

Hugo Braga

Programa de Pós-Graduação em Mecatrônica
Universidade Federal da Bahia
Defesa de Mestrado
Orientador: Dr. Flávio Assis

22 de Outubro de 2012

Estrutura da apresentação I

- 1 Motivação
- 2 Por quê o problema é novo?
- 3 Formalização do problema
- 4 Algoritmos
 - APPROX
 - SIM
- 5 Experimentos
- 6 Função Submodular e Matroids
- 7 Conclusão

A set of small navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

Definição de DSMDStP

- Entrada:
 - Um grafo direcionado $G = (V, E)$, onde o peso de cada aresta é definido por uma função $C : E \rightarrow \mathbb{R}^+$.
 - Um nó de origem s .
 - Um fator de dilatação k ($k \in \mathbb{R}^+, k \geq 1$).
 - Um conjunto de terminais $T \subseteq V \setminus \{s\}$.
- Saída: Encontrar uma arborescência $A = (V_A, E_A)$, com nó origem em s e que cobre T , tal que:
 - $dist(s, t, A) \leq k \cdot dist(s, t, G), \forall t \in T$;
 - o grau máximo de saída dos nós em A , denotado por $D_{max}(A) = \max_{v \in V_A} \{odeg(v, A)\}$, é minimizado.

Algoritmo de aproximação: preliminares

- Nosso algoritmo é baseado no algoritmo apresentado em [Elkin and Kortsarz 2006]. Este foi adaptado para considerar a propriedade de spanner.
- Possui duas fases:
 - fase 1: cômputo da \sqrt{l} -partition.
 - fase 2: utilização do problema do *Multiple Set-Cover* (MSC) para cobrir os demais terminais.
- $V = C \dot{\cup} U$. Seja $UT = U \cap T$ e $CT = C \cap T$.
- Seja $l = |T|$ e d^* o grau máximo de uma solução ótima para uma instância de DSMDStP ($d^* \leq l$).
- Seja $G(S)$ o grafo induzido por um conjunto de nós S .
- Para um nó $u \in U$, $\Delta\text{-neigh}(u) = \{t : t \in UT \wedge \text{dist}(s, u, G) + \text{dist}(u, t, G(U)) \leq k \cdot \text{dist}(s, t, G)\}$.

Algoritmo de aproximação: preliminares

- Nosso algoritmo é baseado no algoritmo apresentado em [Elkin and Kortsarz 2006]. Este foi adaptado para considerar a propriedade de spanner.
- Possui duas fases:
 - fase 1: cômputo da \sqrt{l} -partition.
 - fase 2: utilização do problema do *Multiple Set-Cover* (MSC) para cobrir os demais terminais.
- $V = C \dot{\cup} U$. Seja $UT = U \cap T$ e $CT = C \cap T$.
- Seja $l = |T|$ e d^* o grau máximo de uma solução ótima para uma instância de DSMDStP ($d^* \leq l$).
- Seja $G(S)$ o grafo induzido por um conjunto de nós S .
- Para um nó $u \in U$, $\Delta\text{-neigh}(u) = \{t : t \in UT \wedge \text{dist}(s, u, G) + \text{dist}(u, t, G(U)) \leq k \cdot \text{dist}(s, t, G)\}$.

Algoritmo de aproximação: preliminares

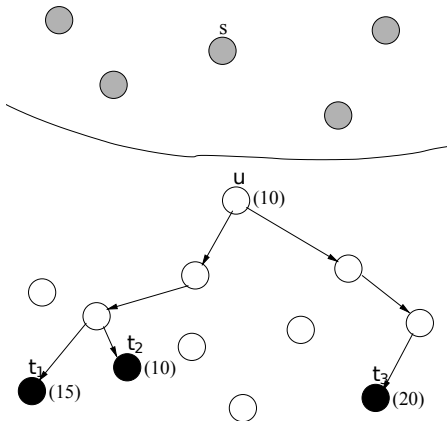
- Nosso algoritmo é baseado no algoritmo apresentado em [Elkin and Kortsarz 2006]. Este foi adaptado para considerar a propriedade de spanner.
- Possui duas fases:
 - fase 1: cômputo da \sqrt{l} -partition.
 - fase 2: utilização do problema do *Multiple Set-Cover* (MSC) para cobrir os demais terminais.
- $V = C \dot{\cup} U$. Seja $UT = U \cap T$ e $CT = C \cap T$.
- Seja $l = |T|$ e d^* o grau máximo de uma solução ótima para uma instância de DSMDStP ($d^* \leq l$).
- Seja $G(S)$ o grafo induzido por um conjunto de nós S .
- Para um nó $u \in U$, $\Delta\text{-neigh}(u) = \{t : t \in UT \wedge \text{dist}(s, u, G) + \text{dist}(u, t, G(U)) \leq k \cdot \text{dist}(s, t, G)\}$.

Algoritmo de aproximação: preliminares

- Nosso algoritmo é baseado no algoritmo apresentado em [Elkin and Kortsarz 2006]. Este foi adaptado para considerar a propriedade de spanner.
- Possui duas fases:
 - fase 1: cômputo da \sqrt{l} -partition.
 - fase 2: utilização do problema do *Multiple Set-Cover* (MSC) para cobrir os demais terminais.
- $V = C \dot{\cup} U$. Seja $UT = U \cap T$ e $CT = C \cap T$.
- Seja $l = |T|$ e d^* o grau máximo de uma solução ótima para uma instância de DSMDStP ($d^* \leq l$).
- Seja $G(S)$ o grafo induzido por um conjunto de nós S .
- Para um nó $u \in U$, $\Delta\text{-neigh}(u) = \{t : t \in UT \wedge \text{dist}(s, u, G) + \text{dist}(u, t, G(U)) \leq k \cdot \text{dist}(s, t, G)\}$.

Δ -neighbourhood

- $t = 2$



$$d(u, t_1, G(U)) = 15$$

$$d(u, t_2, G(U)) = 12 \Rightarrow$$

$$d(u, t_3, G(U)) = 25$$

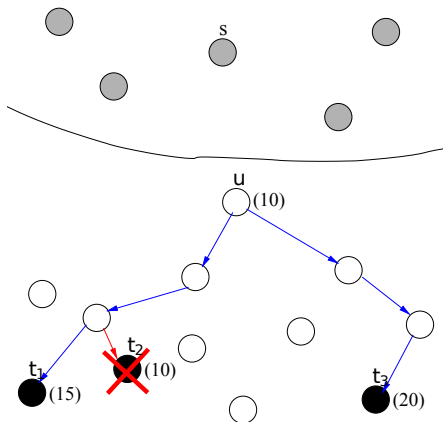
$$d(s, u, G) + d(u, t_1, G(U)) = 10 + 15 \leq 2 \cdot 15$$

