Análise de Algoritmos

Parte destes slides são adaptações de slides

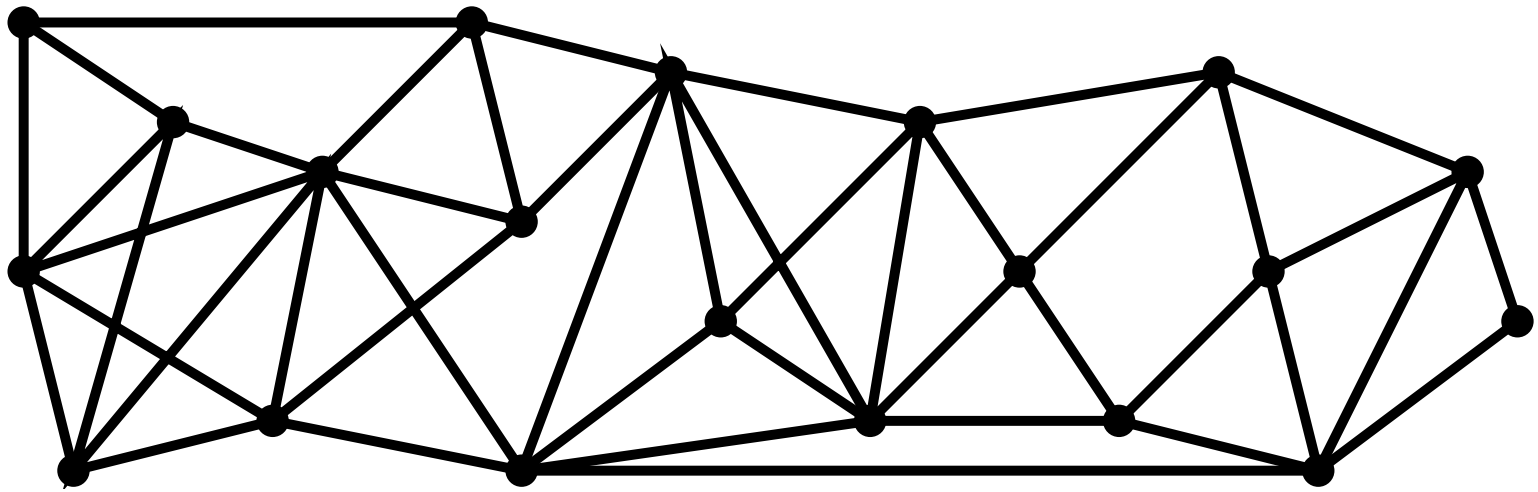
do Prof. Paulo Feofiloff e do Prof. José Coelho de Pina.

