajecto, caminho, enumeração

---

1 Introdução

O Problema dos K Trajectos mais Curtos é um problema bastante conhecido da Optimização em Redes, cujo objectivo é determinar K trajectos entre dois nós de uma rede, por ordem não decrescente dos seus custos, ou seja, determinar o trajecto mais curto, o segundo trajecto mais curto, ..., até ao K-ésimo trajecto mais curto. É de referir que, quando $K = 1$, este problema se reduz ao Problema do Trajecto mais Curto.

Este Problema dos K Trajectos mais Curtos é dito sem restrições, sendo no entanto possível considerar algumas restrições adicionais. Um exemplo conhecido de problemas desta última classe é o Problema dos K Caminhos mais Curtos, onde os trajectos determinados, não podem conter ciclos, isto é, só é permitida a determinação de caminhos.

Apesar de estes dois problemas serem semelhantes na sua descrição, o facto de no segundo ser imposta uma restrição à determinação de trajectos implica que não verifique o Princípio de Optimalidade, ao contrário do problema geral, que sob certas condições satisfaz este princípio.

Na literatura são conhecidos vários algoritmos que permitem a resolução do problema sem restrições, a maioria dos quais baseados no Princípio de Optimalidade e, consequentemente, não aplicáveis à determinação dos K caminhos mais curtos. Por esta razão o Problema dos K Caminhos mais Curtos tem vindo a ser considerado mais difícil do que o Problema dos K Trajectos mais Curtos.

Neste artigo começamos por apresentar os dois problemas acima referidos, e por estudar condições que garantem a respectiva finitude e o facto de estes serem ou não limitados. Em seguida é feita referência a alguns algoritmos conhecidos para a resolução daqueles problemas e, por fim, é introduzida uma noção nova, válida quer para o problema geral quer para o problema com restrições, com as adaptações devidas, com base na qual é possível desenvolver, de forma análoga, algoritmos para ambos os problemas.

2 Notação e definições

Embora neste artigo se utilize a notação usada em [10], trabalho que se encontra acessível no endereço http://www.mat.uc.pt/~marta/Publicacoes.html, tendo em vista facilitar a leitura, nesta secção introduzir-se-á não só a notação utilizada mas também algumas definições.

Denotemos por $(\mathcal{N}, \mathcal{A})$ uma rede, onde $\mathcal{N} = \{v_1, \ldots, v_n\}$ é um conjunto finito com n elementos denominados nós e $\mathcal{A} = \{a_1, \ldots, a_m\} \subseteq \mathcal{N} \times \mathcal{N}$ é um conjunto finito com m elementos denominados arcos. No que se segue, cada nó v_i será por vezes identificado apenas por i. Analogamente, o arco $a_k \in \mathcal{A}$ poderá também ser identificado por um par (i, j), onde $i, j \in \mathcal{N}$. Os nós i e j são denominados antecessor e sucessor do arco (i, j), respectivamente.

Sejam i e j dois nós de $(\mathcal{N}, \mathcal{A})$; se todos os pares (i, j) de $(\mathcal{N}, \mathcal{A})$ são ordenados, isto é, se todos os arcos da rede são orientados, $(\mathcal{N}, \mathcal{A})$ diz-se uma rede orientada; se todos os pares (i, j) são não ordenados, então $(\mathcal{N}, \mathcal{A})$ diz-se uma rede não orientada.

Sem perda de generalidade, no que se segue iremos considerar que $(\mathcal{N}, \mathcal{A})$ é uma rede orientada, onde existe, no máximo, um arco entre cada par de nós da rede e não existem arcos da forma (i, i), onde $i \in \mathcal{N}$.

Sejam s e t dois quaisquer nós de $(\mathcal{N}, \mathcal{A})$, denominados nó inicial e terminal, respectivamente. Um trajecto do nó s para o nó t em $(\mathcal{N}, \mathcal{A})$ é uma sequência alternada de nós e arcos, da forma $p = \langle s = v'_1, a'_1, v'_2, \ldots, a'_{\ell-1}, v'_\ell = t \rangle$, onde $\ell \geq 2$, e:

- $v'_k \in \mathcal{N}$, para todo $k \in \{1, \ldots, \ell\}$;
- $a'_k \equiv (v'_k, v'_{k+1}) \in \mathcal{A}$, para todo $k \in \{1, \ldots, \ell - 1\}$.

Por conveniência, um único nó pode também ser entendido como um trajecto. Com vista a não sobrecarregar a notação, caso não haja conflito, p será representado simplesmente pelos nós que o constituem, isto é, $p = \langle v'_1, \ldots, v'_\ell \rangle$.

Dados $i, j \in \mathcal{N}$, $\mathcal{P}_{ij}$ denotará o conjunto dos trajectos que se definem de i para j em $(\mathcal{N}, \mathcal{A})$.

Um caminho de s para t é um trajecto de s para t onde todos os nós são distintos e um ciclo é um trajecto de um nó para ele mesmo onde todos os nós são diferentes, excepto o primeiro que coincide com o último.

Seja $p \in \mathcal{P}_{ux}$ um trajecto de u para x e $q \in \mathcal{P}_{xv}$ um trajecto de x para v. A concatenação de p e q, denotada por $p \diamond q$, é um trajecto de u para v constituído pelo trajecto p até ao seu último nó, x, e seguido pelo trajecto q de x até v. Dado um ciclo $\mathcal{C}$, pode ser definida a k-concatenação de $\mathcal{C}$ por $\mathcal{C}^k = \mathcal{C}^{k-1} \diamond \mathcal{C}$, para $k \geq 1$, e onde $\mathcal{C}^0$ é um dos nós do ciclo $\mathcal{C}$. Dados dois nós i e j de um trajecto (caminho) p, um subtrajecto (subcaminho) de p de i até j é um trajecto (caminho) de i para j que coincide com p entre esse par de nós, e é denotado por $\text{sub}_p(i, j)$.

Um trajecto diz-se finito se o respectivo número de nós (e, consequentemente, de arcos) é finito.

No que se segue, s e t ($s \neq t$) designarão dois dados nós da rede e $\mathcal{P}$ designará o conjunto $\mathcal{P}_{st}$. Supor-se-á também, e sem perda de generalidade, que $\mathcal{P}_{si}$ e $\mathcal{P}_{it}$ são conjuntos não vazios, para todo o nó i em $(\mathcal{N}, \mathcal{A})$.

A cada arco (i, j) da rede $(\mathcal{N}, \mathcal{A})$ iremos associar um valor real, c_{ij}, que denominamos por custo ou distância de (i, j). Será ainda considerada uma função $c : \mathcal{P} \to \mathbb{R}$, que a cada trajecto p, de s para t, atribui um valor real determinado por $c(p) = \sum_{(i,j) \in p} c_{ij}$. Com o intuito de simplificar a notação escreveremos por vezes $c(p) = \sum_p c_{ij}$.

3 Os problemas da enumeração de trajectos e de caminhos

3.1 Os problemas

Um problema bem conhecido em Optimização é o Problema dos K Trajectos mais Curtos, onde K é um número natural dado. Neste problema pretende-se enumerar K trajectos distintos de s para t em $(\mathcal{N}, \mathcal{A})$ por ordem não decrescente de custos, isto é, determinar $p_1, \ldots, p_K \in \mathcal{P}$ tais que:

- p_i é determinado exactamente antes de p_{i+1}, para todo o $i \in \{1 \ldots, K-1\}$;
- $c(p_i) \leq c(p_{i+1})$, qualquer que seja $i \in \{1, \ldots, K-1\}$;
- $c(p_K) \leq c(p)$, para todo o $p \in \mathcal{P} - \{p_1, \ldots, p_K\}$.

Para $K = 1$ este problema reduz-se ao clássico Problema do Trajecto mais Curto pelo que, no que se segue, iremos considerar $K > 1$.

Dado que, neste problema, pretendemos determinar os K trajectos mais curtos, são permitidas repetições de nós em cada um dos trajectos, pelo que é possível determinar trajectos com ou sem ciclos. Trata-se pois do Problema dos K Trajectos mais Curtos, dito sem restrições. Podemos no entanto considerar variantes deste problema impondo restrições aos trajectos a determinar. Deste modo, se por exemplo pretendermos calcular apenas trajectos sem ciclos, ou seja, caminhos, teremos o Problema dos K Caminhos mais Curtos. Este pode ser visto como um subproblema do anterior; no entanto, apesar de as suas descrições serem similares, os dois problemas são distintos. Por exemplo, como será referido em seguida, o Problema dos K Trajectos mais Curtos verifica o Princípio de Optimalidade, ao contrário do Problema dos K Caminhos mais Curtos. Esta é uma das razões pelas quais o Problema dos K Caminhos mais Curtos vem sendo considerado como "mais difícil", o que se reflecte, como veremos adiante, nos processos existentes para a resolução dos dois problemas.

É ainda de referir que é possível que não existam K trajectos/caminhos de s para t na rede considerada. Nesse caso serão determinados apenas os trajectos/caminhos possíveis, ou seja, todos os existentes entre aqueles dois nós dessa rede. Note-se que, caso existam ciclos em $(\mathcal{N}, \mathcal{A})$, dada a hipótese assumida de $\mathcal{P}_{si} \neq \emptyset$ e $\mathcal{P}_{it} \neq \emptyset$ para todo o nó $i \in \mathcal{N}$, então o número de trajectos é não finito. No entanto, caso $(\mathcal{N}, \mathcal{A})$ não contenha ciclos então todo o trajecto é caminho.

3.2 Finitude

Na prática interessa-nos apenas a determinação de trajectos que sejam finitos pelo que nesta secção tentaremos estabelecer condições que garantam a finitude do Problema dos K Trajectos mais Curtos.