$$d(s, u, G) + d(u, t_1, G(U)) = 10 + 12 > 2 \cdot 10$$

$$d(s, u, G) + d(u, t_1, G(U)) = 10 + 25 \leq 2 \cdot 20$$

Δ -neighbourhood

- $t = 2$



$$d(u, t_1, G(U)) = 15$$

$$d(u, t_2, G(U)) = 12 \Rightarrow$$

$$d(u, t_3, G(U)) = 25$$

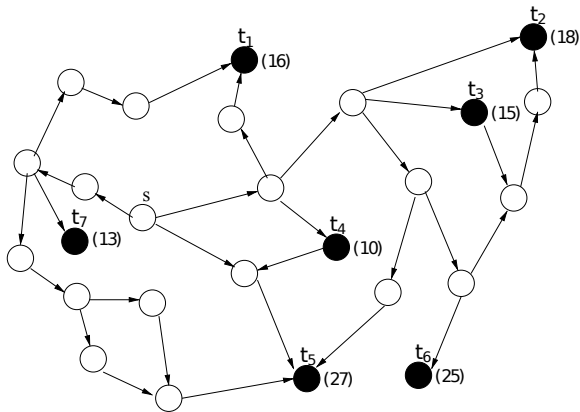
$$d(s, u, G) + d(u, t_1, G(U)) = 10 + 15 \leq 2 \cdot 15$$

$$d(s, u, G) + d(u, t_1, G(U)) = 10 + 12 > 2 \cdot 10 \Rightarrow \Delta - \text{neigh}(u) = \{t_1, t_3\}$$

$$d(s, u, G) + d(u, t_1, G(U)) = 10 + 25 \leq 2 \cdot 20$$

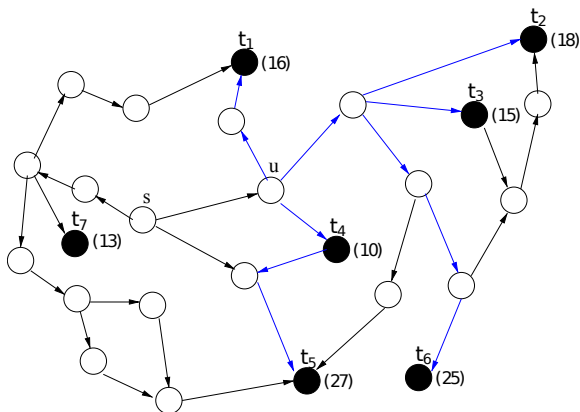
CompPar - Grafo de entrada

• $t = 2$



CompPar - \sqrt{k} -bad node

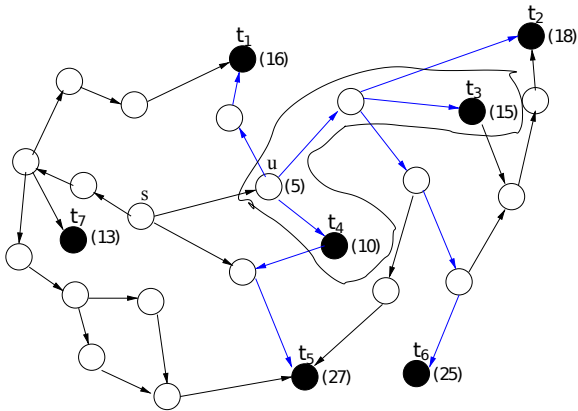
- $t = 2$



- u é \sqrt{k} -bad node.
- $d(s, u, G) = 5$; $d(u, t_4, G(u)) = 5$, $d(u, t_3, G(u)) = 10$, $d(u, t_1, G(u)) = 11$,
 $d(u, t_2, G(u)) = 13$, $d(u, t_6, G(u)) = 20$, $d(u, t_5, G(u)) = 22$.

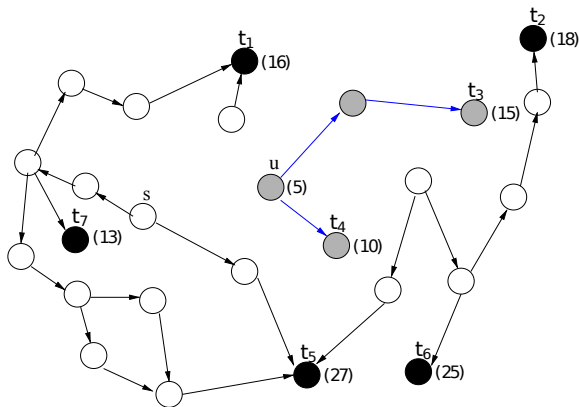
CompPar - $\lfloor \sqrt{I} \rfloor$ terminais mais próximos em UT

- $t = 2$



CompPar - Eliminando os nós cobertos

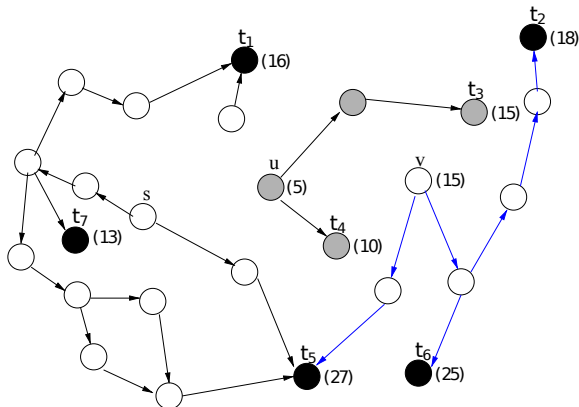
• $t = 2$



• $Root = \{u\}$.

CompPar - \sqrt{k} -bad node

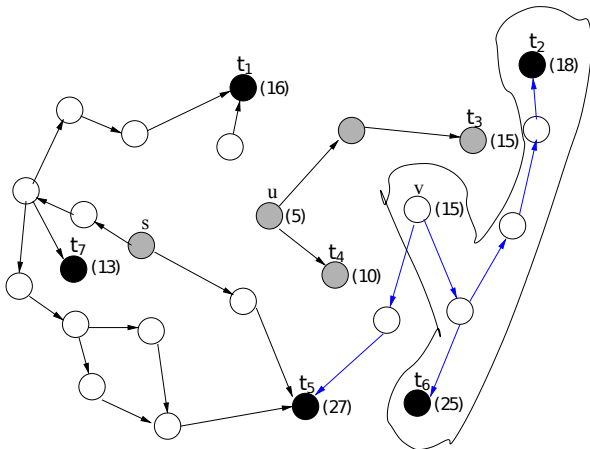
- $t = 2$



- v é \sqrt{k} -bad node.
- $d(v, t_6, G(u)) = 10$, $d(v, t_2, G(u)) = 20$, $d(v, t_5, G(u)) = 24$.

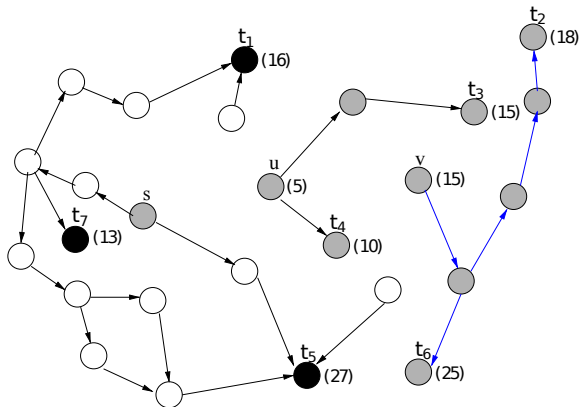
CompPar - \sqrt{I} terminais mais próximos em UT

- $t = 2$



CompPar - Eliminando os nós cobertos

- $t = 2$



- $Root = \{u, v\}$.