Problema do Caixeiro Viajante

Dados

grafo G

comprimento l_{ij} da aresta ij $\quad (ij \in E_G)$

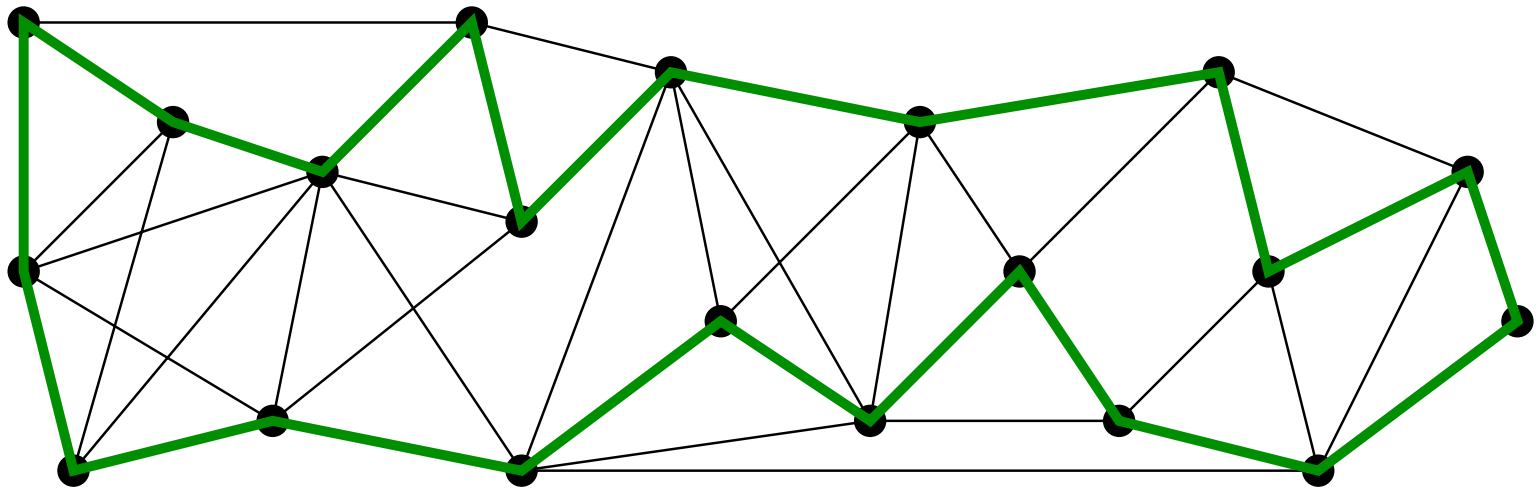


Problema do Caixeiro Viajante

Dados

grafo G

comprimento l_{ij} da aresta ij $\quad (ij \in E_G)$



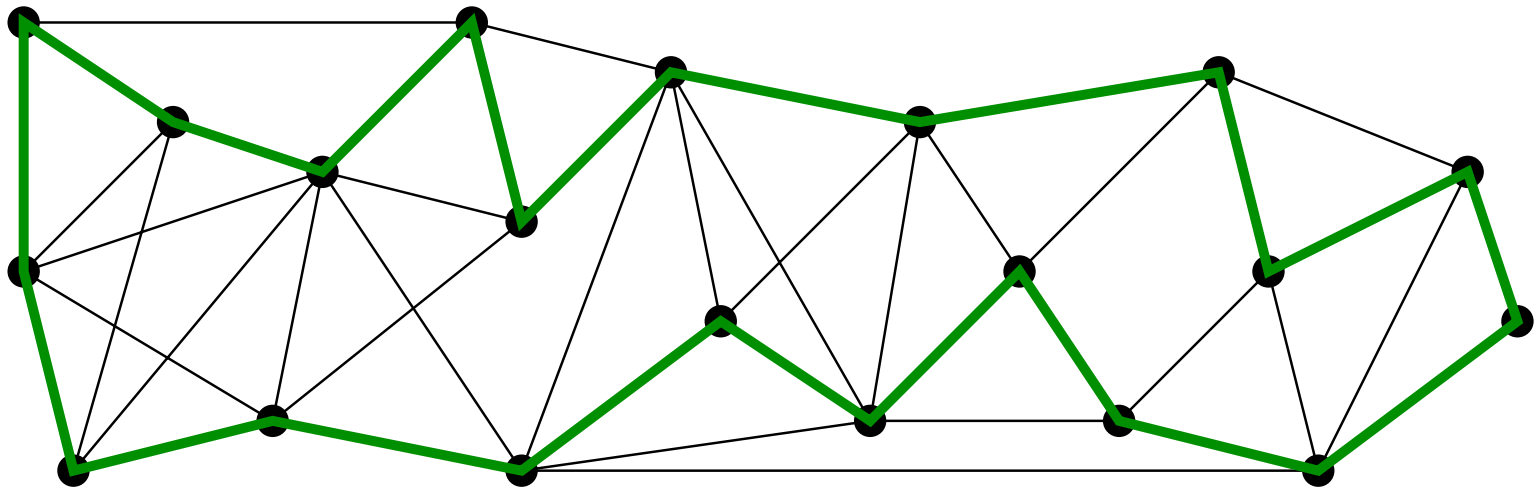
Circuito hamiltoniano: circuito que passa por todos os vértices

