ALGORITMOS DOS DESVIOS PARA A ENUMERAÇÃO DOS K TRAJECTOS MAIS CURTOS

MARTA PASCOAL†

marta@mat.uc.pt

CENTRO DE INFORMÁTICA E SISTEMAS DA UNIVERSIDADE DE COIMBRA
DEPARTAMENTO DE MATEMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

IO2000, Setúbal, Abril de 2000

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.

http://www.mat.uc.pt/~marta/Publicacoes/io2000_2.ps.gz

†Sob a orientação do Professor Doutor Ernesto de Queirós Vieira Martins

Resumo

Algoritmos dos desvios para a enumeração dos K trajectos mais curtos

Nesta comunicação é feita uma retrospectiva descritiva dos algoritmos dos desvios, quer para a enumeração dos K trajectos mais curtos quer para a enumeração dos K caminhos mais curtos entre um par de nós de uma rede.

É ainda apresentada experiência computacional relativa aos algoritmos dos desvios descritos, bem como testes comparativos entre algoritmos dos desvios e algoritmos de rotulação.

Índice

Introdução 1

Problema 1

Subproblemas 1

Árvore dos K trajectos mais curtos 2

Algoritmos dos desvios 4

Algoritmo de Hoffman e Pavley (1957) 6

Algoritmo de Yen – Enumeração de caminhos (1974) 8

Algoritmo elementar – Martins *et al.* (1997) 9

Custos reduzidos 11

Forma emergente ordenada (*Sorted forward star form*) 12

Algoritmo MPS – Martins *et al.* (1997) 13

Estudo comparativo 15

Experiência computacional 15

Tipos de redes 15

Condições das experiências 16

Tempos de execução 16

Conclusões 23

Referências 24

Introdução

Problema

Dados:

- Rede $(\mathcal{N}, \mathcal{A})$;
- Nós inicial e terminal $s, t \in \mathcal{N}$ $(s \neq t)$;
- Custo de $(i,j) \in \mathcal{A}$, c_{ij}; Custo de $p \in \mathcal{P}$, $c(p) = \Sigma_p c_{ij}$;
- $K \in \mathbb{N}$.

Observações:

- Trajecto de $\mathcal{P}$, $p = \langle s = v_1, v_2, \ldots, v_\ell = t \rangle$, onde $v_1, \ldots, v_\ell \in \mathcal{N}$.
- Caminho de $\mathcal{P}$, trajecto de s para t sem nós repetidos.
- Considera-se que nenhum arco se inicia em t.

Pretende-se determinar:

1. p_1, trajecto mais curto de s para t;
2. p_2, 2º trajecto mais curto de s para t;

   $\ldots$

K. p_K, K-ésimo trajecto mais curto de s para t.

Subproblemas

Determinar trajectos que satisfazem a restrições adicionais:

- O Problema dos K caminhos mais curtos;

  $\ldots$.

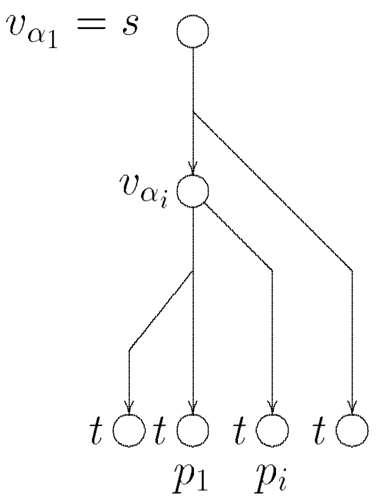
Árvore dos K trajectos mais curtos

Árvore dos K trajectos mais curtos:

> "Pseudo"-árvore que contém os K trajectos mais curtos numa rede entre um par de nós, onde cada trajecto se "desvia" dos "anteriores" num determinado nó.

Árvore dos K caminhos mais curtos:

> Árvore análoga à anterior mas contendo apenas os K caminhos mais curtos numa rede entre um par de nós.



$$v_{\alpha_i} = \begin{cases} s\,, & i = 1 \\ \text{nó de } p_i \text{ tal que } \mathrm{sub}_{p_i}(s, v_{\alpha_i}) \\ \text{é subtrajecto de } p_1, \ldots, p_{i-1} \text{ e}\,, & i > 1 \\ \text{tem número de nós máximo} \end{cases}$$

De entre os algoritmos baseados na construção desta árvore constam:

Para a enumeração de trajectos:

Algoritmos que a partir de p_i determinam "todos" os possíveis "desvios" mais curtos.

- Hoffman e Pavley (1957).

Algoritmos que a partir de p_i determinam apenas "alguns" dos "desvios" mais curtos.

- Martins *et al.* (1997)
  - Elementar.
  - MPS.

Para a enumeração de caminhos:

Algoritmos que determinam apenas caminhos a partir de alterações de $(\mathcal{N}, \mathcal{A})$.

- Yen (1974).

Adaptações dos algoritmos dos desvios para a enumeração de trajectos.

- Martins *et al.* (1997)
  - Hoffman e Pavley.
  - Elementar.
  - MPS.

Algoritmos dos desvios

Observação:

Usam X, um conjunto de trajectos candidatos ao próximo trajecto/caminho mais curto.

Forma geral:

$p_1 \longleftarrow$ trajecto/caminho mais curto de $\mathcal{P}$
$X \longleftarrow \{p_1\}$
Para (i desde 1 até K) **Fazer**
    $p_i \longleftarrow$ elemento de X com menor custo
    $X \longleftarrow X - \{p_i\}$
    Gerar novos trajectos/caminhos de $\mathcal{P}$ a partir de p_i (desvios)
    Juntar os trajectos/caminhos gerados a X

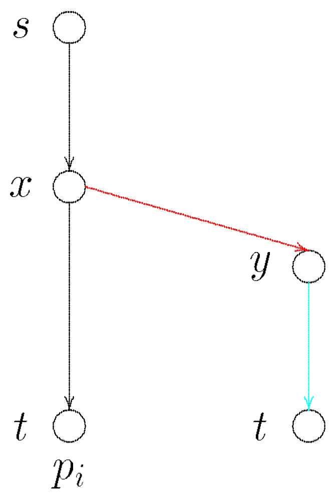
Como gerar novos trajectos/caminhos a partir de p_i?
ou
Que novos trajectos/caminhos gerar a partir de p_i?

Os trajectos/caminhos gerados são da forma

$$\mathrm{sub}_{p_i}(s, x) \diamond (x, y) \diamond p,$$

onde:

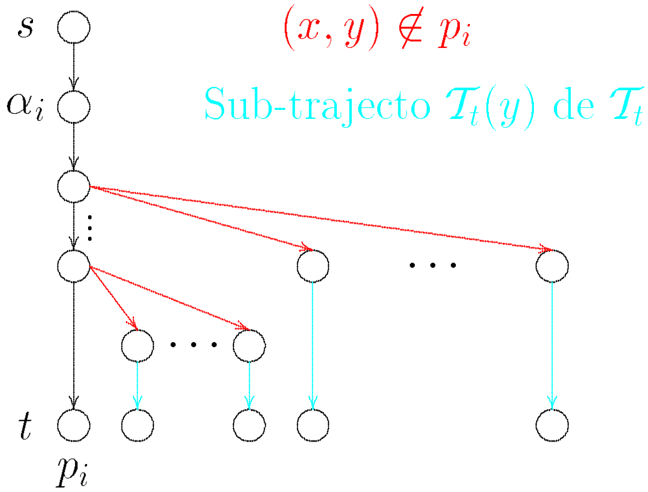
- $\mathrm{sub}_{p_i}(s, x)$, sub-trajecto de p_i de s até x
- $(x, y) \notin p_i$
- p, trajecto de y para t em $(\mathcal{N}, \mathcal{A})$