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ≡ ↺ 🔍 ↻

Construção de caminhos para $t \in CT$

- Algoritmo 2 descreve o procedimento para construção de um grafo que contém caminhos de s para $t \in CT$ e que satisfaz $k \cdot dist(s, t, G)$.

Algorithm 2: CompGraphFirstPh(G, s, Roots, H)

$$\begin{array}{l} 1 \ A_{Roots} \leftarrow SPT(s, Roots, G) \\ 2 \ G_{\sqrt{I}-Par} \leftarrow A_{Roots} \cup H(V_H, E_H) \\ 3 \ C \leftarrow V(G_{\sqrt{I}-Par}), U \leftarrow V \setminus C \\ 4 \ \text{Output } (C, U, G_{\sqrt{I}-Par}) \end{array}$$

- Terminais em $G_{\sqrt{I}-Par}$ respeitam a restrição de spanner.

Problema do *Multiple Set-Cover* (MSC)

- Seja $\beta(V_1, V_2, E)$ um grafo bipartite. Um conjunto $S \subseteq V_1$ é denominado um *set-cover* de V_2 se $N(S, \beta) = V_2$.
- Definição formal do MSC:
 - Entrada: Um grafo bipartite $\beta(V_1, V_2, E)$ com $|V_1| + |V_2| = n$.
 $V_1 = \dot{\bigcup}_{j=1}^d A_j$.
 - Saída: Um *set-cover* $S \subseteq V_1$ de V_2 que minimiza $val(S)$, onde
 $val(S) = \max\{|S \cap A_i|\}_{i=1}^d$.

Solução de aproximação para o MSC

Teorema 2

[Chekuri and Kumar 2004]. Existe um algoritmo guloso com fator de aproximação de $(\log |V_2| + 1)$ para o problema do MSC.

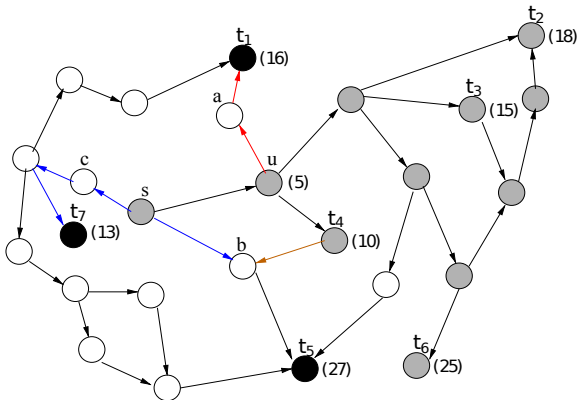
Cobrimos nós em UT através do MSC (Fase 2)

- Utilização de uma instância do MSC para construir caminhos para os nós em UT ao passo que o grau é limitado (similar a [Elkin and Kortsarz 2006]).
- Fase dividida em duas partes: utilização do MSC para encontrar um subconjunto de nós cobertos; em sequência caminhos para os terminais em UT a partir destes nós são definidos.
- Definição da instância $\beta = (V_1, V_2, \varepsilon)$ do MSC:
 - V_1 é formado por *pseudo nós*.
 $V_1 = \{x_{u,v} : u \in C, v \in U, (u, v) \in E\}.$
 - $V_2 = UT.$
 - Existe uma aresta em ε entre $x_{u,v}$ e $t \in V_2$ sse
 $dist(s, u, G) + C(u, v) + dist(v, t, G(U)) \leq k \cdot dist(s, t, G).$
 - Seja $A_u = \{x_{u,v} : v \in N(u, G(U))\}.$ $V_1 = \bigcup_{u \in C} A_u.$

Instância do MSC

- $t = 2$
- Caminhos coloridos satisfazem a relação

$$\text{dist}(s, u, G) + C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G).$$



- $V_1 = \{X_{u,a}, X_{t_4,b}, X_{s,b}, X_{s,c}\}$, partição de $V_1 = \{\{X_{u,a}\}, \{X_{t_4,b}\}, \{X_{s,b}, X_{s,c}\}\}$.
- $V_2 = \{t_1, t_5, t_7\}$.
- $\varepsilon = \{(X_{u,a}, t_1), (X_{t_4,b}, t_5), (X_{s,b}, t_5), (X_{s,c}, t_7)\}$.

partição de V_1

Lema 5

Seja D uma solução para $\beta = (V_1, V_2, \varepsilon)$ com $V_1 = \bigcup_{v \in C} A_v$ usando o algoritmo apresentado em [Chekuri and Kumar 2004]. Então $\max_{v \in C} |D \cap A_v| = O(\log I) \cdot d^*$.

▶ Prova

Algoritmo de Aproximação

Algorithm 3: Algoritmo de Aproximação para o DSMDStP

- 1 Input: $G = (V, E)$, $s \in V$, $T \subset V$, k
 Output: $\mathcal{A}_f = (V_{\mathcal{A}_f}, E_{\mathcal{A}_f})$
- 2 $(C, U, \text{Roots}, H) \leftarrow \text{CompPar}(G, s, k)$
- 3 $(C, U, G_{\sqrt{I-Par}}) \leftarrow \text{CompGraphFirstPh}(G, s, \text{Roots}, H)$
- 4 Build the MSC instance $\beta = (V_1, V_2, \varepsilon)$
- 5 $D \leftarrow$ Apply approximation algorithm [Chekuri and Kumar 2004] on β
- 6 $\Gamma(V_\Gamma, E_\Gamma)$, $V_\Gamma \leftarrow \emptyset$, $E_\Gamma \leftarrow \emptyset$
- 7 **foreach** $t \in V_2$ **do**
- 8 Choose $x_{u,v} \in D : (x_{u,v}, t) \in \varepsilon$ and
 $C(u, v) + \text{dist}(v, t, G(U))$ is minimum
- 9 $s \leftarrow \text{sp}(v, t, G(U))$
- 10 $V_\Gamma \leftarrow V_\Gamma \cup \{u\} \cup V(s)$
- 11 $E_\Gamma \leftarrow E_\Gamma \cup \{(u, v)\} \cup E(s)$
- 12 $G_f \leftarrow G_{\sqrt{I-Par}} \cup \Gamma(V_\Gamma, E_\Gamma)$
- 13 $\mathcal{A}_f \leftarrow \text{SPT}(s, T, G_f)$

Grau máximo de G_f

Lema 6

O grau máximo dos nós em G_f é $\leq 2\sqrt{l} + 2 + O(\log l) \cdot d^$.*

► Prova

Teorema Principal

Teorema 3

Algoritmo 3 gera uma arborescência \mathcal{A}_f com grau máximo $2\sqrt{k} + 2 + O(\log l) \cdot d^$ e que possui caminhos entre s e cada terminal $t \in T$ com custo $\leq k \cdot (\text{dist}(s, t, G) + \text{dist}(s, t_{\max}, G))$.*

► Prova

Complexidade

- $O((\log |T|)(|V|^3|T|^2))$.