A resolução de um Problema dos K Trajectos mais Curtos é levada a cabo recorrendo à resolução de vários problemas do trajecto mais curto. Por esta razão começaremos por analisar o caso de $K = 1$.

O Problema do Trajecto mais Curto diz-se finito se existir uma solução óptima finita, isto é, se existir um trajecto mais curto que seja finito. E diz-se limitado se o valor óptimo é finito.

Quando o Problema do Trajecto mais Curto não é finito não existe nenhum trajecto óptimo finito. Convenciona-se então que a solução óptima é o limite de uma sucessão de trajectos, cuja sucessão dos custos associados converge para o valor óptimo, ou seja, para o menor custo possível.

Teorema 1 *Seja $(\mathcal{N}, \mathcal{A})$ uma rede tal que $\mathcal{P}_{si} \neq \emptyset$ e $\mathcal{P}_{it} \neq \emptyset$, para todo o $i \in \mathcal{N}$. O Problema do Trajecto mais Curto de s para t em $(\mathcal{N}, \mathcal{A})$ é finito se e só se não existem ciclos negativos em $(\mathcal{N}, \mathcal{A})$; isto é, ciclos $\mathcal{C}$ tais que $\sum_{\mathcal{C}} < 0$.*
Demonstração: Ver [10].

Deste Teorema e da sua demonstração surgem de imediato duas consequências. A primeira é que todo o Problema do Trajecto mais Curto não finito é também não limitado. A segunda consequência é que se o Problema do Trajecto mais Curto é finito, então existe uma solução óptima, isto é, um trajecto óptimo, que é também um caminho.

Alguns dos resultados enunciados, ou apenas referidos, nesta secção para $K = 1$ podem ser vistos de forma mais detalhada em [7].

Considerando $K > 1$, o Problema dos K Trajectos mais Curtos diz-se finito se existirem K trajectos mais curtos, de s para para t em $(\mathcal{N}, \mathcal{A})$, finitos.

Teorema 2 *O Problema dos K Trajectos mais Curtos de s para t em $(\mathcal{N}, \mathcal{A})$ é finito se e só se o Problema do Trajecto mais Curto de s para t, na mesma rede, é finito.*
Demonstração: Ver [10].

De forma a obtermos apenas trajectos que sejam finitos, no que se segue iremos considerar, salvo indicação em contrário, que $(\mathcal{N}, \mathcal{A})$ é uma rede sem ciclos negativos. Como referimos, isto garante a finitude do Problema dos K Trajectos mais Curtos, e consequentemente o facto de ser limitado.

Note-se que, ainda que o Problema dos K Trajectos mais Curtos seja finito, podem existir soluções óptimas não finitas. De facto, usando a rede da Figura 1, todos os trajectos da forma $\langle 1, 2, 3 \rangle \diamond \mathcal{C}^k$, com $k \geq 0$ e onde $\mathcal{C} = \langle 3, 2, 3 \rangle$ é um ciclo nulo, isto é, $c(\mathcal{C}) = 0$, são trajectos mais curtos de 1 para 3. É ainda de notar que qualquer que seja o valor de K escolhido, os trajectos obtidos têm todos a mesma forma, o que implica a não consideração do trajecto $\langle 1, 3 \rangle$.

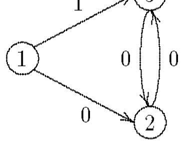

Figura 1: Rede

Situações análogas a esta ocorrem sempre que existe um ciclo nulo contendo um nó que pertence também a um dos k trajectos mais curtos de s para t em $(\mathcal{N}, \mathcal{A})$, com $k < K$. Com o intuito de evitar soluções deste tipo, iremos considerar apenas redes onde não existem ciclos nulos (além de negativos), ou seja, vamos supor que $c(\mathcal{C}) > 0$ para todo o ciclo $\mathcal{C}$ de $(\mathcal{N}, \mathcal{A})$.

Apesar de ser um caso particular do Problema dos K Trajectos mais Curtos, o Problema dos K Caminhos mais Curtos é de natureza diferente. De facto, começando desde logo por analisar o caso $K = 1$, podemos verificar que este problema é finito qualquer que seja a rede considerada, uma vez que um caminho não pode conter nós repetidos. Consequentemente o Problema dos K Caminhos mais Curtos, para $K > 1$, é também finito em qualquer tipo de rede.

4 A enumeração de trajectos

O Problema dos K Trajectos mais Curtos, apresentado na secção anterior, tem sido estudado por vários autores, pelo que são conhecidos alguns algoritmos para a sua resolução, a maior parte dos quais baseados no denominado Princípio de Optimalidade. No entanto, é possível propor novos algoritmos para este problema tendo como suporte o conceito, que introduziremos mais adiante, de árvore dos K trajectos mais curtos.

Nesta secção iremos classificar os algoritmos, tendo em atenção os pressupostos em que se apoiam. Far-se-á ainda uma referência breve a algoritmos conhecidos e apresentar-se-ão também dois novos algoritmos.

No que se segue iremos denotar o k-ésimo trajecto mais curto de s para t, numa rede $(\mathcal{N}, \mathcal{A})$, por $p_k = \langle s = v_1^k, \ldots, t = v_{\ell_k}^k \rangle$, para $k \in \{1, \ldots, K\}$, e, com o objectivo de simplificar a notação, escreveremos $\mathrm{sub}_k(i,j)$, em vez de $\mathrm{sub}_{p_k}(i,j)$, para denotar o subtrajecto de i para j de p_k.

4.1 Algoritmos baseados no Princípio de Optimalidade

4.1.1 Princípio de Optimalidade

Tal como na secção anterior comecemos por analisar o caso $K = 1$.

Diz-se que o Problema do Trajecto mais Curto verifica o Princípio de Optimalidade quando todo o trajecto mais curto é constituído por subtrajectos mais curtos.

Teorema 3 *O Problema do Trajecto mais Curto verifica o Princípio de Optimalidade se e só se não existem ciclos negativos em* $(\mathcal{N}, \mathcal{A})$.

Demonstração: Suponhamos que $(\mathcal{N}, \mathcal{A})$ contém pelo menos um ciclo negativo, $\mathcal{C}$, o que implica, pelo Teorema 1, que o Problema do Trajecto mais Curto não é finito. Seja i um nó de $\mathcal{C}$; por hipótese existem $p_1 \in \mathcal{P}_{si}$ e $p_2 \in \mathcal{P}_{it}$. Denotemos por p_∞ o limite da sucessão $\left\{p_1 \diamond \mathcal{C}^k \diamond p_2\right\}_{k=0}^{\infty}$, que é uma solução óptima do problema uma vez que $c(p_\infty) = -\infty$. Seja então q um subtrajecto de p_∞ de s para i. Como podemos verificar, q não é um subtrajecto mais curto de s para i uma vez que, sendo $c(\mathcal{C}) < 0$, então $c(q \diamond \mathcal{C}) = c(q) + c(\mathcal{C}) < c(q)$. Deste modo p_∞ contém subtrajectos que não são óptimos e portanto, o Problema do Trajecto mais Curto não verifica o Princípio de Optimalidade.

Suponhamos agora que o Problema do Trajecto mais Curto é finito e que não verifica o Princípio de Optimalidade; isto é, existe um trajecto mais curto de s para t, p, que é finito e tal que $\mathrm{sub}_p(i,j)$ não é trajecto mais curto de i para j. Então existe outro trajecto $q \in \mathcal{P}_{ij}$ tal que $c(q) < c(\mathrm{sub}_p(i,j))$. A partir de q podemos obter outro trajecto de s para t em $(\mathcal{N}, \mathcal{A})$, $\mathrm{sub}_p(s,i) \diamond q \diamond \mathrm{sub}_p(j,t)$, e então

$$\begin{aligned} c(\mathrm{sub}_p(s,i) \diamond q \diamond \mathrm{sub}_p(j,t)) &= c(\mathrm{sub}_p(s,i)) + c(q) + c(\mathrm{sub}_p(j,t)) \\ &< c(\mathrm{sub}_p(s,i)) + c(\mathrm{sub}_p(i,j)) + c(\mathrm{sub}_p(j,t)) = c(p), \end{aligned}$$

pelo que p não seria trajecto óptimo. $\square$

Baseados neste Teorema podemos concluir que, desde que $(\mathcal{N}, \mathcal{A})$ não contenha ciclos negativos, dado um trajecto mais curto de s para t, um seu subtrajecto de i para j é também um trajecto mais curto de i para j. Deste modo é possível construir uma árvore com raiz no nó inicial s, contendo todos os nós de $(\mathcal{N}, \mathcal{A})$, e na qual o único trajecto de s para um nó i é um caminho mais curto de s para i. Esta árvore conterá, em particular, um caminho mais curto de s para t e pode ser obtida com um algoritmo de rotulação, algoritmos estes sobejamente conhecidos, [4].

Na realidade, para que um dado Problema do Trajecto Óptimo possa ser resolvido por um algoritmo de rotulação, basta que uma das soluções óptimas seja constituída por subtrajectos óptimos. No entanto, como vimos, no caso do Problema do Trajecto mais Curto sempre que isto acontece (isto é, quando a rede não tem ciclos negativos) podemos mesmo afirmar que acontece para todas as soluções óptimas. Veremos em seguida que sob estas condições, para $K > 1$, o Problema dos K Trajectos mais Curtos verifica uma generalização do Princípio de Optimalidade. De facto, diz-se que o Problema dos K Trajectos mais Curtos verifica o Princípio de Optimalidade se o k-ésimo trajecto mais curto de s para t, com $k \in \{1, \dots, K\}$, é constituído por j-ésimos subtrajectos mais curtos, onde $1 \leq j \leq k$.