$(x, y) \notin p_i$
Trajecto de y para t em $(\mathcal{N}, \mathcal{A})$

Algoritmo de Hoffman e Pavley (1957)

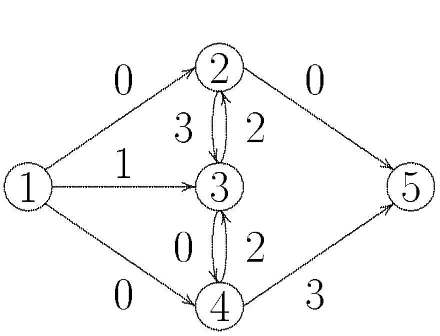
Dado p_i gerar "todos" os desvios mais curtos em cada nó:



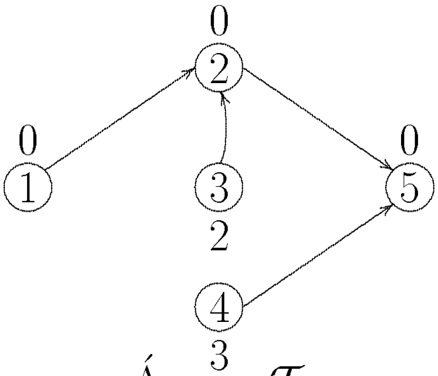
Observações:

- Sendo $\mathcal{T}_t$, árvore dos trajectos mais curtos de $x \in \mathcal{N}$ para t em $(\mathcal{N}, \mathcal{A})$, os trajectos pretendidos são $\mathcal{T}_t(y)$.
- Para não repetir trajectos já gerados analisam-se os nós de p_i seguintes α_i.

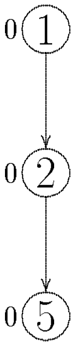
`Para` (x nó de p_i desde o seguinte a α_i até t) `Fazer`
  `Para` ($(x, y) \in \mathcal{A}$ e $(x, y) \notin p_i$) `Fazer`
    $p \longleftarrow (x, y) \diamond \mathcal{T}_t(y)$
    $X \longleftarrow X \cup \{ \text{sub}_{p_i}(s, x) \diamond p \}$



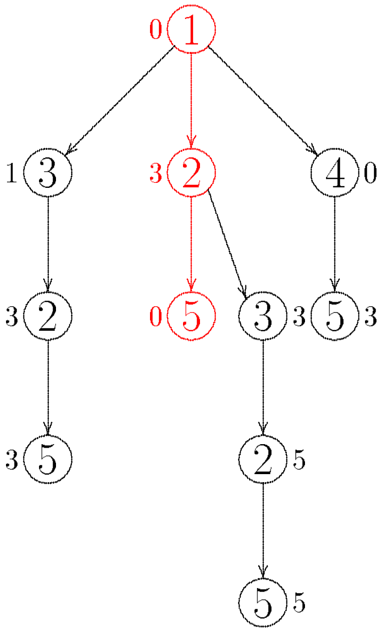
Rede $(\mathcal{N}, \mathcal{A})$



Árvore $\mathcal{T}_t$



$$X = \{\langle 1, 2, 5\rangle\}$$

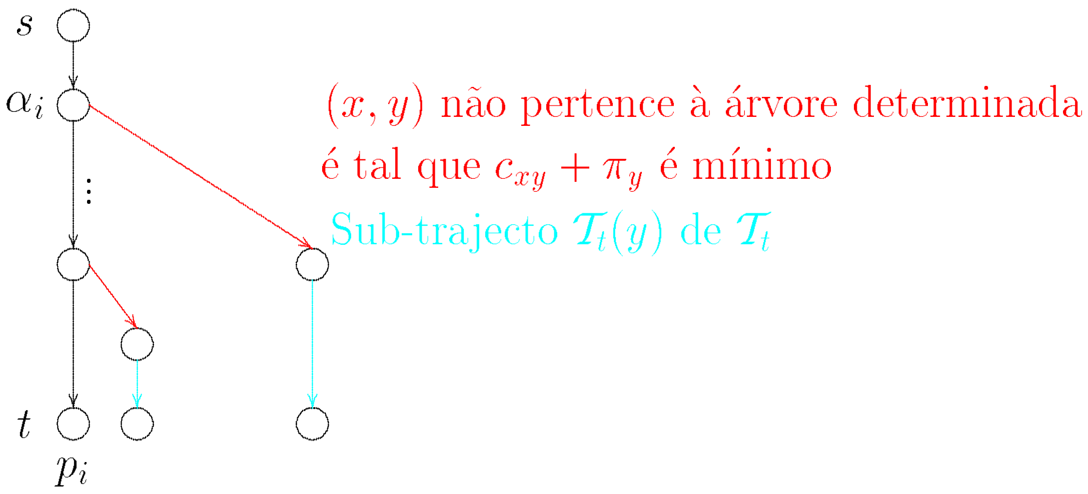


$$p_1 = \langle 1, 2, 5\rangle$$

$$X = \{\langle 1, 3, 2, 5\rangle, \langle 1, 4, 5\rangle, \langle 1, 2, 3, 2, 5\rangle\}$$

Algoritmo de Yen – Enumeração de caminhos (1974)

Dado p_i gerar "o" caminho desvio mais curto em cada nó:



Observações:

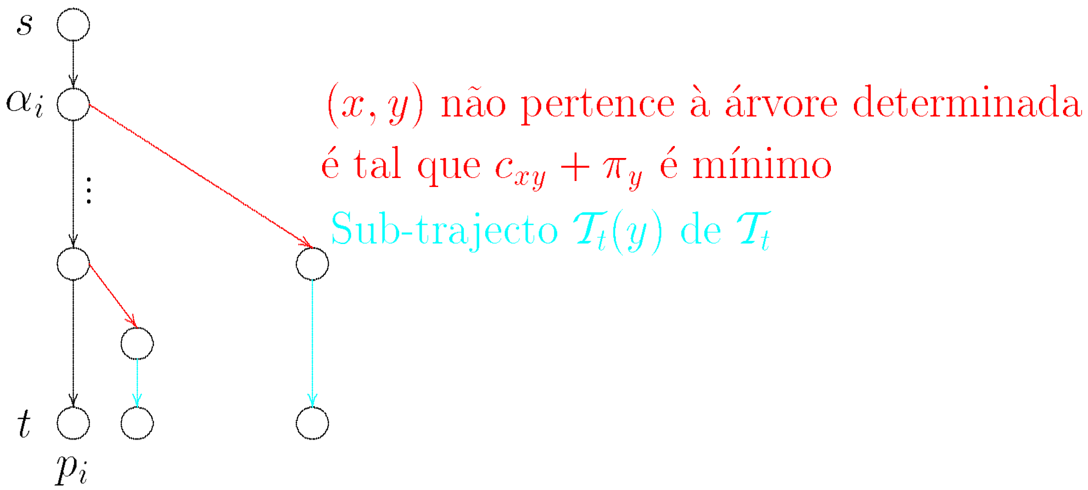
- Para não repetir trajectos já gerados:
  - Analisam-se os nós de p_i a partir de α_i.
  - O arco (x, y) não deve pertencer à árvore já construída.
- Para garantir que o trajecto gerado é um caminho:
  - Removem-se da rede os nós de p_i desde s até x, exclusivé.