► **Detalhe**

SIM

Algorithm 4: SIM - Sliced and Iterative MSC

```

1 Input:  $G = (V, E)$ ,  $s \in V$ ,  $T \subset V$ ,  $k$ 
   Output:  $\mathcal{A}_f = (V_{\mathcal{A}_f}, E_{\mathcal{A}_f})$ 

2  $C \leftarrow \{s\}$ ,  $U \leftarrow V \setminus \{s\}$ ,  $Marked \leftarrow \emptyset$ ,  $I \leftarrow |T|$ 
3 while  $UT \neq \emptyset$  do
4   Set  $V_1, V_2$  and  $\varepsilon$  using the current values of  $C, U, Marked$  and
      $Next_{ter}^{\sqrt{I}}$ 
5   foreach  $t : t \in V_2$  and  $\nexists$  edge  $(x, t) \in \varepsilon$  for any  $x$  do
6     Let  $x_{u,v}$  be a pseudo node such that:
        $u \in Marked$ ,
        $u$  has the smallest out-degree and
        $C(u, v) + dist(v, t, G(U)) \leq k \cdot dist(s, t, G)$ 
7      $V_1 \leftarrow V_1 \cup \{x_{u,v}\}$ ,  $\varepsilon \leftarrow \varepsilon \cup \{(x_{u,v}, t)\}$ 
8    $D \leftarrow$  Apply approximation algorithm [Chekuri and Kumar 2004]
     on  $\beta(V_1, V_2, \varepsilon)$ 
9    $\Gamma(V_\Gamma, E_\Gamma) \leftarrow \emptyset$ 
10  foreach  $t \in V_2$  do
11    Choose a node  $x_{u,v} \in D : (x_{u,v}, t) \in \varepsilon$  and
       $C(u, v) + dist(v, t, G(U))$  is minimum
12     $Marked \leftarrow Marked \cup \{u\}$  {for which  $x_{(u,v)}$ 
      was chosen in the last line}
13     $s \leftarrow sp(v, t, G(U))$ 
14     $V_\Gamma \leftarrow V_\Gamma \cup \{u\} \cup V(s)$ 
15     $E_\Gamma \leftarrow E_\Gamma \cup \{(u, v)\} \cup E(s)$ 
16   $C \leftarrow C \cup V_\Gamma$ ,  $U \leftarrow U \setminus V_\Gamma$ 
17   $\mathcal{A}_f \leftarrow \mathcal{A}_f \cup \Gamma(V_\Gamma, E_\Gamma)$ 

```

Existência de um caminho para todo terminal em \mathcal{A}_f

Lema 8

Seja \mathcal{A}_f a arborescência gerada por SIM. $\forall t \in T$, existe um caminho entre s e t em \mathcal{A}_f .

▶ Prova

Número de iterações do SIM

Lema 9

O número de iterações do SIM é limitado por $\lfloor \sqrt{l} \rfloor + 2$.

▶ Prova

Grafo gerado por SIM é uma arborescência

Lema 10

Seja A_f o grafo gerado por SIM. A_f é uma arborescência.

▶ Prova

Custo máximo dos caminhos gerados por SIM

Teorema 4

Seja \mathcal{A}_f a arborescência gerada por SIM.

$$\forall t \in T, \text{dist}(s, t, \mathcal{A}_f) \leq (\lfloor \sqrt{I} \rfloor + 2) \cdot k \cdot \text{dist}(s, t, G).$$

▶ Prova

Complexidade

- $O((\log \sqrt{|T|})(|V|^3|T|\sqrt{|T|}))$.

► **Detalhe**

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ≡ ↺ 🔍 ↻

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ≡ ↺ 🔍 ↻

A set of small navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

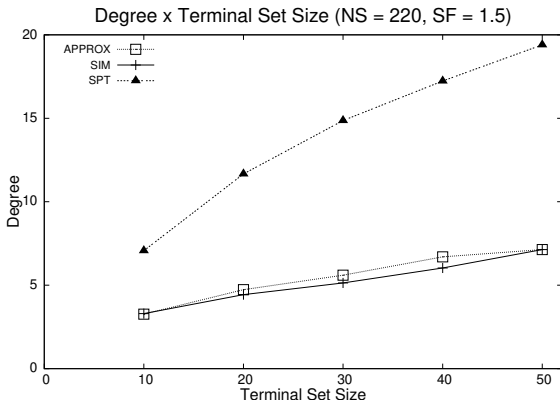
Algoritmos e Métricas

- Algoritmos:
 - DSMDStP é um problema novo.
 - Comparamos com o SPT.
 - Satisfaz as restrições do problema.
 - Algoritmo simples e eficiente*.
 - Similaridades com os algoritmos propostos (principalmente APPROX).
- 4 métricas:
 - Grau máximo.
 - Razão de violação do custo (CVR): $\frac{\sum_{\forall t \in T_{vio}} \frac{dist(s, t, A_f)}{k \cdot dist(s, t, G)}}{|T_{vio}|}$.
 - Pior caso da razão de violação (MAX_CVR).
 - Percentagem de terminais que violam restrições (PVT): $\frac{|T_{vio}|}{|T|} \cdot 100\%$.
 - Percentagem de cenários violados (PVR).

Algoritmos e Métricas

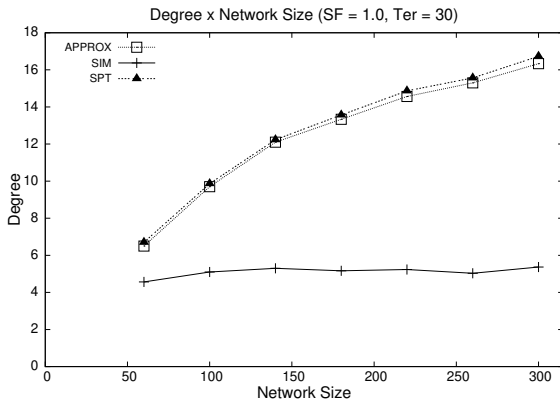
- Algoritmos:
 - DSMDStP é um problema novo.
 - Comparamos com o SPT.
 - Satisfaz as restrições do problema.
 - Algoritmo simples e eficiente*.
 - Similaridades com os algoritmos propostos (principalmente APPROX).
- 4 métricas:
 - Grau máximo.
 - Razão de violação do custo (CVR): $\frac{\sum_{\forall t \in T_{vio}} \frac{dist(s, t, A_f)}{k \cdot dist(s, t, G)}}{|T_{vio}|}$.
 - Pior caso da razão de violação (MAX_CVR).
 - Percentagem de terminais que violam restrições (PVT): $\frac{|T_{vio}|}{|T|} \cdot 100\%$.
 - Percentagem de cenários violados (PVR).

Grau x Terminais



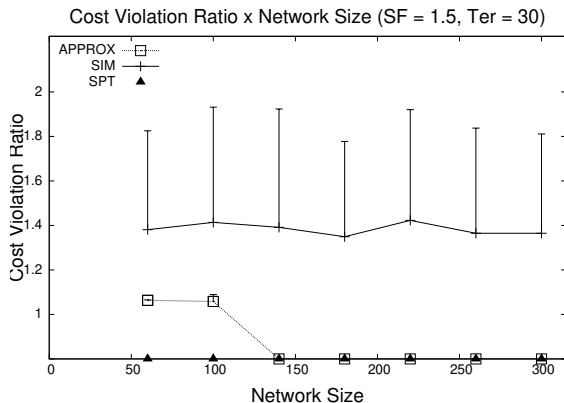
- Crescimento do grau para APPROX e SIM já era esperado.
 - Em cada fase, o grau máximo do(s) algoritmo(s) é afetado pela quantidade de terminais.
- Taxa de crescimento bem maior para SPT.