Problema do Caixeiro Viajante

Dados

grafo G

comprimento l_{ij} da aresta ij $\quad (ij \in E_G)$



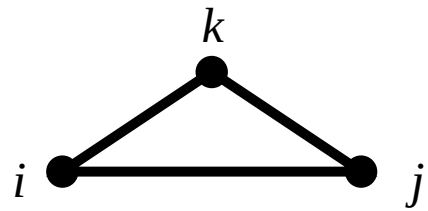
Circuito hamiltoniano: circuito que passa por todos os vértices

Problema (TSP): Dados G e l, encontrar circuito hamiltoniano C em G de comprimento $l(C)$ mínimo.

Variantes do Caixeiro Viajante

TSP métrico

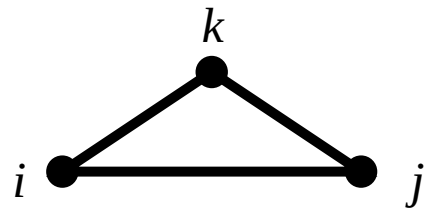
- grafo completo
- função comprimento l satisfaz
  desigualdade triangular: $l_{ij} \leq l_{ik} + l_{kj} \quad \forall i, j, k \in V_G$



Variantes do Caixeiro Viajante

TSP métrico

- grafo completo
- função comprimento l satisfaz
  desigualdade triangular: $l_{ij} \leq l_{ik} + l_{kj} \quad \forall i, j, k \in V_G$

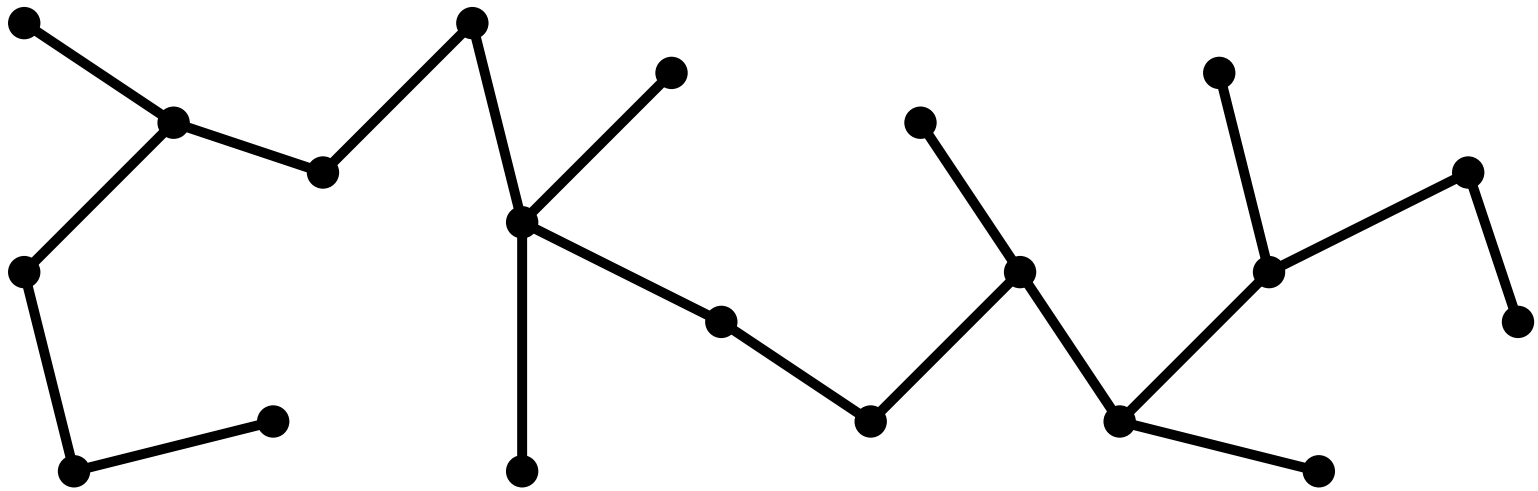


Resultados Conhecidos:

- NP-difícil mesmo se $l_e \in \{1, 2\}$ para toda aresta e [GJ79]
- Difícil de aproximar [SG76]
- $3/2$-aproximação para o caso métrico [C76]
- PTAS para o caso euclideano [A98,M99]

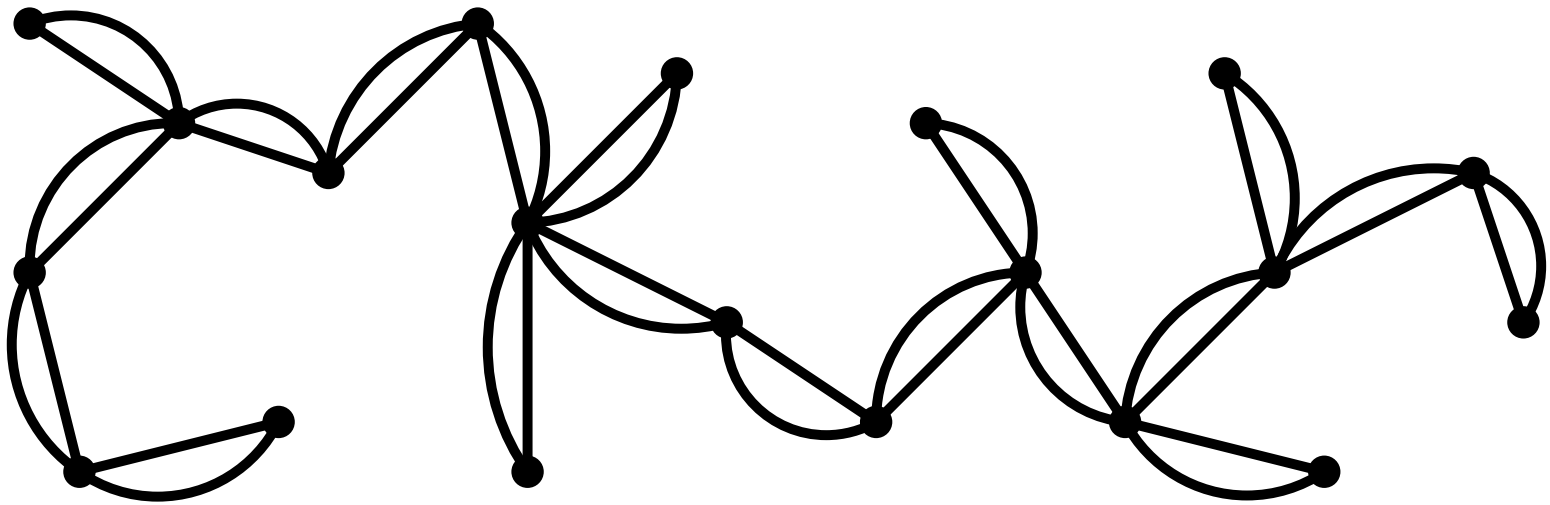
2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)