Teorema 4 *O Problema dos K Trajectos mais Curtos verifica o Princípio de Optimalidade se e só se $c(\mathcal{C}) \geq 0$, para todo o ciclo $\mathcal{C}$ de $(\mathcal{N}, \mathcal{A})$.*

Demonstração: Supondo que existe um ciclo negativo em $(\mathcal{N}, \mathcal{A})$, pelo Teorema 3, existe um trajecto mais curto contendo subtrajectos que não são óptimos.

Suponhamos agora que $(\mathcal{N}, \mathcal{A})$ não contém ciclos negativos. Seja p_k o k-ésimo trajecto mais curto de s para t em $(\mathcal{N}, \mathcal{A})$, que sabemos ser finito. Suponhamos que $\mathrm{sub}_k(u, v)$ não é j-ésimo trajecto mais curto de u para v, com $j \in \{1, \dots, k\}$. Então, existem $q_1, \dots, q_k \in \mathcal{P}_{uv}$ tais que $c(q_1) < c(\mathrm{sub}_k(u, v)), \dots, c(q_k) < c(\mathrm{sub}_k(u, v))$; logo $\mathrm{sub}_k(s, u) \diamond q_1 \diamond \mathrm{sub}_k(v, t), \dots, \mathrm{sub}_k(s, u) \diamond q_k \diamond \mathrm{sub}_k(v, t)$ são k trajectos de s para t mais curtos do que p_k, pelo que este não seria o k-ésimo trajecto mais curto de s para t em $(\mathcal{N}, \mathcal{A})$. $\square$

4.1.2 Algoritmos

Como referimos o Problema dos K Trajectos mais Curtos, para $K > 1$, tem sido estudado por vários autores, sendo conhecidos vários algoritmos para a sua resolução. A maioria destes algoritmos baseia-se no Princípio de Optimalidade uma vez que este permite:

- A generalização de algoritmos para o problema do trajecto mais curto, devida a Dreyfus, [5], mas utilizando K rótulos para cada nó da rede. Esta generalização não determina apenas os K trajectos mais curtos entre dois nós, mas sim os K trajectos mais curtos de um nó para todos os outros.

- A utilização de algoritmos de rotulação que são a generalização dos algoritmos para o Problema do Trajecto mais Curto mas utilizando K-úplos como rótulos dos nós. Analogamente ao que acontece para $K = 1$, podemos considerar algoritmos de rótulos definitivos (ou permanentes) e algoritmos de rótulos não definitivos (ou temporários), [11, 12].

- O desenvolvimento de algoritmos baseados no apagamento de trajectos, devido a Martins *et al.*, [1, 2, 3, 9]. Nestes algoritmos, uma vez determinado o trajecto mais curto entre os nós inicial e terminal em $(\mathcal{N}, \mathcal{A})$, esta rede é transformada numa outra onde se podem definir os mesmos trajectos do nó inicial para o terminal, à excepção de p_1. Como consequência o trajecto mais curto na nova rede será o mais curto em $(\mathcal{N}, \mathcal{A})$ ignorando p_1, ou seja, o segundo trajecto mais curto entre os nós inicial e terminal em $(\mathcal{N}, \mathcal{A})$. Utilizando repetidamente o mesmo raciocínio, é possível determinar os K trajectos mais curtos entre dois nós de $(\mathcal{N}, \mathcal{A})$.

4.2 Algoritmos baseados na construção de uma árvore de trajectos

Como referimos anteriormente é possível desenvolver algoritmos cujo objectivo é construir uma árvore de trajectos que contém os K trajectos mais curtos que pretendemos determinar. Este conceito e os algoritmos que nele se baseiam são muito recentes, razão pela qual existem poucas referências a indicar. Deste modo, no que respeita à noção de árvore de trajectos e a um algoritmo baseado neste conceito podem consultar-se [6] e [10]. Relativamente aos algoritmos novos, utilizando também uma árvore de trajectos, que aqui irão ser propostos pode consultar-se [10].

4.2.1 Árvore dos K trajectos mais curtos

Com o objectivo de apresentar o conceito de árvore dos K trajectos mais curtos entre dois nós numa rede comecemos por considerar um caso particular. Seja $(\mathcal{N}, \mathcal{A})$ a rede representada na Figura 2, com nó inicial $s = 1$ e nó terminal $t = 5$.

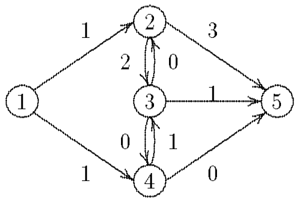

Figura 2: Rede

Para esta rede e considerando $K = 4$, obtemos os seguintes trajectos mais curtos,

$$p_1 = \langle 1, 4, 5 \rangle, \quad p_2 = \langle 1, 4, 3, 4, 5 \rangle, \quad p_3 = \langle 1, 2, 3, 4, 5 \rangle, \quad p_4 = \langle 1, 4, 3, 5 \rangle,$$

com custos $c(p_1) = 1$, $c(p_2) = 2$ e $c(p_3) = c(p_4) = 3$. É de notar que podem existir trajectos distintos com o mesmo custo (o que acontece com p_3 e p_4). Isto significa que o conjunto dos K trajectos mais curtos numa rede pode não ser único, o que no entanto não interfere no que se segue dado que pretendemos apenas calcular um qualquer conjunto de K trajectos mais curtos de s para t em $(\mathcal{N}, \mathcal{A})$.

O trajecto p_1 é uma árvore por definição – Figura 3(a). Consideremos agora os dois trajectos mais curtos, p_1 e p_2. O trajecto p_2 é constituído por uma parte inicial comum a p_1, o subtrajecto $\langle 1, 4 \rangle$, "desviando-se" de p_1 no segundo nó, 4. Estes dois trajectos formam a árvore representada na Figura 3(b). Analogamente, o terceiro trajecto mais curto, p_3, contém um subtrajecto comum a p_1 e um subtrajecto comum a p_2, iguais e constituídos apenas pelo nó inicial. Deste modo, p_3 "desvia-se" dos trajectos anteriores, p_1 e p_2, naquele nó – Figura 3(c). Por fim, p_4 coincide com p_1 em $\langle 1, 4 \rangle$, com p_2 em $\langle 1, 4, 3 \rangle$, e com p_3 em $\langle 1 \rangle$, ou seja, tem um subtrajecto inicial comum à árvore formada por p_1, p_2 e p_3, $\langle 1, 4, 3 \rangle$, "desviando-se" destes trajectos no nó 3 – Figura 3(d).

Na realidade a estrutura formada, tal como a apresentámos, não é a de uma árvore de acordo com a definição usual uma vez que alguns dos nós e arcos aparecem repetidos. No entanto, de modo a evitar estas situações, podemos utilizar uma função que a cada nó de um trajecto p_i, $i \in \{1, \ldots, K\}$, associa um número natural, por foma a que as imagens do mesmo nó em trajectos diferentes sejam também diferentes. No presente trabalho não será utilizada esta função para não sobrecarregar a notação, pelo que a distinção dos nós deverá ser feita mediante o contexto.

Denotaremos por $\mathcal{T}_k$, para $k \in \{1, \ldots, K\}$, a árvore constituída pelos k trajectos mais curtos de s para t em $(\mathcal{N}, \mathcal{A})$, $p_1, \ldots, p_k$. O resultado que se apresenta em seguida formaliza o conceito, introduzido de forma intuitiva, de árvore dos k trajectos mais curtos entre dois nós de uma rede.

Teorema 5 *Seja $k \in \{1, \ldots, K\}$; então o $(k+1)$-ésimo trajecto mais curto de s para t em $(\mathcal{N}, \mathcal{A})$ é da forma $p_{k+1} = sub_j(s, u) \diamond \langle u, v \rangle \diamond p$, para algum $j \in \{1, \ldots, k\}$, com $(u, v) \in (\mathcal{N}, \mathcal{A})$ não pertencente*

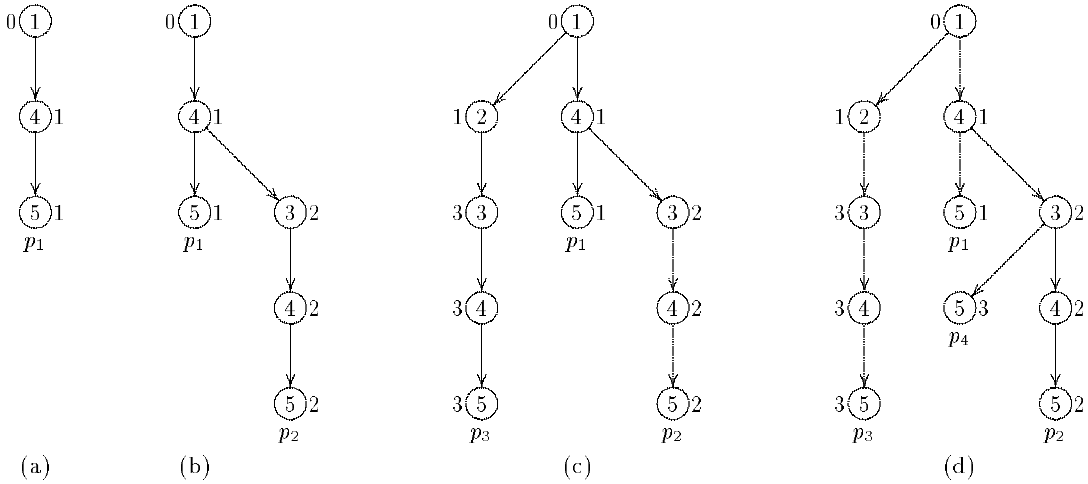

Figura 3: Árvores de trajectos mais curtos de 1 para 5

à árvore $\mathcal{T}_k$ *e onde* $p \in \mathcal{P}_{vt}$.