Grau x Rede, SF = 1



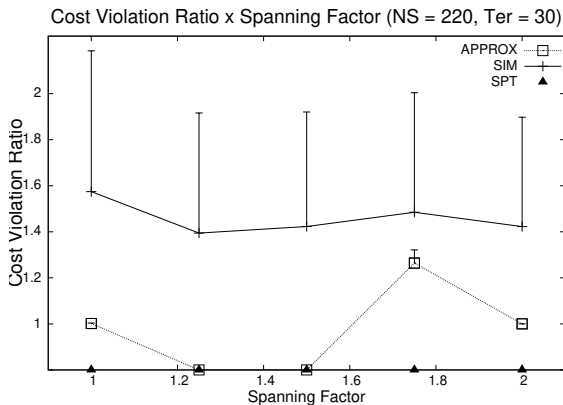
- Curva do APPROX assemelha-se à curva do SPT.
 - $< sf \Leftrightarrow$ menos opções de caminhos para os terminais não-cobertos.
 - Basicamente os caminhos que sobram fazem parte da árvore de custo mínimo (SPT).

CVR x Rede



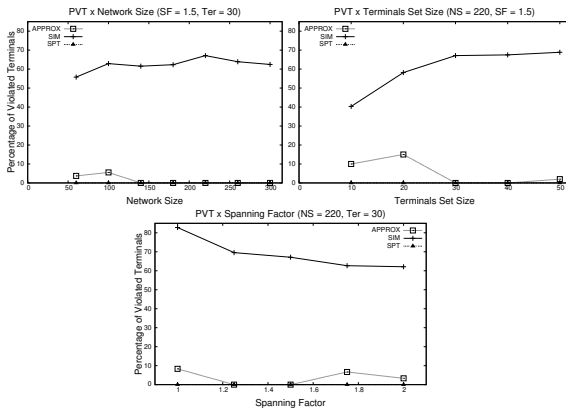
- Valores de APROX próximos de 1, ou não há violação.
- SIM apresenta valores uniformes e baixos.
- MAX_CVR abaixo de 2 para o SIM e próximo do CVR para APPROX.

CVR x Fator de dilatação



- Comportamento semelhante ao gráfico anterior.

PVT



- Percentagem pequena dos terminais que violam para o APPROX (10%).

Experimentos - Resultado

- Grau baixo para os algoritmos propostos.
 - Exceto para APPROX no pior cenário com relação ao FD.
 - Neste cenário, APPROX equivale ao SPT com relação ao grau.
- SIM possui melhor grau e escala bem.
- APPROX é melhor que SIM nas métricas que concerne a propriedade de spanner.
- Com APPROX, em metade das situações não ocorreu violação; na outra metade, a violação foi por um fator (CVR) baixo.
- Com relação ao spanner, SIM apresenta bons resultados para a principal métrica (CVR), a métrica qualitativa.
 - CVR: 1.4.
 - MAX CVR: 2.

Experimentos - Resultado

- Grau baixo para os algoritmos propostos.
 - Exceto para APPROX no pior cenário com relação ao FD.
 - Neste cenário, APPROX equivale ao SPT com relação ao grau.
- SIM possui melhor grau e escala bem.
- APPROX é melhor que SIM nas métricas que concerne a propriedade de spanner.
- Com APPROX, em metade das situações não ocorreu violação; na outra metade, a violação foi por um fator (CVR) baixo.
- Com relação ao spanner, SIM apresenta bons resultados para a principal métrica (CVR), a métrica qualitativa.
 - CVR: 1.4.
 - MAX CVR: 2.

Experimentos - Resultado

- Grau baixo para os algoritmos propostos.
 - Exceto para APPROX no pior cenário com relação ao FD.
 - Neste cenário, APPROX equivale ao SPT com relação ao grau.
- SIM possui melhor grau e escala bem.
- APPROX é melhor que SIM nas métricas que concerne a propriedade de spanner.
- Com APPROX, em metade das situações não ocorreu violação; na outra metade, a violação foi por um fator (CVR) baixo.
- Com relação ao spanner, SIM apresenta bons resultados para a principal métrica (CVR), a métrica qualitativa.
 - CVR: 1.4.
 - MAX CVR: 2.

Experimentos - Resultado

- Grau baixo para os algoritmos propostos.
 - Exceto para APPROX no pior cenário com relação ao FD.
 - Neste cenário, APPROX equivale ao SPT com relação ao grau.
- SIM possui melhor grau e escala bem.
- APPROX é melhor que SIM nas métricas que concerne a propriedade de spanner.
- Com APPROX, em metade das situações não ocorreu violação; na outra metade, a violação foi por um fator (CVR) baixo.
- Com relação ao spanner, SIM apresenta bons resultados para a principal métrica (CVR), a métrica qualitativa.
 - CVR: 1.4.
 - MAX CVR: 2.

Funções submodulares e Matroids

- MCG pode ser modelado através dos conceitos de *funções submodulares* e *matroids* ([Călinescu et al. 2011]).

Maximização de função submodular

- O problema abordado em [Călinescu et al. 2011] foi o da maximização de função submodular sob restrição de uma matroid (SUB-M).
- Este problema já tinha sido abordado em outros trabalhos [Nemhauser et al. 1978, Fisher et al. 1978].
- Os autores em [Călinescu et al. 2011] conseguem generalizar o resultado para qualquer tipo de matroid (incluindo matroid de partição), mantendo a garantia de aproximação $((e/(e - 1)))$.
 - O resultado dos autores é probabilístico:
 - Garantia do grau passaria a ser probabilística.
 - Para o MSC, o limite no número de iterações para cobrir todos os terminals seria probabilístico.

Maximização de função submodular

- O problema abordado em [Călinescu et al. 2011] foi o da maximização de função submodular sob restrição de uma matroid (SUB-M).
- Este problema já tinha sido abordado em outros trabalhos [Nemhauser et al. 1978, Fisher et al. 1978].
- Os autores em [Călinescu et al. 2011] conseguem generalizar o resultado para qualquer tipo de matroid (incluindo matroid de partição), mantendo a garantia de aproximação $((e/(e-1)))$.
 - O resultado dos autores é probabilístico:
 - Garantia do grau passaria a ser probabilística.
 - Para o MSC, o limite no número de iterações para cobrir todos os terminais seria probabilístico.

- A set of small navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

- Tentar garantir as restrições de spanner.
- Solução distribuída.
- Modelar a solução através de função submodular e matroids.

Trabalhos Futuros

- Tentar garantir as restrições de spanner.
- Solução distribuída.
- Modelar a solução através de função submodular e matroids.

Limite no grau máximo de $G_{\sqrt{I}-Par}$ (Algoritmo 2)

Demonstração.

- As árvores T_v computadas na linha 5 de Algoritmo 1 são disjuntas.
- Cada nó em T_v possui grau máximo de \sqrt{l} .
- A_{Roots} é uma SPT de s para todos os nós em $Roots$ e a cardinalidade máxima de $Roots$ é de $\sqrt{l} + 2$ (Lema 2) \Rightarrow grau máximo de A_{Roots} é $\sqrt{l} + 2$.
- $G_{\sqrt{l}-Par}$ corresponde à união de todas estas árvores (A_{Roots} e T_v) \Rightarrow grau máximo de qualquer nó é de $\sqrt{l} + \sqrt{l} + 2 = 2\sqrt{l} + 2$.