2-aproximação p/o TSP Métrico

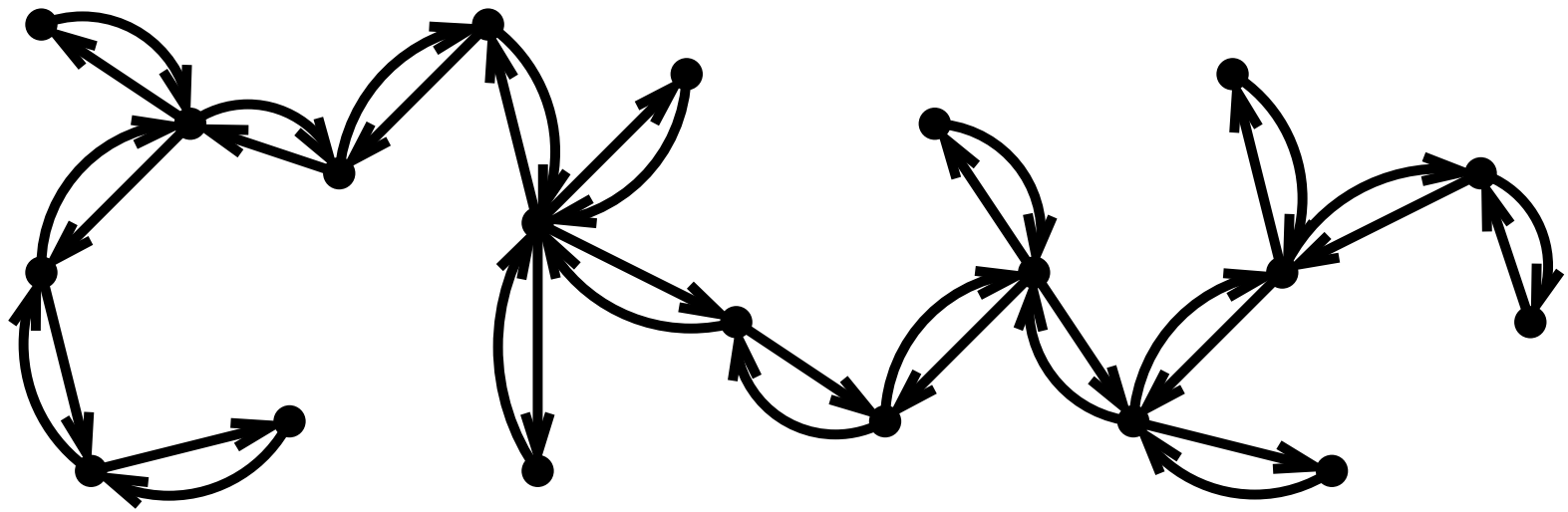
(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)



2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa
exatamente uma vez por cada aresta