Demonstração: Utilizando indução sobre k comecemos por analisar o caso $k = 1$.

Sendo p_1 o trajecto mais curto de s para t e p_2 o segundo trajecto mais curto entre os mesmos nós, p_1 e p_2 são distintos. Seja então $u \neq t$ o nó de p_1 e p_2 mais distante (relativamente ao número de arcos) de s, tal que $\text{sub}_1(s,u) = \text{sub}_2(s,u)$ e, portanto sendo (u,v) o arco de p_2 com início em u, $\text{sub}_1(s,v) \neq \text{sub}_2(s,v) = \text{sub}_2(s,u) \diamond \langle u,v \rangle$. A árvore $\mathcal{T}_1$ é constituída apenas pelo trajecto p_1 e portanto não contém o arco (u,v). Concluímos pois que $p_2 = \text{sub}_j(s,u) \diamond \langle u,v \rangle \diamond p$, com $j = 1 < 2$ e $p \in \mathcal{P}_{vt}$.

Consideremos agora que, para todo o $i \leq k$, $p_i = \text{sub}_{j_i}(s,u_i) \diamond \langle u_i, v_i \rangle \diamond q_i$, com $j_i \in \{1, \ldots, i-1\}$ e (u_i, v_i) não pertencente a $\mathcal{T}_{i-1}$.

Seja p_{k+1} o $(k+1)$-ésimo trajecto mais curto de $\mathcal{P}$; então p_{k+1} coincide com cada p_j, para todo o $j \in \{1, \ldots, k\}$, desde o nó inicial até um nó x_j distinto do nó t uma vez que $p_{k+1} \notin \{p_1, \ldots, p_k\}$. De entre estes nós $x_1 \ldots, x_k$, denotemos por $u = x_j$, para algum $j \in \{1, \ldots, k\}$, aquele que está mais distante de s em p_{k+1}. Então $p_{k+1} = \text{sub}_j(s,u) \diamond \langle u,v \rangle \diamond p$, com $p \in \mathcal{P}_{vt}$, onde, de acordo com a escolha de u, (u,v) não está na árvore $\mathcal{T}_k$. $\square$

O nó u de p_k é denominado por nó desvio de p_k e é denotado por $v^k_{\alpha_k}$, onde α_k é a ordem respectiva no trajecto p_k (e p_j). O arco (u,v) é, geralmente, denominado arco desvio de p_k e p_j o seu trajecto pai. Por convenção, o nó desvio de p_1 é o nó inicial, s.

Com base na Figura 2 podemos verificar que os nós desvio de p_1, p_2, p_3 e p_4 são, respectivamente, $v^1_1 = 1, v^2_2 = 4, v^3_1 = 1$ e por fim $v^4_3 = 3$.

Deste modo a determinação dos K trajectos mais curtos pode ser feita através da construção de uma árvore.

4.2.2 Um algoritmo baseado na construção da árvore de trajectos

Ao estudar este problema, facilmente se conclui que o ideal seria determinar o menor número possível de trajectos, isto é, determinar somente os K trajectos pretendidos. No entanto, como veremos no que se segue, os algoritmos que propomos determinam uma sobreárvore da árvore, $\mathcal{T}_K$, dos K trajectos mais curtos.

O primeiro trajecto a determinar é o mais curto de s para t em $(\mathcal{N}, \mathcal{A})$, sendo para isso utilizado um qualquer algoritmo. A partir deste trajecto, p_1, como poderemos obter p_2? De acordo com o Teorema 5, p_2 é da forma $\text{sub}_1(s,u) \diamond \langle u,v \rangle \diamond p$, onde $(u,v) \notin p_1$ e $p \in \mathcal{P}_{vt}$, pelo que poderíamos tentar determinar todos os trajectos com esta forma, o mais curto dos quais seria p_2. Seguindo este raciocínio seria necessário, para cada nó $u \in p_1$, hipotético nó desvio de p_2, e cada arco $(u,v) \notin p$, conhecer todos os trajectos de $\mathcal{P}_{vt}$. Este procedimento pode ser melhorado se atendermos ao Teorema 6.

Dada a rede $(\mathcal{N},\mathcal{A})$ e o nó terminal, t, denotaremos por $\mathcal{T}_t$ a árvore dos caminhos mais curtos com raiz em t, constituída pelos caminhos mais curtos de todos os nós de $(\mathcal{N},\mathcal{A})$ para t. Nesta árvore, dado um nó u, denotamos por $\mathcal{T}_t(u)$ o caminho de u para t em $\mathcal{T}_t$. A Figura 4 representa a árvore $\mathcal{T}_t$ para a rede $(\mathcal{N},\mathcal{A})$ da Figura 2, associando a cada nó u o valor $c(\mathcal{T}_t(u))$.

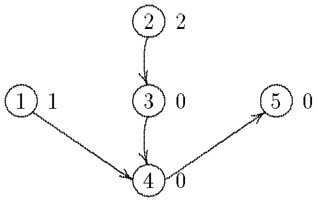

Figura 4: Árvore dos trajectos mais curtos com raiz em t

Teorema 6 *Seja $k \in \{1,\dots,K\}$; então p_{k+1} é da forma $sub_j(s, v^{k+1}_{\alpha_{k+1}}) \diamond \langle v^{k+1}_{\alpha_{k+1}}, v\rangle \diamond \mathcal{T}_t(v)$, para algum $j \in \{1,\dots,k\}$, onde $(v^{k+1}_{\alpha_{k+1}}, v)$ é um arco não pertencente à árvore $\mathcal{T}_k$.*

Demonstração: No Teorema 5 vimos que $p_{k+1} = \mathrm{sub}_j(s, v^{k+1}_{\alpha_{k+1}}) \diamond \langle v^{k+1}_{\alpha_{k+1}}, v\rangle \diamond p$, pelo que basta mostrar que p tem que ser um trajecto mais curto de v para t. Suponhamos que isto não acontecia, isto é, que $c(\mathcal{T}_t(v)) < c(p)$ e consideremos o trajecto $q = \mathrm{sub}_{k+1}(s,v) \diamond \mathcal{T}_t(v) \in \mathcal{P}$. Então, como o arco de q com início em $v^{k+1}_{\alpha_{k+1}}$ não está na árvore $\mathcal{T}_k$, $q \notin \{p_1,\dots,p_k\}$. Além disso

$$c(q) = c(\,\mathrm{sub}_{k+1}(s,v)) + c(\mathcal{T}_t(v)) < c(\,\mathrm{sub}_{k+1}(s,v)) + c(p) = c(p_{k+1}),$$

pelo que p_{k+1} não seria o $(k+1)$-ésimo trajecto mais curto de s para t, como queríamos mostrar. □

Deste modo, voltando à determinação de p_2 a partir de p_1, bastaria calcular para cada nó u de p_1 o trajecto $\mathrm{sub}_1(s,u) \diamond \langle u,v\rangle \diamond \mathcal{T}_t(v)$, para todos os arcos $(u,v) \notin p_1$. De acordo com este procedimento são gerados vários elementos de $\mathcal{P}$, o mais curto dos quais é p_2. Ou seja, se a partir de p_1 forem gerados estes trajectos, candidatos a p_2, e colocados num conjunto X, após a análise de todos os nós de p_1, o trajecto mais curto de entre os trajectos de X é p_2.

Este raciocínio, depois de generalizado de modo a determinar p_k, foi o utilizado no algoritmo proposto por Eppstein e baseado na árvore de trajectos, [6]. Tem no entanto o inconveniente de aumentar muito rapidamente o número de trajectos na árvore em construção, uma vez que, para cada nó u de um trajecto nessa árvore, são colocados todos os arcos não repetidos com início em u.

Para tentarmos diminuir o número de trajectos determinados a partir do nó u de p_1 basta observar que se u for de facto o nó desvio de p_2, então devemos escolher, entre todos aqueles trajectos, aquele que for o mais curto. Por outras palavras, basta determinar $\mathrm{sub}_1(s,u) \diamond \langle u,v\rangle \diamond \mathcal{T}_t(v)$, onde $v \in \mathcal{N}$ é tal que $(u,v) \notin p_1$, e minimiza o custo de $\langle u,v\rangle \diamond \mathcal{T}_t(v)$, isto é, minimiza $c_{uv} + c(\mathcal{T}_t(v))$. Deste modo, para cada nó u são analisados todos os arcos $(u,v) \notin p_1$, sendo escolhido um nó v, e consequentemente um único trajecto $\mathrm{sub}_1(s,u) \diamond \langle u,v\rangle \diamond \mathcal{T}_t(v)$. Os trajectos gerados são também acrescentados ao conjunto X de candidatos.

É de notar que, num passo k, o conjunto $X \cup \{p_1,\dots,p_k\}$ contém todos os trajectos calculados até então, ou seja, contém os trajectos da árvore construída pelo algoritmo até àquele momento, constituída pelos k trajectos mais curtos já enumerados e por alguns candidatos a j-ésimos trajectos mais curtos, com $j > k$.

A enumeração de trajectos é conseguida a partir de sucessivas escolhas (e consequente remoção) de um trajecto no conjunto X. Deste modo, apesar dos trajectos não serem calculados por ordem dos seus custos, repetindo este procedimento, num passo genérico k, com $k \in \{1,\dots,K\}$, será escolhido o elemento de X com menor custo, que será o k-ésimo trajecto mais curto de s para t em $(\mathcal{N}, \mathcal{A})$. Ao retirar p_k do conjunto X devem ser gerados novos trajectos, de forma análoga à indicada para $k = 1$, que serão candidatos a p_j, com $j > k$.