Para (x nó de p_i desde α_i até t) **Fazer**

  Remover de $(\mathcal{N}, \mathcal{A})$ os arcos (x, y) na árvore construída
  Remover de $(\mathcal{N}, \mathcal{A})$ nós de p_i de s até x, exclusivé
  $p \longleftarrow$ caminho mais curto de x para t na rede alterada
  $X \longleftarrow X \cup \{ \text{sub}_{p_i}(s, x) \diamond p \}$
  Repor rede original

Algoritmo elementar – Martins *et al.* (1997)

Combinando os dois algoritmos anteriores, dado p_i gerar "o" trajecto desvio mais curto em cada nó:



Observações:

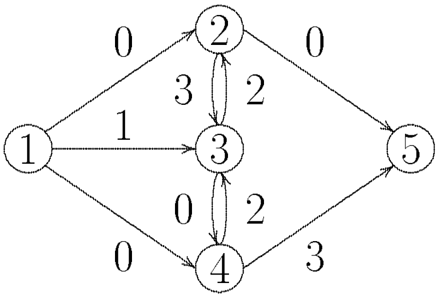
- Para não repetir trajectos já gerados:
  - Analisam-se os nós de p_i a partir de α_i.
  - O arco (x, y) não deve pertencer à árvore já construída.
- Escolher (x, y) tal que $c_{xy} + \pi_y$ é mínimo entre os arcos analisados, onde $\pi_y = c(\mathcal{T}_t(y))$.

Para (x nó de p_i desde α_i até t) **Fazer**

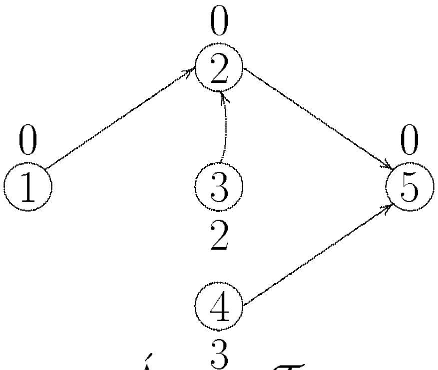
$(x, y) \longleftarrow$ arco tal que $c_{xy} + \pi_y \leq c_{uv} + \pi_v$, com $(x, y), (u, v)$ não pertencentes à árvore construída

$p \longleftarrow (x, y) \diamond \mathcal{T}_t(y)$

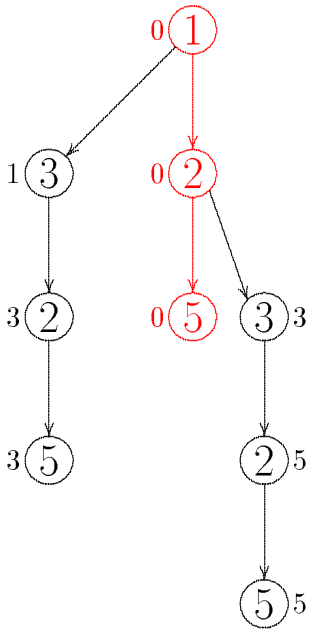
$X \longleftarrow X \cup \{\, \mathrm{sub}_{p_i}(s, x) \diamond p \}$



Rede $(\mathcal{N}, \mathcal{A})$



Árvore $\mathcal{T}_t$



$$p_1 = \langle 1, 2, 5 \rangle$$

$$X = \{\langle 1, 3, 2, 5 \rangle, \langle 1, 2, 3, 2, 5 \rangle\}$$

Custos reduzidos

Custo reduzido associado a $\mathcal{T}_t$ – Eppstein (1994)

$$\bar{c}_{ij} = c_{ij} + \pi_j - \pi_i$$

Propriedades:

1. $c(p) \leq c(q) \Leftrightarrow \bar{c}(p) \leq \bar{c}(q)$
2. $\bar{c}_{ij} = 0, \forall (i,j) \in \mathcal{T}_t \Rightarrow \bar{c}(\mathcal{T}_t(x)) = 0, \forall x \in \mathcal{N}$

No algoritmo elementar:

Basta escolher (x, y) não pertencente à árvore construída com custo reduzido mínimo

`Para` (x nó de p_i desde α_i até t) `Fazer`

$(x, y) \longleftarrow$ arco tal que $\bar{c}_{xy} \leq \bar{c}_{xz}$, com $(x, y), (x, z)$ não pertencentes à árvore construída

$p \longleftarrow (x, y) \diamond \mathcal{T}_t(y)$

$X \longleftarrow X \cup \{\ \mathrm{sub}_{p_i}(s, x) \diamond p\}$

Forma emergente ordenada (*Sorted forward star form*)

$\bar{c}_{ij}$ denota o custo a acrescentar ao do trajecto mais curto de i para t, π_i, quando o obrigamos a passar por (i, j).

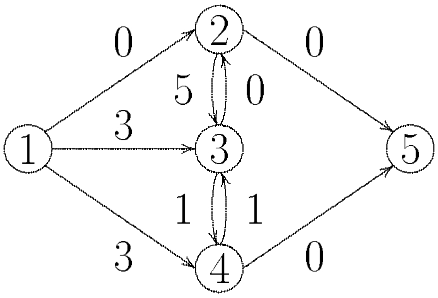
Ordenar $\mathcal{A}$ de modo que, $(i_1, j_1), (i_2, j_2) \in \mathcal{A}$,

$$\left.\begin{array}{l} i_1 < i_2 \\ \\ \quad \text{ou} \\ \\ i_1 = i_2 \ \text{ e } \left\{\begin{array}{l} \bar{c}_{i_1 j_1} < \bar{c}_{i_2 j_2} \\ \qquad \text{ou} \\ \bar{c}_{i_1 j_1} = \bar{c}_{i_2 j_2} \ \text{ e } \ (i_1, j_1) \in \mathcal{T}_t \end{array}\right. \end{array}\right\} \Rightarrow (i_1, j_1) \text{``} < \text{''} (i_2, j_2)$$

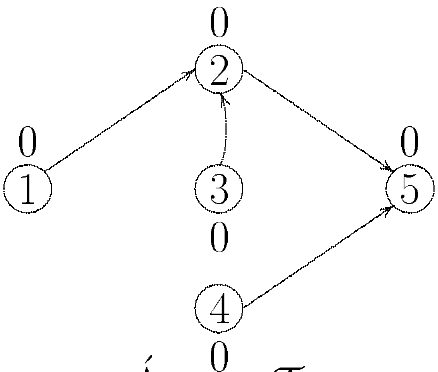
Algoritmo MPS – Martins *et al.* (1997)

$$\left.\begin{array}{c}\text{Algoritmo elementar}\\ +\\ \text{Custos reduzidos}\\ +\\ \text{Forma emergente ordenada}\end{array}\right\} \Longrightarrow \text{Algoritmo MPS}$$