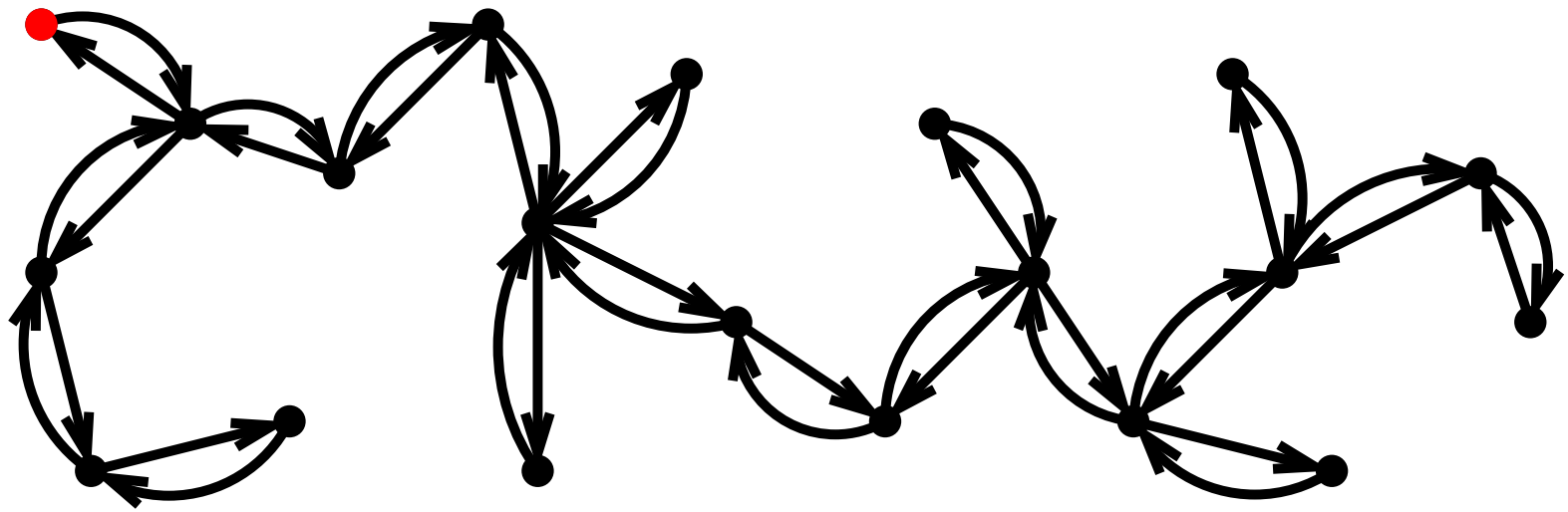


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa
exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