De modo a que um dado trajecto não seja determinado mais do que uma vez, ao escolher p_k no conjunto X são analisados apenas os nós que se seguem ao nó desvio, inclusivé, e, para cada um destes nós, digamos u, procura-se $v \in \mathcal{N}$ tal que $(u, v) \in \mathcal{A}$, não pertence à árvore construída até ao momento e que, nestas condições, minimiza o custo do trajecto $\langle u, v\rangle \diamond \mathcal{T}_t(v)$.

Denotemos por $\mathcal{A}_{\mathcal{T}_k}(u)$ o conjunto dos arcos da árvore dos trajectos no passo k do algoritmo com início no nó u. Pretendemos pois determinar outro nó v tal que $(u, v) \notin \mathcal{A}_{\mathcal{T}_k}(u)$ e v minimiza $c_{uv} + c(\mathcal{T}_k(v))$. Mais uma vez, cada trajecto obtido deste modo é acrescentado ao conjunto X. É de notar que quando $k = 1$ o único trajecto na árvore é p_1, pelo que $(u, v) \notin \mathcal{A}_{\mathcal{T}_k}(u)$ se e somente se $(u, v) \notin p_1$, como havíamos exigido.

Os passos principais do algoritmo acima descrito são apresentados resumidamente em seguida.

Algoritmo 1: *Algoritmo baseado na construção da árvore dos trajectos*

Determinar $\mathcal{T}_t$
$k \longleftarrow 0$
$X \longleftarrow \{\mathcal{T}_t(s)\}$
`Enquanto` $((k \leq K)$ e $(X \neq \emptyset))$ `Fazer`
  $k \longleftarrow k + 1$
  $p_k \longleftarrow$ trajecto com menor custo em X
  $\alpha_k \longleftarrow$ ordem do nó desvio, $v^k_{\alpha_k}$, de p_k
  $X \longleftarrow X - \{p_k\}$
  `Para` (todo o $u \in \mathrm{sub}_k(v^k_{\alpha_k}, t)$) `Fazer`
    `Se` $(\mathcal{A} - \mathcal{A}_{\mathcal{T}_k}(u) \neq \emptyset)$ `Então`
      $v \longleftarrow$ nó tal que $c_{uv} + c(\mathcal{T}_t(v))$ é mínimo em $\mathcal{A} - \mathcal{A}_{\mathcal{T}_k}(u)$
      $p \longleftarrow \mathrm{sub}_k(s, u) \diamond \langle u, v\rangle \diamond \mathcal{T}_t(v)$
      $X \longleftarrow X \cup \{p\}$
    `FimSe`
  `FimPara`
`FimEnquanto`

Nas Figuras 5 e 6 encontram-se representadas as árvores de trajectos construídas no final dos primeiros quatro passos do Algoritmo 1, quando aplicado à rede da Figura 2.

Como referimos, o primeiro trajecto determinado, p_1, é o mais curto de s para t, escolhido quando $k = 1$. Sendo o seu nó desvio $v^1_1 = 1$, serão analisados sucessivamente os nós 1, 4 e 5. A partir do nó 1 apenas podemos escolher o arco $(1, 2)$ (uma vez que $(1, 4)$ pertence a p_1) e então o novo trajecto, $q_1 = \langle 1, 2\rangle \diamond \mathcal{T}_t(2) = \langle 1, 2, 3, 4, 5\rangle$ é acrescentado ao conjunto X. Em seguida é analisado o nó 4, a partir do qual somos levados a escolher o único arco possível, $(4, 3)$, obtendo então o trajecto $q_2 = \langle 1, 4\rangle \diamond \langle 4, 3\rangle \diamond \mathcal{T}_t(3) = \langle 1, 4, 3, 4, 5\rangle$. A partir de $t = 5$ não é gerado qualquer trajecto, dado não existirem arcos com início neste nó. No final deste passo a árvore determinada contém os trajectos p_1, q_1 e q_2, e $X = \{q_1, q_2\}$. No passo seguinte $k = 2$ e p_2 é o trajecto mais curto em X, isto é, $p_2 = \langle 1, 4, 3, 4, 5\rangle$, cujo nó desvio é $v^2_2 = 4$. Depois de analisado o nó 4 é analisado o nó 3, a partir do qual podemos escolher $\langle 4, 2\rangle \diamond \mathcal{T}_t(2)$ ou $\langle 3, 5\rangle$, com custos 3 ou 1, respectivamente. Uma vez que o mais curto entre estes é $\langle 3, 5\rangle$, o novo trajecto será $\langle 1, 4, 3, 5\rangle$. O algoritmo continua do mesmo modo, até serem escolhidos K trajectos em X.

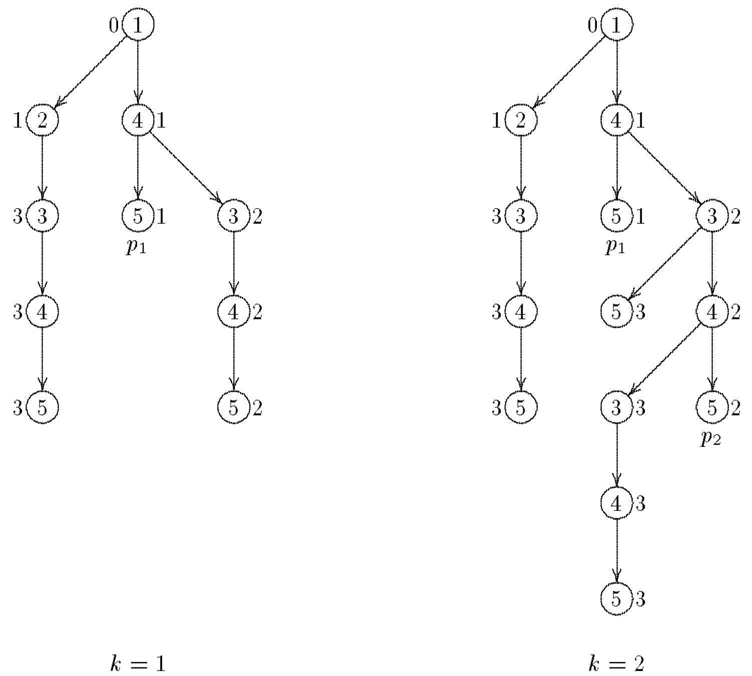

Figura 5: Árvores de trajectos obtidas no final dos passos 1 e 2

4.2.3 O algoritmo MPS

O algoritmo que apresentamos nesta subsecção, também ele novo, combina o algoritmo anterior com uma mudança de variável e uma diferente representação da rede $(\mathcal{N}, \mathcal{A})$. O objectivo desta combinação é construir exactamente a mesma árvore obtida com o algoritmo da subsecção anterior, mas fazê-lo de forma mais eficiente. Para obter mais pormenores acerca deste algoritmo sugerimos a consulta de [6] ou [10]. Na última destas referências não só se aborda todo o algoritmo mas também a forma utilizada para representação de $(\mathcal{N}, \mathcal{A})$; na primeira pode ver-se mais pormenorizadamente a mudança de variável que é realizada.

Mudança de variável

Voltemos a considerar uma árvore dos trajectos mais curtos de todos os nós para t, $\mathcal{T}_t$, e dado $u \in \mathcal{N}$ denotemos por π_u^t o custo do trajecto de u para t na árvore $\mathcal{T}_t$. Com base em $\mathcal{T}_t$ associemos a cada $(i,j) \in \mathcal{A}$ o respectivo custo reduzido, definido por $\bar{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij}$. Utilizando estes valores definimos custo reduzido de um trajecto, de modo análogo ao custo de um trajecto, por $\bar{c}(p) = \sum_p \bar{c}_{ij}$.

A Figura 7 representa a rede $(\mathcal{N}, \mathcal{A})$ da Figura 2, associando-se a cada arco o respectivo custo reduzido. Sendo $p = \langle 1, 2, 5 \rangle$ um trajecto de 1 para 5 em $(\mathcal{N}, \mathcal{A})$, de acordo com a definição apresentada, o custo reduzido de p é $\bar{c}(p) = \bar{c}_{12} + \bar{c}_{25} = 3$.

É possível demonstrar que é indiferente enumerar trajectos quer utilizando os respectivos custos quer os custos reduzidos, o que é feito seguidamente, no Lema 1 e no Teorema 7.

Lema 1 *Seja p um trajecto qualquer de $\mathcal{P}$ e p_1 um trajecto mais curto, também elemento de $\mathcal{P}$. Então $\bar{c}(p) = c(p) - c(p_1)$.*

Demonstração: Seja $p = \langle v_1 = s, \ldots, v_\ell = t \rangle \in \mathcal{P}$, então

$$\begin{aligned}\bar{c}(p) &= \textstyle\sum_p \bar{c}_{ij} = \sum_p(\pi_j^t - \pi_i^t + c_{ij}) = \pi_{v_2}^t - \pi_{v_1}^t + c_{v_1 v_2} + \pi_{v_3}^t - \pi_{v_2}^t + c_{v_2 v_3} + \cdots \pi_{v_\ell}^t - \pi_{v_{\ell-1}}^t + c_{v_{\ell-1} v_\ell} \\ &= c(p) + \pi_t^t - \pi_s^t.\end{aligned}$$

Sendo p_1 um trajecto mais curto em $\mathcal{P}$, $c(p_1) = \pi_s^t - \pi_t^t$, e então $\bar{c}(p) = c(p) - c(p_1)$, como pretendíamos mostrar. □