Limite da interseção entre a solução do MSC e cada partição de V_1

► Voltar

Demonstração.

- $d^* \geq \text{val}(S^*)$ para uma solução ótima S^* de β .
- $|V_2| \leq l$.
- O lema é direto.



- $d^* \geq \text{val}(S^*)$ para uma solução ótima S^* de β .
- $|V_2| \leq l$.
- O lema é direto.





- $d^* \geq \text{val}(S^*)$ para uma solução ótima S^* de β .
- $|V_2| \leq l$.
- O lema é direto.



Custo máximo dos caminhos para os terminais

Demonstração.

- Se t é coberto na primeira fase, $\text{dist}(s, t, G_f) \leq k \cdot \text{dist}(s, t, G)$.
- Todos os nós v cobertos na fase 1 fazem parte de um caminho entre s e um t t.q. seu custo é $\leq k \cdot \text{dist}(s, t, G) \rightarrow \leq k \cdot \text{dist}(s, t_{\max}, G)$.
- $\forall t \in T$ cobertos apenas na fase 2, $x_{u,v}$ e $(x_{u,v}, t)$ são inseridas na instância do MSC se $\text{dist}(s, u, G) + C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G)$.
- Em particular, $C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G)$.
- Como $\text{dist}(s, u, G) \leq k \cdot \text{dist}(s, t_{\max}, G) \Rightarrow \text{dist}(s, t, G_f) \leq k \cdot (\text{dist}(s, t, G) + \text{dist}(s, t_{\max}, G))$.



Custo máximo dos caminhos para os terminais

Demonstração.

- Se t é coberto na primeira fase, $\text{dist}(s, t, G_f) \leq k \cdot \text{dist}(s, t, G)$.
- Todos os nós v cobertos na fase 1 fazem parte de um caminho entre s e um t t.q. seu custo é $\leq k \cdot \text{dist}(s, t, G) \rightarrow \leq k \cdot \text{dist}(s, t_{\max}, G)$.
- $\forall t \in T$ cobertos apenas na fase 2, $x_{u,v}$ e $(x_{u,v}, t)$ são inseridas na instância do MSC se $\text{dist}(s, u, G) + C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G)$.
- Em particular, $C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G)$.
- Como $\text{dist}(s, u, G) \leq k \cdot \text{dist}(s, t_{\max}, G) \Rightarrow \text{dist}(s, t, G_f) \leq k \cdot (\text{dist}(s, t, G) + \text{dist}(s, t_{\max}, G))$.



Demonstração.



Demonstração.

- \mathcal{A}_f é gerada computando caminhos de custo mínimo em G_f entre s e $t \in T$.
- O teorema segue do resultado de grau máximo (Lema 6) e do resultado de custo máximo (Lema 7).



- \mathcal{A}_f é gerada computando caminhos de custo mínimo em G_f entre s e $t \in T$.
- O teorema segue do resultado de grau máximo (Lema 6) e do resultado de custo máximo (Lema 7).



5

Número de iterações do SIM [▶ Voltar](#)

Demonstração.

- $\lfloor \sqrt{I} \rfloor + (2 \cdot \mathcal{F}) + (\mathcal{F} \cdot \frac{\mathcal{F}}{\lfloor \sqrt{I} \rfloor}) = \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor}$.
- Como $\lfloor \sqrt{I} \rfloor^2$ é o maior quadrado perfeito menor que $I \rightarrow (I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (prova adiada).
 - $\rightarrow \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor} \leq \lfloor \sqrt{I} \rfloor + \frac{2 \cdot \lfloor \sqrt{I} \rfloor}{\lfloor \sqrt{I} \rfloor} = \lfloor \sqrt{I} \rfloor + 2$.
- Provar que $(I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (quadrados perfeitos).
 - Seja SR_x um quadrado perfeito e R_x sua raiz.
 - Para dois quadrados perfeitos consecutivos SR_i e $SR_j \rightarrow SR_j - SR_i = R_i + R_j$.
 - Em outras palavras: $SR_j - SR_i = 2 \cdot R_i + 1$.
 - Seja X um quadrado não-perfeito e seja GSR_X o maior quadrado perfeito menor que X . Seja R_X a raiz de GSR_X .



Número de iterações do SIM

Demonstração.

- $\lfloor \sqrt{I} \rfloor + (2 \cdot \mathcal{F}) + (\mathcal{F} \cdot \frac{\mathcal{F}}{\lfloor \sqrt{I} \rfloor}) = \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor}$.
- Como $\lfloor \sqrt{I} \rfloor^2$ é o maior quadrado perfeito menor que $I \rightarrow (I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (prova adiada).
 - $\rightarrow \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor} \leq \lfloor \sqrt{I} \rfloor + \frac{2 \cdot \lfloor \sqrt{I} \rfloor}{\lfloor \sqrt{I} \rfloor} = \lfloor \sqrt{I} \rfloor + 2$.
- Provar que $(I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (quadrados perfeitos).
 - Seja SR_x um quadrado perfeito e R_x sua raiz.
 - Para dois quadrados perfeitos consecutivos SR_i e $SR_j \rightarrow SR_j - SR_i = R_i + R_j$.
 - Em outras palavras: $SR_j - SR_i = 2 \cdot R_i + 1$.
 - Seja X um quadrado não-perfeito e seja GSR_X o maior quadrado perfeito menor que X . Seja R_X a raiz de GSR_X .



Número de iterações do SIM

Demonstração.

- $\lfloor \sqrt{I} \rfloor + (2 \cdot \mathcal{F}) + (\mathcal{F} \cdot \frac{\mathcal{F}}{\lfloor \sqrt{I} \rfloor}) = \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor}$.
- Como $\lfloor \sqrt{I} \rfloor^2$ é o maior quadrado perfeito menor que $I \rightarrow (I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (prova adiada).
 - $\rightarrow \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor} \leq \lfloor \sqrt{I} \rfloor + \frac{2 \cdot \lfloor \sqrt{I} \rfloor}{\lfloor \sqrt{I} \rfloor} = \lfloor \sqrt{I} \rfloor + 2$.
- Provar que $(I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (quadrados perfeitos).
 - Seja SR_x um quadrado perfeito e R_x sua raiz.
 - Para dois quadrados perfeitos consecutivos SR_i e $SR_j \rightarrow SR_j - SR_i = R_i + R_j$.
 - Em outras palavras: $SR_j - SR_i = 2 \cdot R_i + 1$.
 - Seja X um quadrado não-perfeito e seja GSR_X o maior quadrado perfeito menor que X . Seja R_X a raiz de GSR_X .
 - $X - GSR_X < 2 \cdot R_X$.



Número de iterações do SIM

Demonstração.