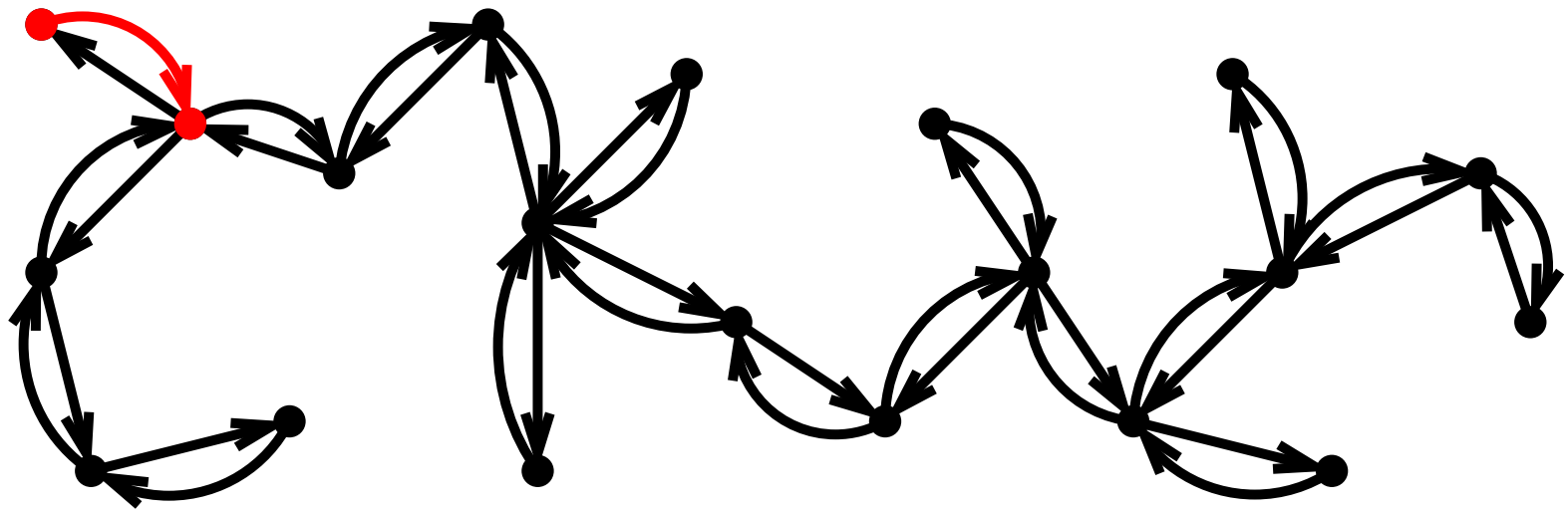


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

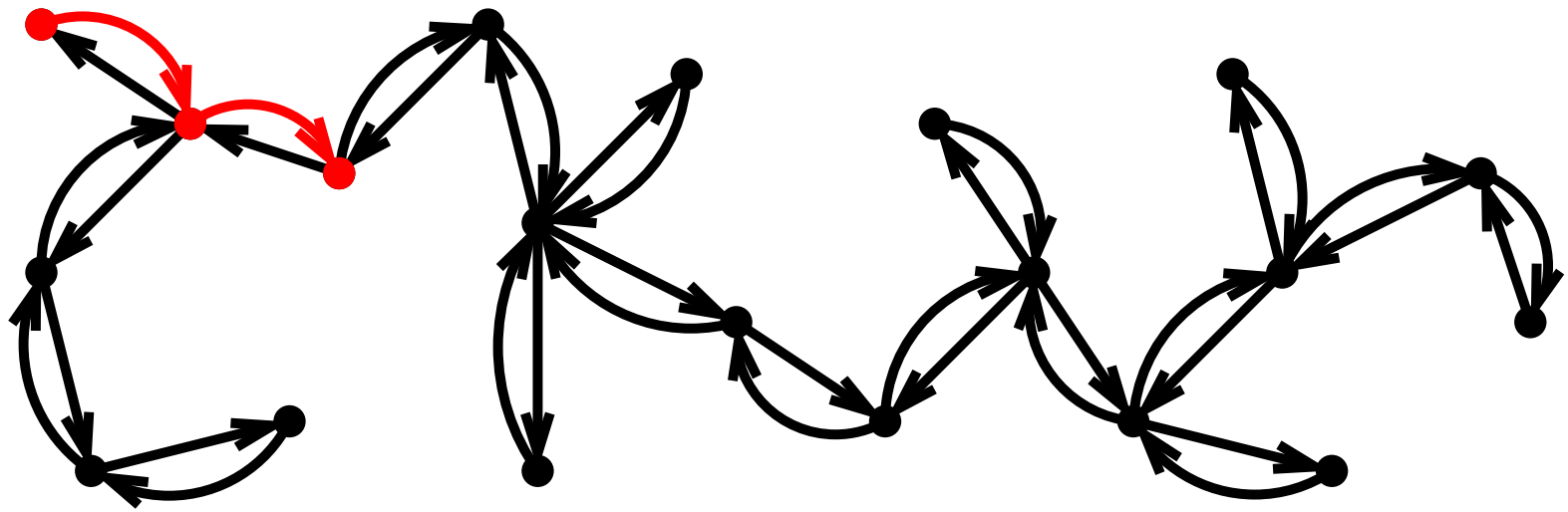


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