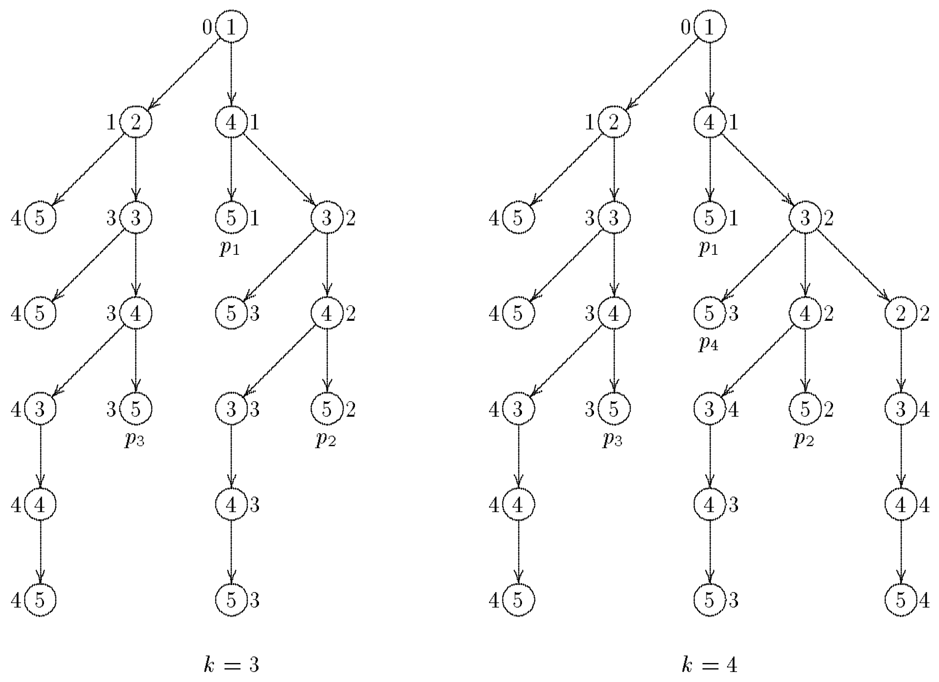

Figura 6: Árvores de trajectos obtidas no final dos passos 3 e 4

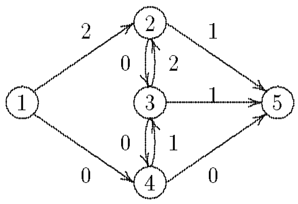

Figura 7: Rede $(\mathcal{N}, \mathcal{A})$ com custos reduzidos

Teorema 7 *Sejam p e q dois trajectos quaisquer de $\mathcal{P}$. Então:*

1. $\bar{c}(p) = \bar{c}(q)$ *se e somente se* $c(p) = c(q)$;
2. $\bar{c}(p) < \bar{c}(q)$ *se e somente se* $c(p) < c(q)$.

Demonstração: As demonstrações dos pontos 1 e 2 são análogas, pelo que apresentaremos apenas a demonstração de 1.

Consideremos pois dois trajectos $p, q \in \mathcal{P}$; então, pelo Lema 1, $\bar{c}(p) = \bar{c}(q)$ se e somente se $c(p) + c(p_1) = c(q) + c(p_1)$ e $c(p) + c(p_1) = c(q) + c(p_1)$ se e só se $c(p) = c(q)$, como queríamos demonstrar. □

Deste modo, é possível enumerar trajectos, quer pelos seus custos quer pelos seus custos reduzidos, pelo que é também possível aplicar o algoritmo anterior à rede $(\mathcal{N}, \mathcal{A})$ realizando a mudança de variável $\bar{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij}$, para todos os arcos $(i, j) \in \mathcal{A}$. De facto, assim continuamos a determinar os K trajectos com menor custo reduzido, e consequentemente os K trajectos com menor custo. Mas, qual a utilidade deste procedimento? Isto é, existirá alguma vantagem na utilização de custos reduzidos em vez da utilização dos custos usuais de cada um dos arcos? Para responder a esta questão comecemos por notar que os custos reduzidos verificam algumas propriedades importantes, que enunciamos e demonstramos no Teorema 8.

Teorema 8 *Seja $\bar{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij}$, o custo reduzido associado a $\mathcal{T}_t$, para todo o $(i,j) \in \mathcal{A}$. Então:*

1. *$\bar{c}_{ij} \geq 0$, para todo o $(i,j) \in \mathcal{A}$;*
2. *$\bar{c}_{ij} = 0$, para todo o $(i,j) \in \mathcal{A} \cap \mathcal{T}_t$.*

Demonstração:

1. Seja $(i,j) \in \mathcal{A}$ e sejam $\mathcal{T}_t(i)$ e $\mathcal{T}_t(j)$, respectivamente, trajectos mais curtos de i para t e de j para t em $(\mathcal{N},\mathcal{A})$ (e simultaneamente na árvore $\mathcal{T}_t$). Então, sendo $\langle i,j\rangle \diamond \mathcal{T}_t(j)$ também um trajecto de i para t, $c(\langle i,j\rangle \diamond \mathcal{T}_t(j)) = c_{ij} + \pi_j^t \geq \pi_i^t = c(\mathcal{T}_t(i))$, portanto $\bar{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij} \geq 0$.
2. Consideremos agora que é dado $(i,j) \in \mathcal{A}$, pertencente também a $\mathcal{T}_t$. Então $\langle i,j\rangle \diamond \mathcal{T}_t(j) = \mathcal{T}_t(i)$ e $c_{ij} + \pi_j^t = \pi_i^t$, logo $\bar{c}_{ij} = 0$. □

Como consequência deste Teorema temos imediatamente o seguinte resultado.

Corolário 8.1 *Seja p um trajecto qualquer pertencente a $\mathcal{T}_t$; então $\bar{c}(p) = 0$.*

Com base neste Corolário podemos responder à questão que atrás foi colocada. De facto, no algoritmo apresentado na subsecção anterior, dado um nó u do k-ésimo trajecto mais curto é necessário encontrar um nó v no conjunto $\mathcal{A} - \mathcal{A}_{\mathcal{T}_k}(u)$ que minimize $c_{uv} + c(\mathcal{T}_t(v))$. Utilizando custos reduzidos em substituição dos custos usuais, isto é o mesmo que minimizar $\bar{c}_{uv} + \bar{c}(\mathcal{T}_t(v))$. Como $\mathcal{T}_t(v)$ é um trajecto de $\mathcal{T}_t$, então $c(\mathcal{T}_t(v)) = 0$, bastando portanto encontrar $v \in \mathcal{A} - \mathcal{A}_{\mathcal{T}_k}(u)$ que minimize $\bar{c}_{uv}$. Podemos então concluir que a utilização dos custos reduzidos dos arcos da rede no Algoritmo 1 simplifica realmente a determinação de cada nó v neste algoritmo.

Representação da rede na *sorted forward star form*

Como vimos, em cada passo do algoritmo são analisados todos os nós u de p_k que se seguem ou coincidem com o seu nó desvio e, para cada u, é necessário determinar $v \in \mathcal{A}$ tal que $(u,v) \in \mathcal{A}$ tenha um custo reduzido mínimo de entre os sucessores de arcos com início em u que não se encontrem na árvore construída até ao momento. Além disso, como se demonstra em seguida, $\bar{c}_{ij}$ representa o custo a acrescentar ao trajecto obtido do trajecto mais curto de i para t, quando se obriga a que seja utilizado o arco (i,j).

Teorema 9 *Seja $(i,j) \in \mathcal{A}$ e $p = \langle i,j\rangle \diamond \mathcal{T}_t(j)$; então $c(p) = \bar{c}_{ij} + c(\mathcal{T}_t(i))$.*
Demonstração: Seja $p = \langle i,j\rangle \diamond \mathcal{T}_t(j)$, para $(i,j) \in \mathcal{A}$. Então,
$$c(p) = c_{ij} + c(\mathcal{T}_t(j)) = \pi_i^t - \pi_j^t + \bar{c}_{ij} + \pi_j^t = \bar{c}_{ij} + \pi_i^t = \bar{c}_{ij} + c(\mathcal{T}_t(i)),$$
como pretendíamos demonstrar. □

Podemos deste modo concluir que, ao analisar um nó u, é de toda conveniência escolher "mais cedo" sucessores de arcos com início em u que tenham um menor custo reduzido. Com esse objectivo podemos representar $(\mathcal{N},\mathcal{A})$ na *sorted forward star form*, [6]. Nesta forma, o conjunto $\mathcal{A}$ encontra-se ordenado pelos antecessores dos arcos da rede, e os arcos com iguais antecessores encontram-se ordenados por ordem não decrescente dos respectivos custos reduzidos.

Utilizando a *sorted forward star form* para a rede representada na Figura 7 obteríamos o conjunto de arcos $\mathcal{A} = \{(1,4),(1,2),(2,3),(2,5),(3,4),(3,5),(3,2),(4,5),(4,3)\}$.

Considerando $(\mathcal{N},\mathcal{A})$ na *sorted forward star form*, um nó u de p_k e sendo (u,x) o arco de p_k com início em u, ao analisar u será escolhido o nó v tal que (u,v) é o arco de $(\mathcal{N},\mathcal{A})$ (representada na *sorted forward star form*) que se segue a (u,x), se tal existir. O arco (u,v) será aquele que tem custo reduzido imediatamente superior, ou eventualmente igual, ao de (u,x).

Algoritmo

Como referimos, o algoritmo MPS segue os passos do Algoritmo 1, utilizando não só custos reduzidos em vez dos custos usuais, mas também a rede $(\mathcal{N}, \mathcal{A})$ representada na *sorted forward star form*. Os passos principais deste algoritmo são apresentados resumidamente em seguida.