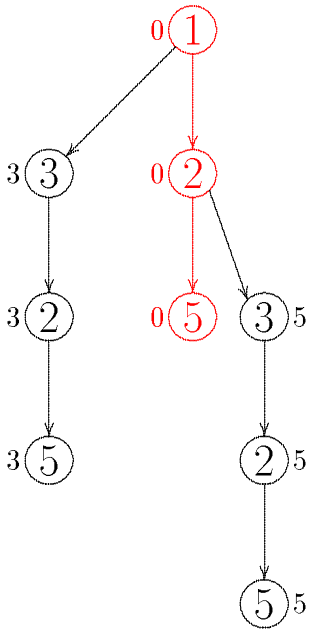
Determinar $\mathcal{T}_t$
$p_1 \longleftarrow \mathcal{T}_t(s)$
Substituir c_{ij} por $\bar{c}_{ij}$
Colocar $\mathcal{A}$ na forma emergente ordenada
$X \longleftarrow \{p_1\}$
`Para` (i desde 1 até K) `Fazer`
    $p_i \longleftarrow$ trajecto com menor custo reduzido em X
    $X \longleftarrow X - \{p_i\}$
    `Para` (x nó de p_i desde α_i até t) `Fazer`
        $(x, y) \longleftarrow$ arco seguinte ao de p_i com início em x
        $p \longleftarrow (x, y) \diamond \mathcal{T}_t(y)$
        $X \longleftarrow X \cup \{\, \mathrm{sub}_{p_i}(s, x) \diamond p\}$



Rede $(\mathcal{N}, \mathcal{A})$



Árvore $\mathcal{T}_t$



$$p_1 = \langle 1, 2, 5 \rangle$$

$$X = \{\langle 1, 3, 2, 5 \rangle, \langle 1, 2, 3, 2, 5 \rangle\}$$

(i,j)	$(1,2)$	$(1,3)$	$(1,4)$	$(2,5)$	$(2,3)$	$(3,2)$	$(3,4)$	$(4,5)$	$(4,3)$
$\bar{c}_{ij}$	0	3	3	0	5	0	1	0	1

$\mathcal{A}$ na forma emergente ordenada

Estudo comparativo

Experiência computacional

Tipos de redes

Redes aleatórias

Nós (n)	5000	10000
Arcos (m)	7500, 10000, 75000, 100000	15000, 20000, 150000, 200000
Densidades (m/n)	1.5, 2, 15, 20	

Redes completas

Nós (n)	100, 250, 500, 1000

Grelhas

Nós (n)	500, 2500, 4500
$(q = w^2/n = n/h^2)$	1, 2, 5, 10

Condições das experiências

Características da máquina utilizada (`rena4`):

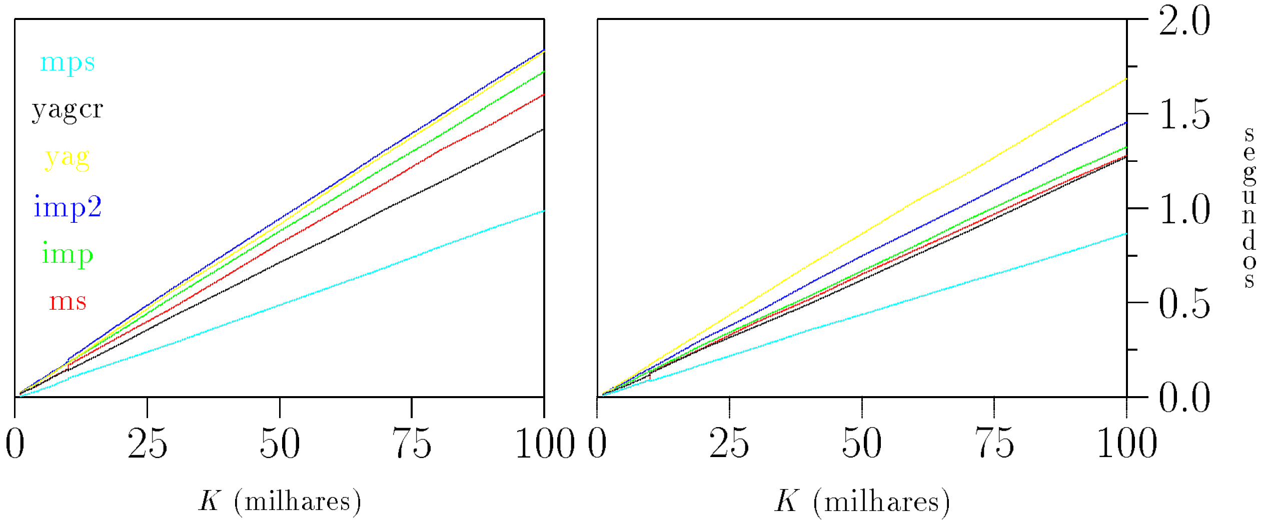
- Pentium II
- 266 MHz
- Sistema Linux

Tempos médios obtidos ao resolver:

- 15 problemas de cada tipo;
- 3 tipos de redes de diferentes dimensões.

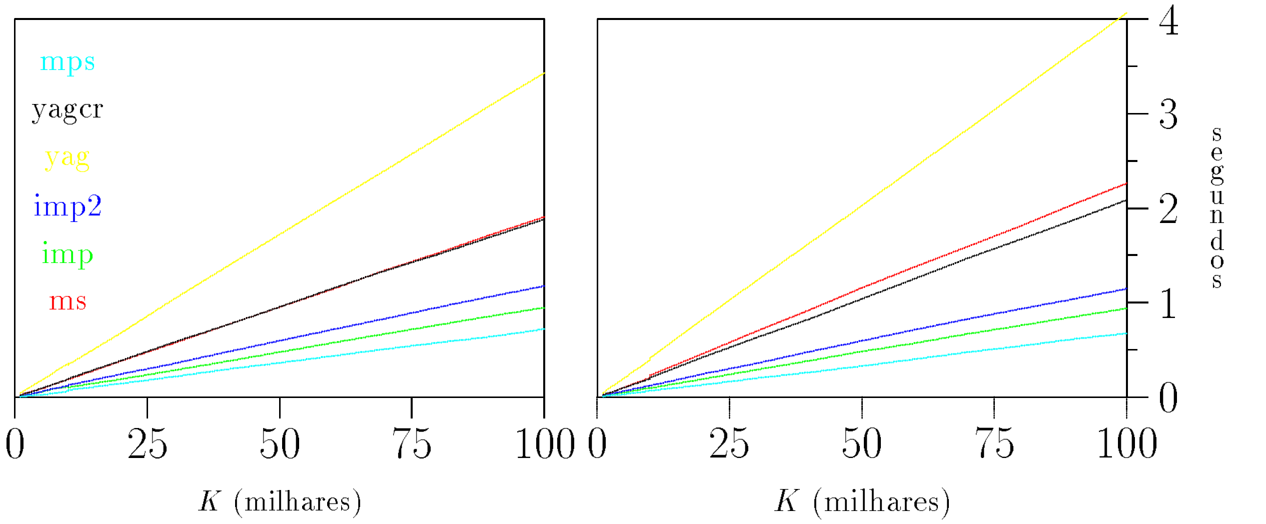
Testadas implementações, em linguagem C, de:

- Algoritmo MS, `ms`.
- Melhoramento do algoritmo MS usando inserção linear, `imp`.
- Melhoramento do algoritmo MS usando *heaps* binários, `imp2`.
- Algoritmo elementar, `yag`.
- Algoritmo elementar com custo reduzidos, `yagcr`.
- Algoritmo MPS, `mps`.