- $\lfloor \sqrt{I} \rfloor + (2 \cdot \mathcal{F}) + (\mathcal{F} \cdot \frac{\mathcal{F}}{\lfloor \sqrt{I} \rfloor}) = \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor}$.
- Como $\lfloor \sqrt{I} \rfloor^2$ é o maior quadrado perfeito menor que $I \rightarrow (I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (prova adiada).
 - $\rightarrow \lfloor \sqrt{I} \rfloor + \frac{I - \lfloor \sqrt{I} \rfloor^2}{\lfloor \sqrt{I} \rfloor} \leq \lfloor \sqrt{I} \rfloor + \frac{2 \cdot \lfloor \sqrt{I} \rfloor}{\lfloor \sqrt{I} \rfloor} = \lfloor \sqrt{I} \rfloor + 2$.
- Provar que $(I - \lfloor \sqrt{I} \rfloor^2) \leq 2 \cdot \lfloor \sqrt{I} \rfloor$ (quadrados perfeitos).
 - Seja SR_x um quadrado perfeito e R_x sua raiz.
 - Para dois quadrados perfeitos consecutivos SR_i e $SR_j \rightarrow SR_j - SR_i = R_i + R_j$.
 - Em outras palavras: $SR_j - SR_i = 2 \cdot R_i + 1$.
 - Seja X um quadrado não-perfeito e seja GSR_X o maior quadrado perfeito menor que X . Seja R_X a raiz de GSR_X .
 - $X - GSR_X < 2 \cdot R_X$.







Grafo gerado por SIM é uma arborescência [▶ Voltar](#)

Demonstração.

- No início, C contém apenas s .
- Ao final da primeira iteração, \mathcal{A}_f será uma arborescência, pois apenas caminhos de menor custo a partir de s são adicionados.
- De forma análoga, a partir da segunda iteração, para cada $(x_{u,v})$, apenas caminhos de custo mínimo entre u e t são adicionados a \mathcal{A}_f .
 - Estes caminhos são formados por nós apenas de U (exceto para o primeiro nó u) \rightarrow inexistência de ciclos e múltiplos caminhos entre nós.



- No início, C contém apenas s .
- Ao final da primeira iteração, \mathcal{A}_f será uma arborescência, pois apenas caminhos de menor custo a partir de s são adicionados.
- De forma análoga, a partir da segunda iteração, para cada $(x_{u,v})$, apenas caminhos de custo mínimo entre u e t são adicionados a \mathcal{A}_f .
 - Estes caminhos são formados por nós apenas de U (exceto para o primeiro nó u) \rightarrow inexistência de ciclos e múltiplos caminhos entre nós.



Custo máximo dos caminhos gerados por SIM

Demonstração.

- Seja t_h o terminal $u \in \text{Next}_{\text{ter}}^{\sqrt{l}}$ cujo valor de $\text{dist}(s, u, G)$ é máximo.
- Todos os caminhos adicionados a \mathcal{A}_f na iteração i terão custo $\leq k \cdot \text{dist}(s, t_h, G)$.
 - $(x_{u,v}, t)$ pertencerá a ε se $C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G) \leq k \cdot \text{dist}(s, t_h, G)$.
- Haverá no máximo $\lfloor \sqrt{l} \rfloor + 2$ iterations (Lema 9).
- Seja t um terminal adicionado a \mathcal{A}_f na iteração j .
- t pode ser alcançado a partir de s através de uma série de caminhos adicionados a \mathcal{A}_f em $j - 1$ iterações.
- Em cada iteração, os terminais em $\text{Next}_{\text{ter}}^{\sqrt{l}}$ seguem uma ordem não-decrescente de custo.
- O teorema segue.



Custo máximo dos caminhos gerados por SIM

Demonstração.

- Seja t_h o terminal $u \in \text{Next}_{\text{ter}}^{\sqrt{l}}$ cujo valor de $\text{dist}(s, u, G)$ é máximo.
- Todos os caminhos adicionados a \mathcal{A}_f na iteração i terão custo $\leq k \cdot \text{dist}(s, t_h, G)$.
 - $(x_{u,v}, t)$ pertencerá a ε se $C(u, v) + \text{dist}(v, t, G(U)) \leq k \cdot \text{dist}(s, t, G) \leq k \cdot \text{dist}(s, t_h, G)$.
- Haverá no máximo $\lfloor \sqrt{l} \rfloor + 2$ iterations (Lema 9).
- Seja t um terminal adicionado a \mathcal{A}_f na iteração j .
- t pode ser alcançado a partir de s através de uma série de caminhos adicionados a \mathcal{A}_f em $j - 1$ iterações.
- Em cada iteração, os terminais em $\text{Next}_{\text{ter}}^{\sqrt{l}}$ seguem uma ordem não-decrescente de custo.
- O teorema segue.



Custo máximo dos caminhos gerados por SIM

▶ Voltar

Demonstração.

- Seja t_h o terminal $u \in Next_{ter}^{\sqrt{l}}$ cujo valor de $dist(s, u, G)$ é máximo.
- Todos os caminhos adicionados a \mathcal{A}_f na iteração i terão custo $\leq k \cdot dist(s, t_h, G)$.
 - $(x_{u,v}, t)$ pertencerá a ε se $C(u, v) + dist(v, t, G(U)) \leq k \cdot dist(s, t, G) \leq k \cdot dist(s, t_h, G)$.
- Haverá no máximo $\lfloor \sqrt{l} \rfloor + 2$ iterations (Lema 9).
- Seja t um terminal adicionado a \mathcal{A}_f na iteração j .
- t pode ser alcançado a partir de s através de uma série de caminhos adicionados a \mathcal{A}_f em $j - 1$ iterações.
- Em cada iteração, os terminais em $Next_{ter}^{\sqrt{l}}$ seguem uma ordem não-decrescente de custo.
- O teorema segue.



Complexidade [▶ Voltar](#)

- Procedimento *CompPar* $\rightarrow (\sqrt{|T|} + 2)O(|V|^3)$:
 - $\sqrt{|T|} + 2$: número de iterações (Lema 2).
 - $O(|V|^3)$: construção da SPT para os candidatos a $\sqrt{|T|}$ -bad node.
- Procedimento *CompGraphFirstPh*: $O(|V|^2)$ (construção da SPT).
- Construção da instância do MSC:
 - Composição de V_1 : $O(|V|^2)$.
 - Composição de ε : $O(|V|^3)$ (executar o algoritmo de APSP).
- Algoritmo guloso de [Chekuri and Kumar 2004]:
 - Depende da execução do algoritmo para o MCG.
 - O algoritmo para o MCG depende da execução de oráculo $\rightarrow O(|V|^2|T|)$ no nosso caso.
 - Complexidade do algoritmo para o MCG: $O(|V|^3|T|^2)$.
 - O algoritmo para o MSC aplica iterativamente o algoritmo para o MCG $\rightarrow O((\log |T|)(|V|^3|T|^2))$.
- Loop entre as linhas 8 e 12:
 - Linha 10 se beneficia da execução do algoritmo do APSP.
 - Linha 9: $|D|$ é proporcional a $|S_{(u,v)}|$ que é proporcional a $O(|V|)$.
 - Linha 8: $|V_2| = |T|$.
 - Complexidade do loop: $|T|O(|V|)$.
- Complexidade final: $O((\log |T|)(|V|^3|T|^2))$.