Algoritmo 2: *Algoritmo de Martins, Pascoal e Santos (MPS)*

Determinar $\mathcal{T}_t$
`Para` (todo o $(i,j) \in \mathcal{A}$) `Fazer` $\bar{c}_{ij} \longleftarrow \pi_j^t - \pi_i^t + c_{ij}$
Colocar $(\mathcal{N}, \mathcal{A})$ na *sorted forward star form*
$k \longleftarrow 0$
$X \longleftarrow \{\mathcal{T}_t(s)\}$
`Enquanto` $((k \leq K)$ e $(X \neq \emptyset))$ `Fazer`
  $k \longleftarrow k+1$
  $p_k \longleftarrow$ trajecto com menor custo reduzido em X
  $\alpha_k \longleftarrow$ ordem do nó desvio, $v^k_{\alpha_k}$, de p_k
  $X \longleftarrow X - \{p_k\}$
  `Para` (todo o nó $u \in \mathrm{sub}_k(v^k_{\alpha_k}, t)$) `Fazer`
    $v \longleftarrow$ nó tal que (u,v) é o arco que se segue ao arco com início em u de p_k
    `Se` (v está definido) `Então`
      $p \longleftarrow \mathrm{sub}_k(s,u) \diamond \langle u,v\rangle \diamond \mathcal{T}_t(v)$
      $X \longleftarrow X \cup \{p\}$
    `FimSe`
  `FimPara`
`FimEnquanto`

A árvore obtida com este algoritmo coincide, como foi referido, com a que é obtida pelo Algoritmo 1, sendo no entanto construída de forma mais eficiente.

5 A enumeração de caminhos

Podendo o Problema dos K Caminhos mais Curtos ser considerado um subproblema do Problema dos K Trajectos mais Curtos, poderíamos também ser levados a sugerir, para aquele problema, o tipo de algoritmos e a classificação que foi mencionada na secção anterior. No entanto, os problemas são suficientemente diferentes para que não possa ser feita uma tal analogia. De facto, como vimos atrás, no Problema dos K Caminhos mais Curtos os trajectos determinados devem restringir-se apenas a caminhos, isto é, a trajectos sem repetição de nós. Isto reflecte-se, desde logo, nas condições em que cada um dos problemas verifica o Princípio de Optimalidade.

Analogamente ao que foi feito na secção 4, $p_k = \langle s = v_1^k, \ldots, t = v^k_{\ell_k}\rangle$ e $\mathrm{sub}_k(i,j)$ denotarão o k-ésimo caminho mais curto de s para t e o subcaminho de p_k do nó i para o nó j, respectivamente.

5.1 O Princípio de Optimalidade

Como veremos em seguida, ao contrário do que acontece para $K = 1$ em que podemos afirmar que o Princípio de Optimalidade é satisfeito se $(\mathcal{N}, \mathcal{A})$ não contiver ciclos negativos, mesmo sob estas condições não é possível garantir o mesmo para $K > 1$.

Comecemos, como foi feito para trajectos, por considerar o caso $K = 1$. Sob certas condições podemos garantir, como se demonstra em seguida, que o Problema do Caminho mais Curto verifica o Princípio de Optimalidade.

Teorema 10 *Se* $(\mathcal{N}, \mathcal{A})$ *não contém ciclos de custo negativo, então todos os caminhos mais curtos de* s *para* t *em* $(\mathcal{N}, \mathcal{A})$*, são constituídos por subcaminhos mais curtos.*

Demonstração: Sabemos já (ver secção 3) que o Problema do Caminho mais Curto é finito qualquer que seja a rede considerada. Suponhamos que existe um caminho mais curto de s para t, p, e que dados u e v, dois nós de p, o subtrajecto $\text{sub}_p(u,v)$ não é óptimo. Então existe um caminho $q \in \mathcal{P}_{uv}$ tal que $c(q) < c(\text{sub}_p(u,v))$. A partir da concatenação dos caminhos $\text{sub}_p(s,u)$, q e $\text{sub}_p(v,t)$ obtemos o trajecto de s para t, $q' = \text{sub}_p(s,u) \diamond q \diamond \text{sub}_p(v,t)$.

Usando um raciocínio análogo ao da demonstração do Teorema 3, $c(q') < c(p)$. Se q' for um caminho fica demonstrado o pretendido; senão q' tem nós repetidos. Mas então é possível retirar os ciclos de q' obtendo-se um caminho, q'', tal que, sendo por hipótese $c(\mathcal{C}) \geq 0$ para todo o ciclo $\mathcal{C}$ de $(\mathcal{N}, \mathcal{A})$, o seu custo é não superior ao de q', isto é, $c(q'') \leq c(q') < c(p)$; portanto p não seria o caminho mais curto. $\square$

A implicação contrária pode ou não ser verdadeira. De facto, na rede representada da Figura 8(a) o ciclo $\mathcal{C} = \langle 4,3,2,4 \rangle$ tem custo -1. Considerando $s = 1$ e $t = 4$ o caminho mais curto é $\langle 1,2,4 \rangle$ que contém o subcaminho $\langle 1,2 \rangle$ de custo 0. No entanto, $\langle 1,4,3,2 \rangle$ é também caminho de 1 para 2 e tem custo menor que o anterior.

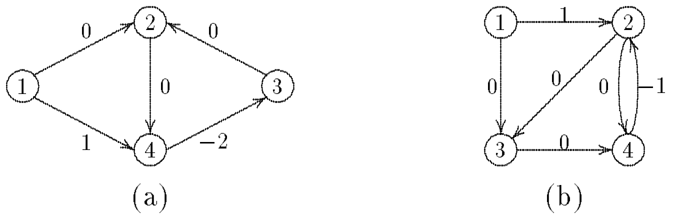

Figura 8: Redes

Usando agora a rede da Figura 8(b), também com um ciclo negativo $\mathcal{C} = \langle 2,4,2 \rangle$, o caminho mais curto de 1 para 4 é $\langle 1,3,4 \rangle$, constituído apenas por subcaminhos óptimos.

Assim, uma vez que estamos a considerar somente redes sem ciclos negativos, os algoritmos de rotulação referidos para trajectos podem ser utilizados também para a determinação do caminho mais curto.

Para $K > 1$ diz-se que o Problema dos K Caminhos mais Curtos verifica o Princípio de Optimalidade quando todo o k-ésimo caminho mais curto de s para t em $(\mathcal{N}, \mathcal{A})$, $k \in \{1, \ldots, K\}$, é constituído por j-ésimos subcaminhos mais curtos, com $j \in \{1, \ldots, k\}$.

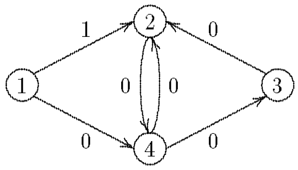

Figura 9: Rede

Como referimos atrás, sendo $K > 1$, este princípio pode não ser satisfeito. Um exemplo disto pode ser obtido atendendo a que o caminho mais curto de $s = 1$ para $t = 4$ na rede $(\mathcal{N}, \mathcal{A})$ da Figura 9 é $p_1 = \langle 1,4 \rangle$, enquanto que o segundo é $p_2 = \langle 1,2,4 \rangle$. No entanto existem em $(\mathcal{N}, \mathcal{A})$ três caminhos de 1 para 2, nomeadamente e por ordem não decrescente de custos, $t_1 = \langle 1,4,2 \rangle$, $t_2 = \langle 1,4,3,2 \rangle$ e $t_3 = \langle 1,2 \rangle$. Daqui concluímos que o terceiro caminho mais curto de 1 para 2 é subcaminho de p_2, ou seja, este problema não verifica o Princípio de Optimalidade.

5.2 Árvore dos K caminhos mais curtos

Sendo um caminho também um trajecto, o conjunto dos caminhos entre s e t em $(\mathcal{N},\mathcal{A})$ é um subconjunto de $\mathcal{P}$. Assim, K caminhos mais curtos entre dois nós de uma rede formam uma árvore análoga à do problema anterior.

Consideremos novamente a rede representada na Figura 2. A árvore dos seis caminhos mais curtos de 1 para 5 em $(\mathcal{N},\mathcal{A})$ encontra-se representada na Figura 10.

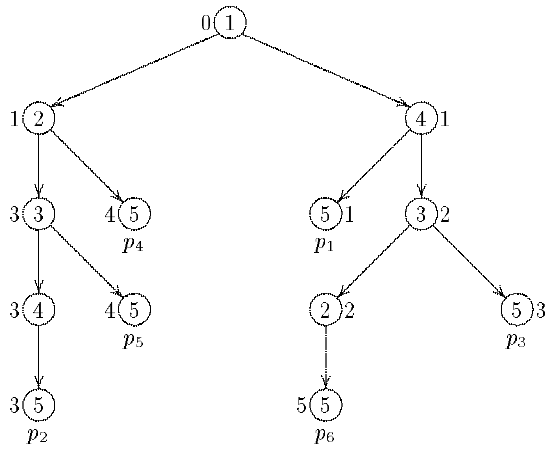

Figura 10: Árvore de caminhos de 1 para 5

Deste modo, é possível utilizar esta estrutura para a obtenção de algoritmos que enumeram os K caminhos mais curtos entre dois nós de uma rede.

5.3 Adaptação de algoritmos baseados na construção da árvore

Analogamente ao que foi feito para o Problema dos K Trajectos mais Curtos, o nosso objectivo é desenvolver algoritmos que construam uma árvore que contenha os K caminhos mais curtos entre dois nós, s e t, de $(\mathcal{N},\mathcal{A})$.