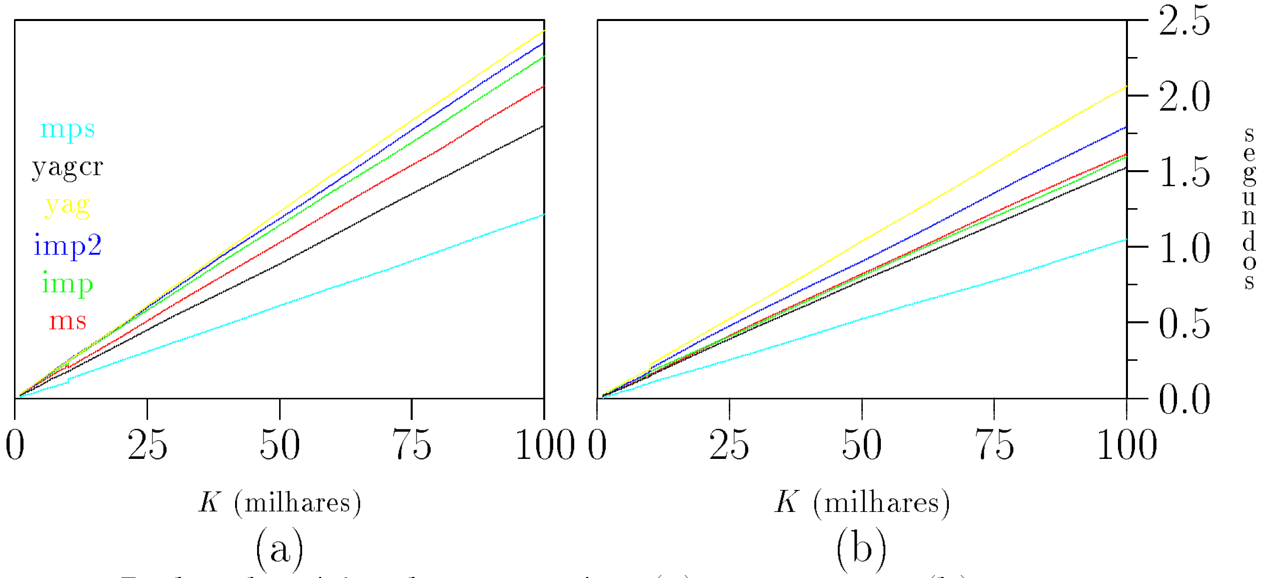
(a) (b)

Redes aleatórias de 5000 nós e (a) 7500 arcos, (b) 10000 arcos

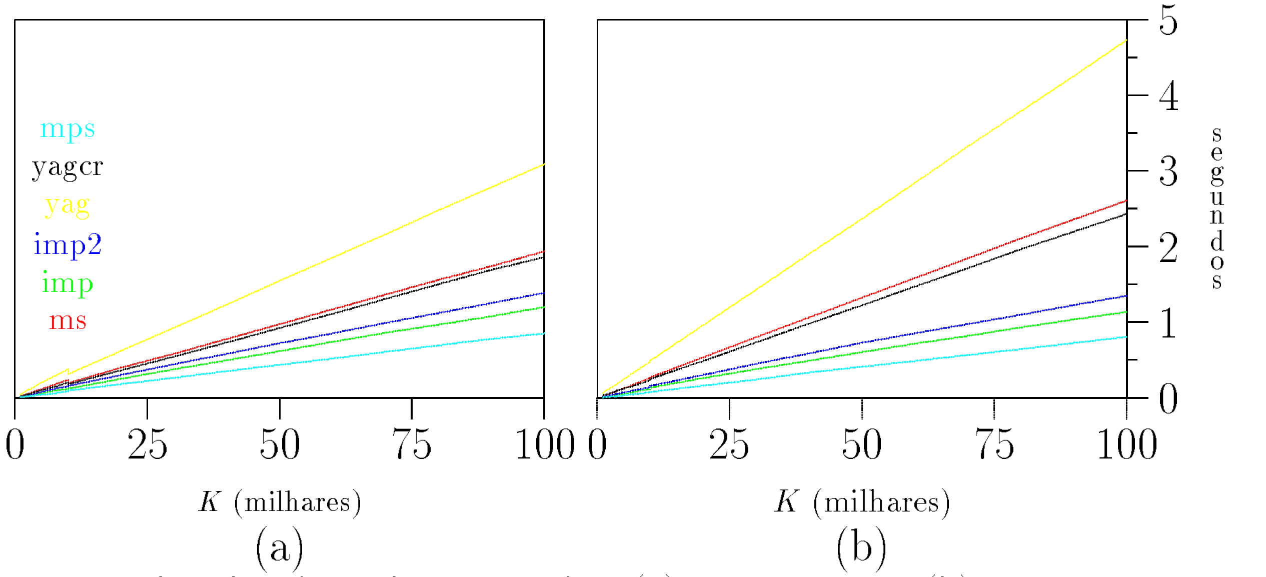


(a) (b)

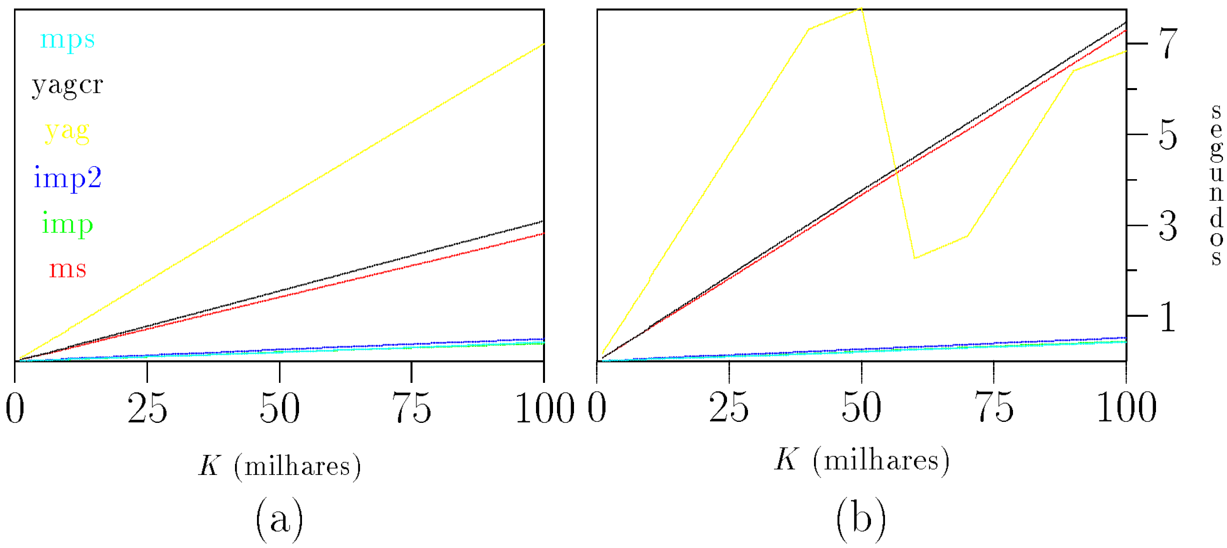
Redes aleatórias de 5000 nós e (a) 75000 arcos, (b) 100000 arcos