Complexidade

- Procedimento *CompPar* $\rightarrow (\sqrt{|T|} + 2)O(|V|^3)$:
 - $\sqrt{|T|} + 2$: número de iterações (Lema 2).
 - $O(|V|^3)$: construção da SPT para os candidatos a $\sqrt{|T|}$ -bad node.
- Procedimento *CompGraphFirstPh*: $O(|V|^2)$ (construção da SPT).
- Construção da instância do MSC:
 - Composição de V_1 : $O(|V|^2)$.
 - Composição de ε : $O(|V|^3)$ (executar o algoritmo de APSP).
- Algoritmo guloso de [Chekuri and Kumar 2004]:
 - Depende da execução do algoritmo para o MCG.
 - O algoritmo para o MCG depende da execução de oráculo $\rightarrow O(|V|^2|T|)$ no nosso caso.
 - Complexidade do algoritmo para o MCG: $O(|V|^3|T|^2)$.
 - O algoritmo para o MSC aplica iterativamente o algoritmo para o MCG $\rightarrow O((\log |T|)(|V|^3|T|^2))$.
- Loop entre as linhas 8 e 12:
 - Linha 10 se beneficia da execução do algoritmo do APSP.
 - Linha 9: $|D|$ é proporcional a $|S_{(u,v)}|$ que é proporcional a $O(|V|)$.
 - Linha 8: $|V_2| = |T|$.
 - Complexidade do loop: $|T|O(|V|)$.
- Complexidade final: $O((\log |T|)(|V|^3|T|^2))$.

Complexidade

- Procedimento *CompPar* $\rightarrow (\sqrt{|T|} + 2)O(|V|^3)$:
 - $\sqrt{|T|} + 2$: número de iterações (Lema 2).
 - $O(|V|^3)$: construção da SPT para os candidatos a $\sqrt{|T|}$ -bad node.
- Procedimento *CompGraphFirstPh*: $O(|V|^2)$ (construção da SPT).
- Construção da instância do MSC:
 - Composição de V_1 : $O(|V|^2)$.
 - Composição de ε : $O(|V|^3)$ (executar o algoritmo de APSP).
- Algoritmo guloso de [Chekuri and Kumar 2004]:
 - Depende da execução do algoritmo para o MCG.
 - O algoritmo para o MCG depende da execução de oráculo $\rightarrow O(|V|^2|T|)$ no nosso caso.
 - Complexidade do algoritmo para o MCG: $O(|V|^3|T|^2)$.
 - O algoritmo para o MSC aplica iterativamente o algoritmo para o MCG $\rightarrow O((\log |T|)(|V|^3|T|^2))$.
- Loop entre as linhas 8 e 12:
 - Linha 10 se beneficia da execução do algoritmo do APSP.
 - Linha 9: $|D|$ é proporcional a $|S_{(u,v)}|$ que é proporcional a $O(|V|)$.
 - Linha 8: $|V_2| = |T|$.
 - Complexidade do loop: $|T|O(|V|)$.
- Complexidade final: $O((\log |T|)(|V|^3|T|^2))$.

Complexidade

- Complexidade semelhante em algumas partes. O tamanho de V_2 é $\sqrt{|T|}$ ao invés de $O(|T|)$.
- Linha 4: $O(|V|^3)$ (semelhante ao *APPROX*).
- Loop (5-7):
 - $\lfloor \sqrt{|T|} \rfloor$: número máximo de iterações.
 - $O(|T|\sqrt{|T|})$: tamanho máximo de *Marked* na última iteração.
 - As outras instruções se beneficiam dos resultados da APSP e da SPT.
 - Complexidade do loop: $O(|T|^2)$.
- Algoritmo para o MSC (linha 8): $O((\log \sqrt{|T|})(|V|^3|T|))$ (novo tamanho de V_2).
- Loop (10-15):
 - $\sqrt{|T|} + 2$: número máximo de iterações.
 - $O(V)$: tamanho de D .
 - $O(1)$ para as outras instruções (também se beneficia do resultado do APSP e da SPT).
- Complexidade final: $O((\log \sqrt{|T|})(|V|^3|T|\sqrt{|T|}))$.

Complexidade

- Complexidade semelhante em algumas partes. O tamanho de V_2 é $\sqrt{|T|}$ ao invés de $O(|T|)$.
- Linha 4: $O(|V|^3)$ (semelhante ao *APPROX*).
- Loop (5-7):
 - $\lfloor \sqrt{|T|} \rfloor$: número máximo de iterações.
 - $O(|T|\sqrt{|T|})$: tamanho máximo de *Marked* na última iteração.
 - As outras instruções se beneficiam dos resultados da APSP e da SPT.
 - Complexidade do loop: $O(|T|^2)$.
- Algoritmo para o MSC (linha 8): $O((\log \sqrt{|T|})(|V|^3|T|))$ (novo tamanho de V_2).
- Loop (10-15):
 - $\sqrt{|T|} + 2$: número máximo de iterações.
 - $O(V)$: tamanho de D .
 - $O(1)$ para as outras instruções (também se beneficia do resultado do APSP e da SPT).
- Complexidade final: $O((\log \sqrt{|T|})(|V|^3|T|\sqrt{|T|}))$.



Arora, S. and Barak, B. (2009).

Computational Complexity: A Modern Approach.

Cambridge University Press, New York, NY, USA, 1st edition.



Călinescu, G., Chekuri, C., Pál, M., and Vondrák, J. (2011).

Maximizing a monotone submodular function subject to a matroid constraint.

SIAM J. Comput., 40(6):1740–1766.



Chekuri, C. and Kumar, A. (2004).

Maximum coverage problem with group budget constraints and applications.

In *APPROX-RANDOM*, pages 72–83. Springer-Verlag New York, Inc.



Elkin, M. and Kortsarz, G. (2006).

An approximation algorithm for the directed telephone multicast problem.

Algorithmica, 45:569–583.



Elkin, M. and Solomon, S. (2009).

Narrow-shallow-low-light trees with and without steiner points.

In *ESA*, pages 215–226.



Elkin, M. and Solomon, S. (2011).

Narrow-shallow-low-light trees with and without steiner points.
SIAM J. Discret. Math., 25(1):181–210.



Fisher, M. L., Nemhauser, G. L., and Wolsey, L. A. (1978).

An analysis of approximations for maximizing submodular set functions-ii.

In *Polyhedral Combinatorics*, volume 8 of *Mathematical Programming Studies*, pages 73–87. Springer Berlin Heidelberg.



Fraigniaud, P. (2001).

Approximation algorithms for minimum-time broadcast under the vertex-disjoint paths mode.

In *Proceedings of the 9th Annual European Symposium on Algorithms*, ESA '01, pages 440–451, London, UK. Springer-Verlag.



Khandekar, R., Kortsarz, G., and Nutov, Z. (2011).

Network-design with degree constraints.

In Goldberg, L., Jansen, K., Ravi, R., and Rolim, J., editors, *Approximation, Randomization, and Combinatorial Optimization*.

Algorithms and Techniques, volume 6845 of *Lecture Notes in Computer Science*, pages 289–301. Springer Berlin.



Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978).

An analysis of approximations for maximizing submodular set functions-i.

Mathematical Programming, 14:265–294.



Williamson, D. P. and Shmoys, D. B. (2011).

The Design of Approximation Algorithms.

Cambridge University Press, New York, NY, USA, 1st edition.