Um algoritmo conhecido para a resolução do Problema dos K Caminhos mais Curtos, e que podemos considerar ser baseado na determinação da árvore dos caminhos mais curtos, é o de Yen, [13]. De facto, sendo X um conjunto de candidatos a k-ésimo caminho mais curto, no passo k é escolhido o caminho mais curto em X, p_k. A partir deste são analisados os nós seguintes ou coincidentes com o seu nó desvio e são gerados novos caminhos que são desvios de p_k, guardados em seguida no conjunto X. Um vez que se pretendem determinar somente caminhos, ao analisar cada nó u de p_k são apagados de $(\mathcal{N},\mathcal{A})$ todos os nós e arcos que não podem ser repetidos, determinando-se posteriormente o caminho mais curto de u para t, p, na rede alterada. Deste modo, o novo trajecto será $\mathrm{sub}_k(s,u)\diamond p$ que, pela forma como é calculado, é também um caminho.

Dois inconvenientes deste algoritmo são desde logo a necessidade, em cada um dos seus passos, de:

- alterar a rede $(\mathcal{N},\mathcal{A})$, alterações essas a repor posteriormente;
- resolver um Problema do Caminho mais Curto na rede alterada.

É de notar que o Algoritmo 1 pode ser visto como uma generalização do algoritmo de Yen, onde é admitida a determinação de trajectos com e sem ciclos.

Tentaremos em seguida adaptar o algoritmo MPS, apresentado na subsecção 4.2.3 para a enumeração de trajectos, com vista à resolução deste problema.

5.3.1 Adaptação do algoritmo MPS

Como vimos, neste algoritmo começamos por determinar $\mathcal{T}_t$, a árvore dos trajectos mais curtos de todos os nós para t, trajectos estes que são também caminhos, como foi referido na subsecção 3.2. Deste modo determinamos $p_1 = \mathcal{T}_t(s)$, caminho mais curto de s para t em $(\mathcal{N}, \mathcal{A})$.

Ao gerar candidatos ao segundo caminho mais curto obtemos, de acordo com o mesmo algoritmo, trajectos da forma $\mathrm{sub}_1(s,u) \diamond \langle u, v\rangle \diamond \mathcal{T}_t(v)$. Apesar de quer $\mathrm{sub}_1(s,u)$ quer $\langle u, v\rangle$ e $\mathcal{T}_t(v)$ serem caminhos, facilmente se verifica que a sua concatenação pode não o ser, isto é, pode conter ciclos. Para isso basta notar que na rede representada na Figura 9, quer $\langle 1,2,4\rangle$ quer $\langle 4,2,3\rangle$ são caminhos e, no entanto, a sua concatenação, $\langle 1,2,4,2,3\rangle$ contém o ciclo $\langle 2,4,2\rangle$, pelo que não é um caminho.

Assim, em cada passo do algoritmo ao escolher, p', o trajecto com menor custo reduzido no conjunto de candidatos a k-ésimos caminhos mais curtos, para $k \in \{2, \ldots, K\}$, este pode não ser um caminho. No entanto, ainda que p' contenha ciclos, deverá ser analisado uma vez que a partir dele podem eventualmente vir a ser gerados outros caminhos. Deste modo analisam-se os nós de p' seguintes ou coincidentes com o respectivo nó desvio, $v'_{\alpha'}$, mas anteriores ao primeiro nó que forma um ciclo (caso contrário seriam gerados trajectos que saberíamos à partida conterem ciclos). Podemos assim reduzir o número de trajectos com ciclos determinados pela adaptação do algoritmo MPS. Se o último nó analisado for o terminal, então podemos concluir que p' não contém ciclos, sendo portanto k-ésimo caminho mais curto de s para t em $(\mathcal{N}, \mathcal{A})$, para algum $k \in \{1, \ldots, K\}$.

Além disso, ao analisar um nó u de p' escolhendo para v um nó de $\mathrm{sub}_{p'}(s,u)$, sabemos que obteremos um trajecto com ciclo pelo que vamos evitar fazê-lo. Ou seja, o nó v escolhido será tal que (u,v) é o arco que se segue ao arco com início em u de p' na rede $(\mathcal{N}, \mathcal{A})$ representada na *sorted forward star form* e tal que $v \notin \mathrm{sub}_{p'}(s,u)$.

Os passos principais desta adaptação encontram-se resumidos no Algoritmo 3.

Algoritmo 3: *Adaptação do algoritmo de Martins, Pascoal e Santos para a enumeração de caminhos*

Determinar $\mathcal{T}_t$
`Para` (todo o $(i,j) \in \mathcal{A}$) `Fazer` $\bar{c}_{ij} \longleftarrow \pi^t_j - \pi^t_i + c_{ij}$
Colocar $(\mathcal{N}, \mathcal{A})$ na *sorted forward star form*
$k \longleftarrow 0$
$X \longleftarrow \{\mathcal{T}_t(s)\}$
`Enquanto` $((k \leq K)$ e $(X \neq \emptyset))$ `Fazer`
  $p' \longleftarrow$ trajecto com menor custo reduzido em X
  $\alpha' \longleftarrow$ ordem do nó desvio, $v'_{\alpha'}$, de p'
  $X \longleftarrow X - \{p'\}$
  `Para` (todo o $u \in \mathrm{sub}_{p'}(v'_{\alpha'}, t)$ tal que $\mathrm{sub}_{p'}(s,u)$ é um caminho) `Fazer`
    $v \longleftarrow$ nó tal que (u,v) é o arco que se segue ao arco com início em u de p' e $v \notin \mathrm{sub}_{p'}(s,u)$
    `Se` (v está definido) `Então`
      $p \longleftarrow \mathrm{sub}_{p'}(s,u) \diamond \langle u,v\rangle \diamond \mathcal{T}_t(v)$
      $X \longleftarrow X \cup \{p\}$
    `FimSe`
    `Se` $(u = t)$ `Então`
      $k \longleftarrow k + 1$
      $p_k \longleftarrow p'$
    `FimSe`
  `FimPara`
`FimEnquanto`

Com um raciocínio idêntico poderíamos adaptar outros algoritmos para determinação dos K trajectos mais curtos que se baseiem na construção de uma árvore que os contém, nomeadamente o

Algoritmo 1 ou o algoritmo de Eppstein, [6].

6 Complexidade e experiência computacional

De entre os algoritmos referidos para o Problema dos K Trajectos mais Curtos, aquele que apresenta a melhor complexidade, no pior dos casos, é o de Eppstein, com $\mathcal{O}(m+n\log n+K\log K)$, [10]. Relativamente aos novos algoritmos apresentados, podemos afirmar que, no pior dos casos, o Algoritmo 1 tem uma complexidade de $\mathcal{O}(Km)$ (ver [10]), enquanto que o Algoritmo 2 tem complexidade de $\mathcal{O}(Kn+m)$ (ver também [10]). Relativamente ao algoritmo de Yen podemos afirmar que, no pior dos casos, são resolvidos Kn problemas do caminho mais curto.

Quanto à adaptação do algoritmo MPS para o Problema dos K Caminhos mais Curtos, uma vez que em geral não conhecemos, à partida, um limite superior para o número de trajectos que é necessário determinar neste algoritmo, não foi possível estabelecer uma ordem de complexidade polinomial. No entanto, testes computacionais efectuados têm revelado um bom comportamento deste algoritmo na resolução de problemas.

Alguns dos algoritmos, quer para trajectos quer para caminhos, foram testados computacionalmente. Os resultados podem ver-se em [8, 9, 10]. Dos testes efectuados concluiu-se que o algoritmo MS parece ser o mais eficiente para a enumeração de trajectos, enquanto que o algoritmo MPS se mostrou bastante eficiente relativamente à enumeração de caminhos. Um estudo computacional mais completo será assunto de um futuro trabalho.

A autora pertence à Linha *Ciências da Computação* do Centro de Informática e Sistemas da Universidade de Coimbra, parcialmente subsidiado pelo Ministério das Ciências e Tecnologia através do Projecto PRAXIS XXI da Junta Nacional de Investigação Científica e Tecnológica.

Ao meu orientador, Prof. Doutor Ernesto de Queirós Vieira Martins, agradeço o incansável apoio e a colaboração prestada na elaboração da minha tese de mestrado bem como na elaboração deste artigo.

Referências

[1] J.A. Azevedo, M.E.O.S. Costa, J.J.E.R.S. Madeira, and E.Q.V. Martins. An algorithm for the ranking of shortest paths. *European Journal of Operational Research*, 69:97–106, 1993.

[2] J.A. Azevedo, J.J.E.R.S. Madeira, E.Q.V. Martins, and F.M.A. Pires. A shortest paths ranking algorithm, 1990. Proceedings of the Annual Conference AIRO'90, Models and Methods for Decision Support, Operational Research Society of Italy, 1001-1011.

[3] J.A. Azevedo, J.J.E.R.S. Madeira, E.Q.V. Martins, and F.M.A. Pires. A computational improvement for a shortest paths ranking algorithm. *European Journal of Operational Research*, 73:188–191, 1994.

[4] B.V. Cherkassky, A.V. Goldberg, and T. Radzik. Shorthest paths algorithms: Theory and experimental evaluation. *Mathematical Programming*, 73:129–196, 1996.

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

[6] D. Eppstein. Finding the k shortest paths. SIAM Journal on Computing (aceite para publicação).

[7] E.Q.V. Martins, M.M.B. Pascoal, D.M.L. Rasteiro, and J.L.E. Santos. The optimal path problem. Submetido, 1998.

[8] E.Q.V. Martins, M.M.B. Pascoal, and J.L.E. Santos. A new algorithm for ranking loopless paths. Submetido, 1997.

[9] E.Q.V. Martins and J.L.E. Santos. A new shortest paths ranking algorithm. Submetido, 1996.

[10] M.M.B. Pascoal. Algoritmos para a enumeração dos k trajectos mais curtos, 1997. (Dissertação de Mestrado; Departamento de Matemática; Universidade de Coimbra).

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

[12] D. Shier. On algorithms for finding the k shortest paths in a network. *Networks*, 9:195–214, 1979.

[13] J.Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17:712–716, 1971.