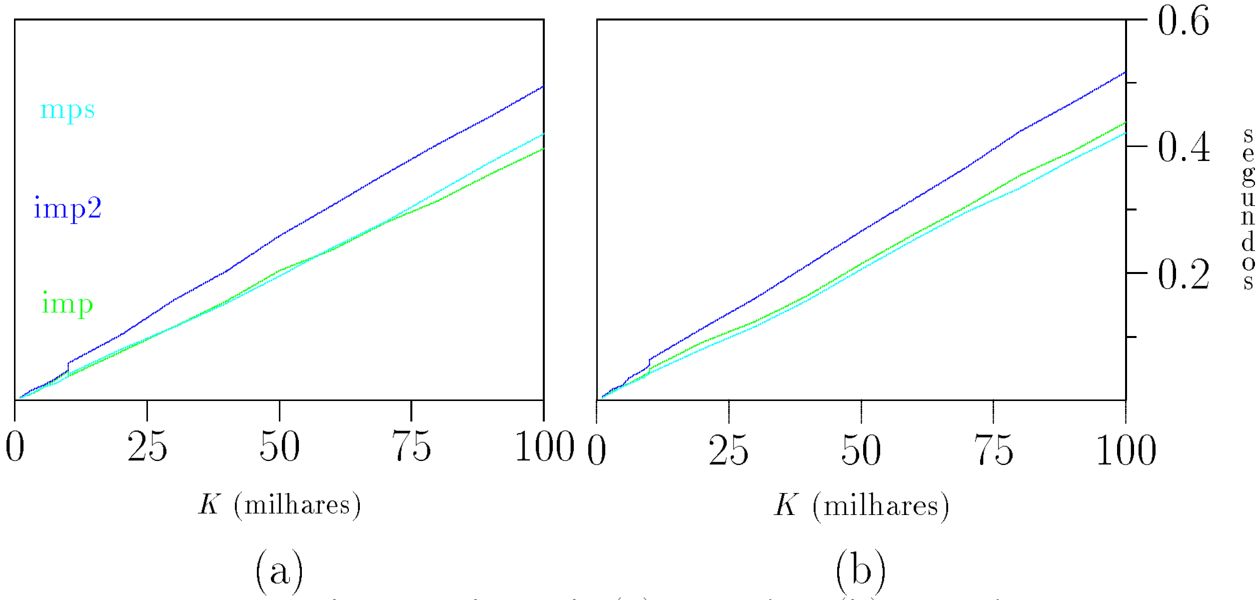


Redes aleatórias de 10000 nós e (a) 15000 arcos, (b) 20000 arcos

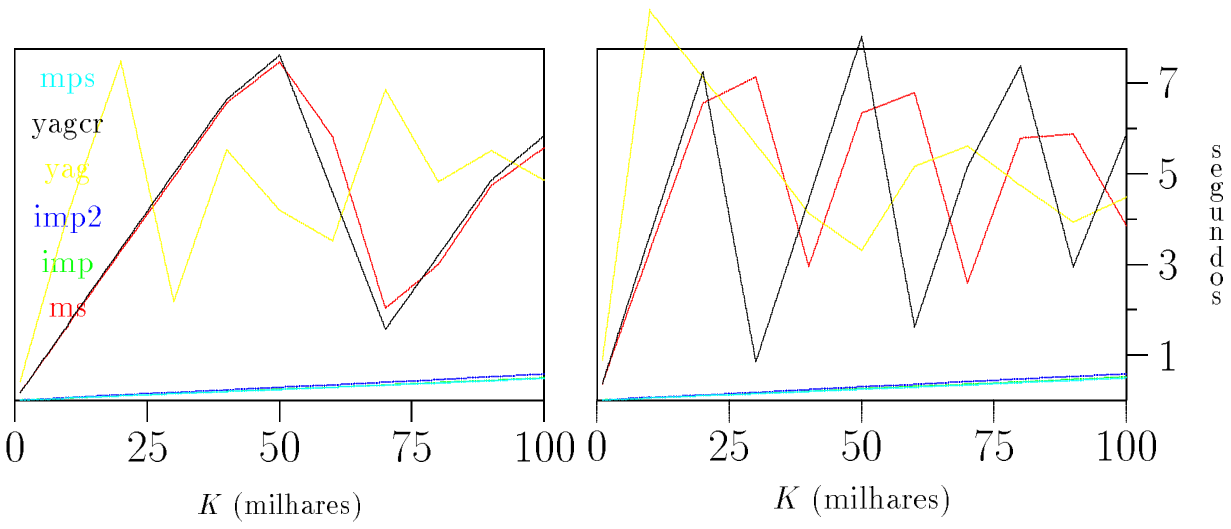


Redes aleatórias de 10000 nós e (a) 150000 arcos, (b) 200000 arcos

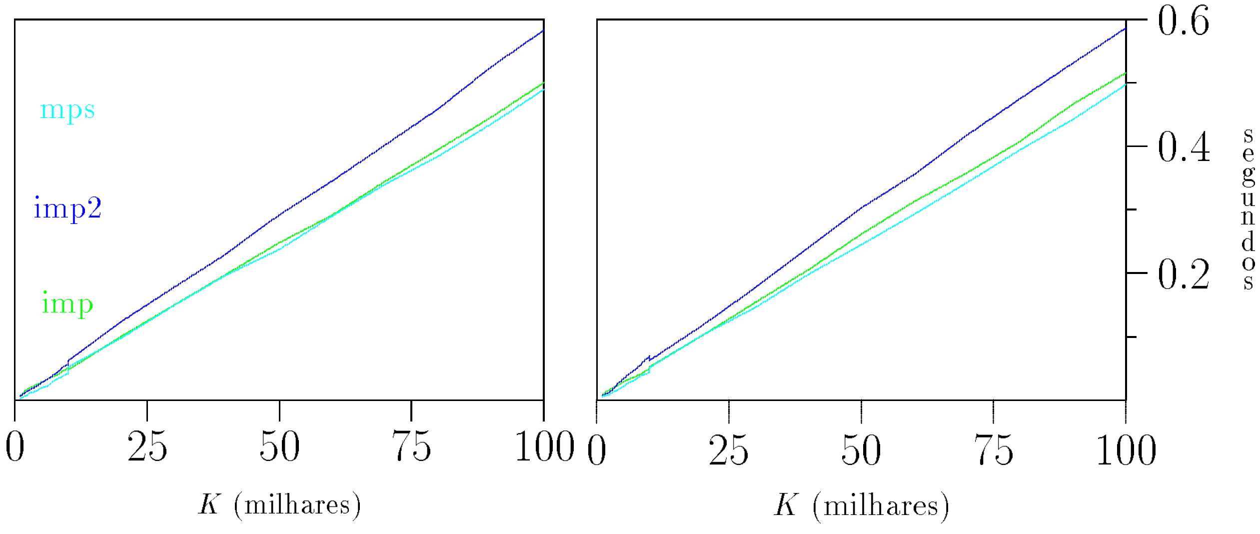




Redes completas de (a) 100 nós e (b) 250 nós

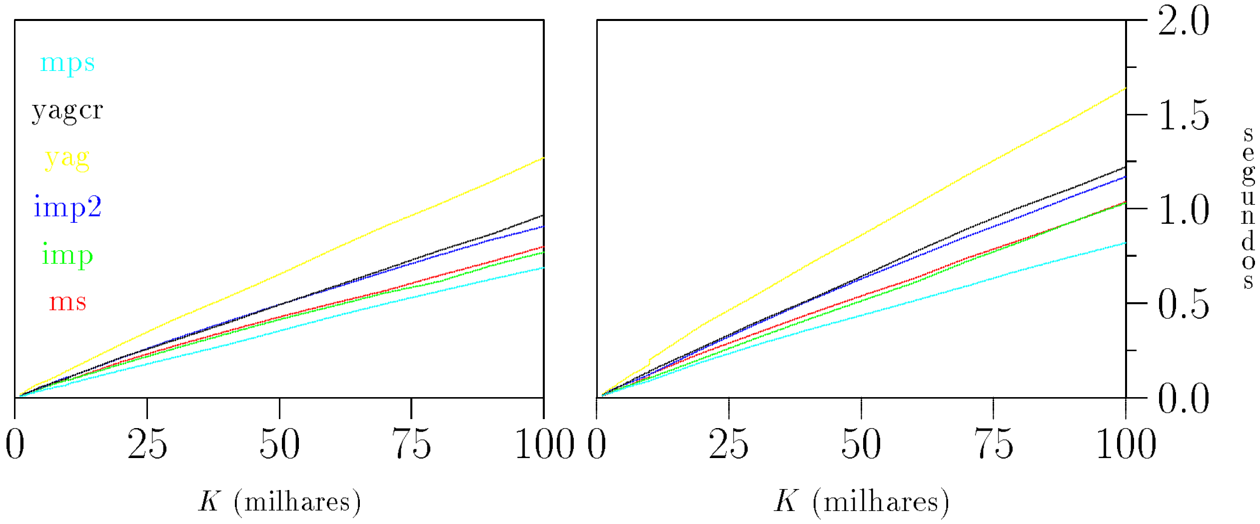


(a) (b)



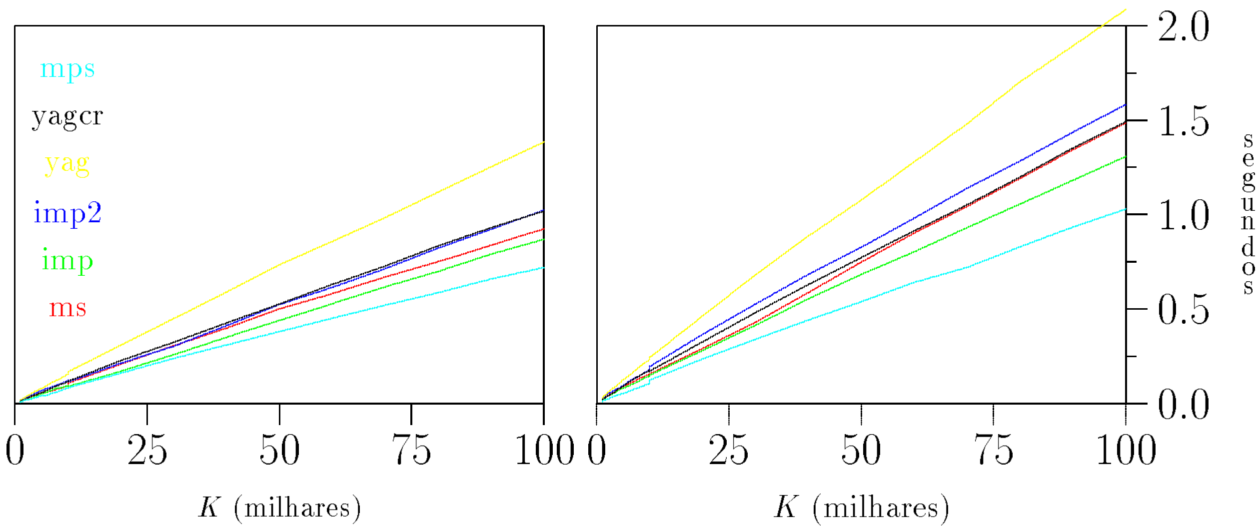
(a) (b)

Redes completas de (a) 500 nós e (b) 1000 nós



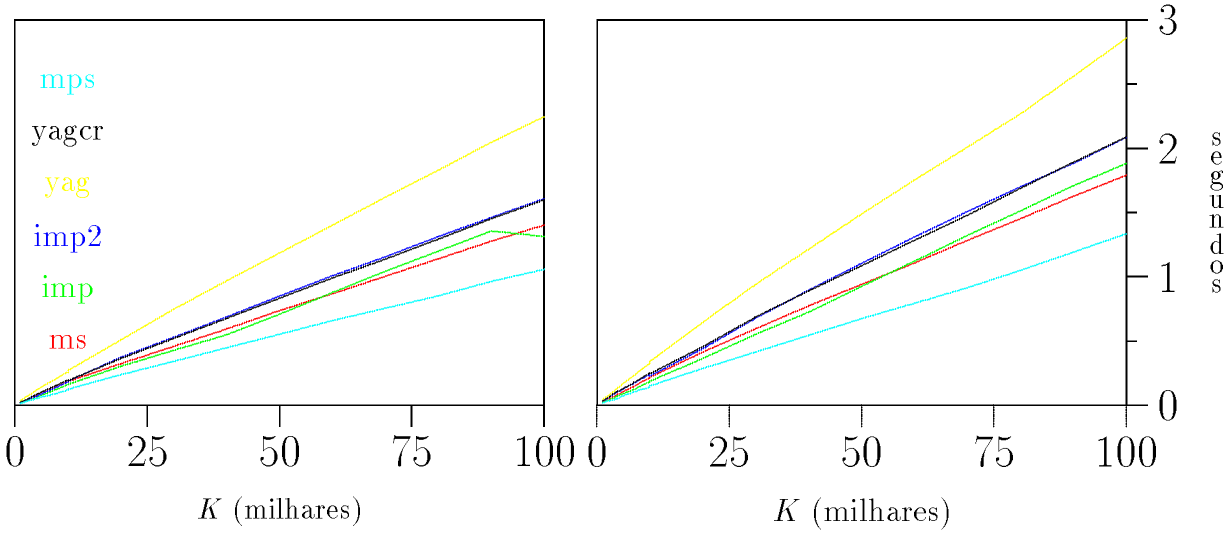
(a) (b)

Grelhas de 500 nós e (a) $q = 1$, (b) $q = 2$



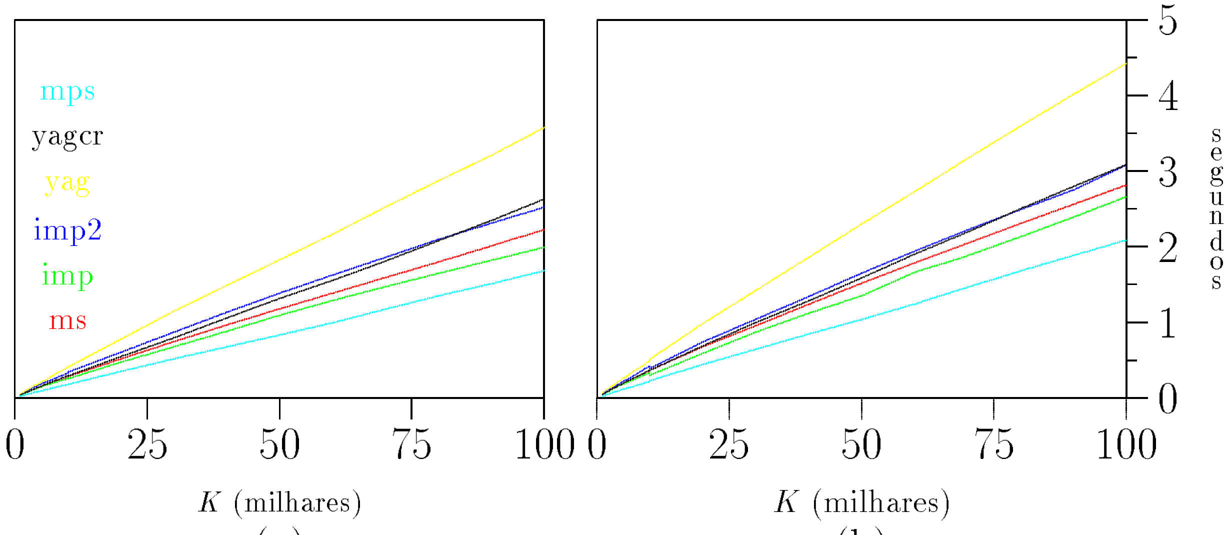
(a) (b)

Grelhas de 500 nós e (a) $q = 5$, (b) $q = 10$



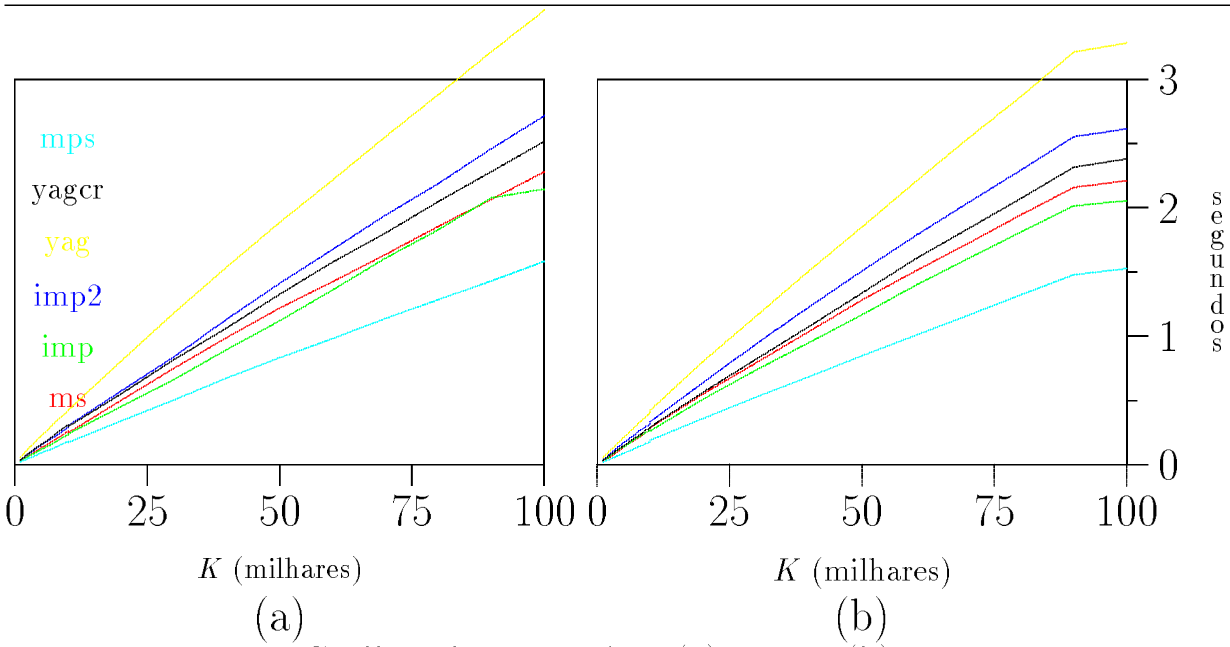
(a) (b)

Grelhas de 2500 nós e (a) $q = 1$, (b) $q = 2$



(a) (b)

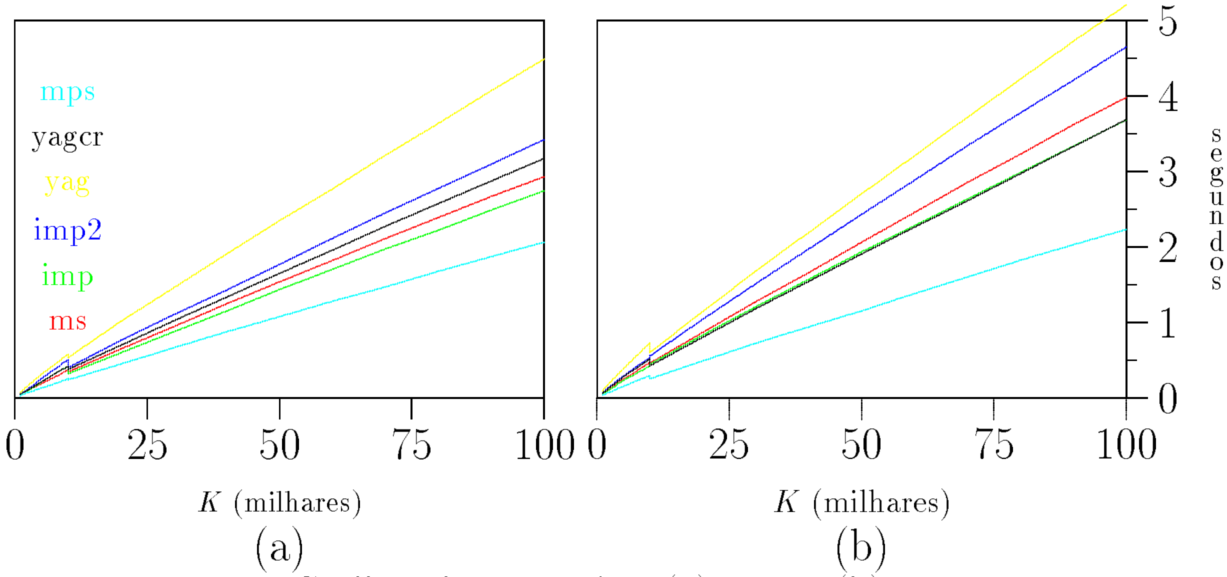
Grelhas de 2500 nós e (a) $q = 5$, (b) $q = 10$



Grelhas de 4500 nós e (a) $q = 1$, (b) $q = 2$

Conclusões

- Os algoritmos testados podem dividir-se em dois grupos:
  - `yag`, `yagcr` e `ms`, cujos tempos de execução aumentam muito rapidamente à medida que aumentam d e n;
  - `mps`, `imp` e `imp2`, cujos tempos de execução aumentam à medida que aumenta d, e que são "quase" constantes quando n dobra.
- De entre os algoritmos do segundo grupo:
  - o algoritmo dos desvios, `mps`, apresenta, em geral, melhores tempos de execução;
  - exceptuando o caso das grelhas, em que os algoritmos dos apagamentos, `imp` e `imp2` apresentam melhores resultados.



Grelhas de 4500 nós e (a) $q = 5$, (b) $q = 10$

Referências

- R. Hoffman and R. R. Pavley. A method for the solution of the n th best path problem. *Journal of the Association for Computing Machinery*, 6:506–514, 1959.

- J. Y. Yen. Finding the K shortest loopless paths in a network. *Management Science*, 17:712–716, 1971.

- E. Q. V. Martins, M. M. B. Pascoal and J. L. E. Santos. An algorithm for ranking loopless paths. Technical Report 99/007, CISUC, Dezembro 1999, (1^a versão – A new algorithm for ranking loopless paths).

  (http://www.mat.uc.pt/~eqvm/cientificos/investigacao/Artigos/MPS_novo.ps.gz).

- E. Q. V. Martins, M. M. B. Pascoal e J. L. E. Santos. Deviation algorithms for ranking shortest paths. *The International Journal of Foundations of Computer Science*, 10:(3):247–263, 1999.

  (http://www.mat.uc.pt/~eqvm/cientificos/investigacao/Artigos/deviation.ps.gz).