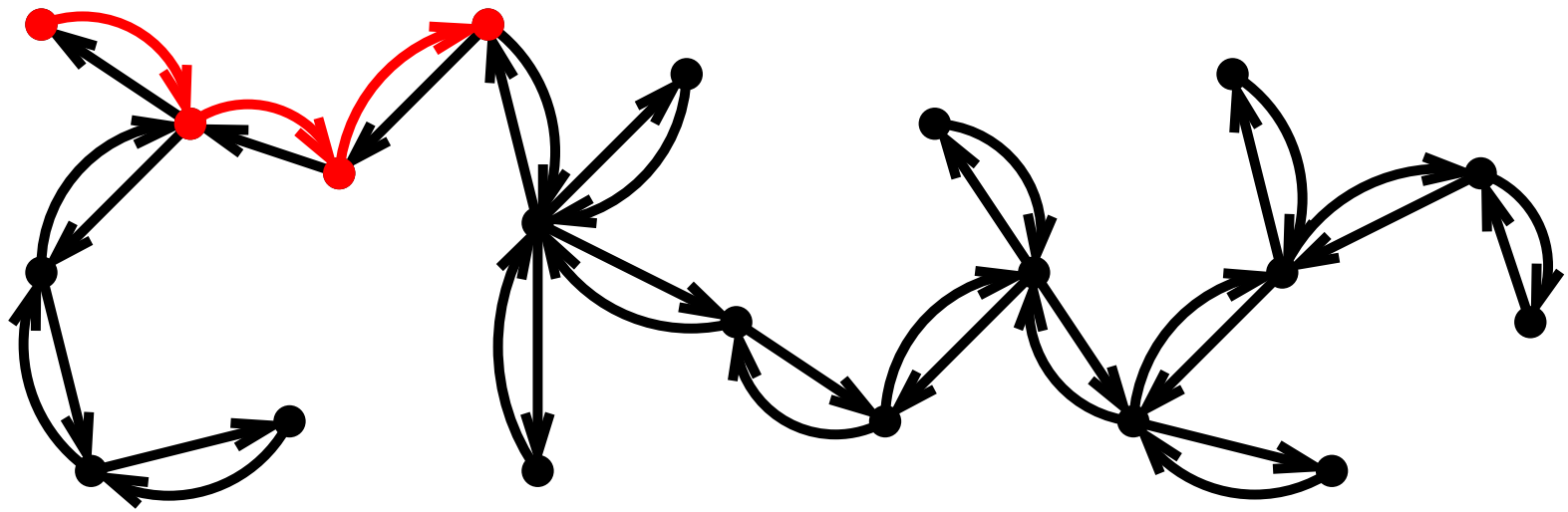


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

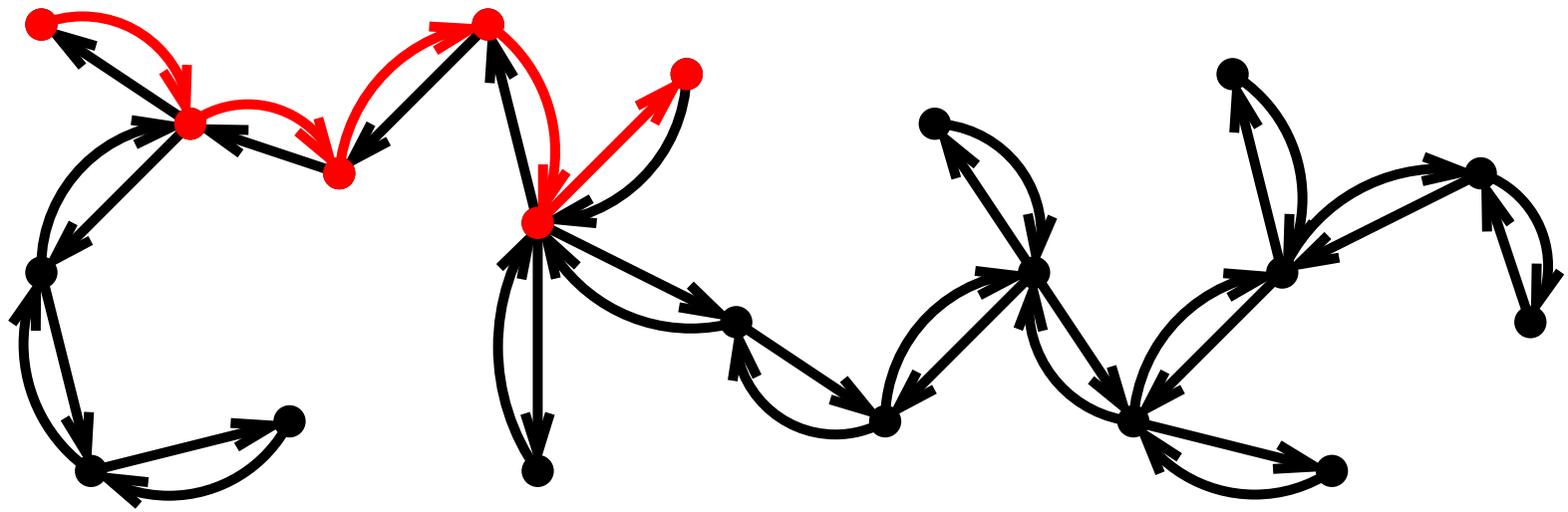


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

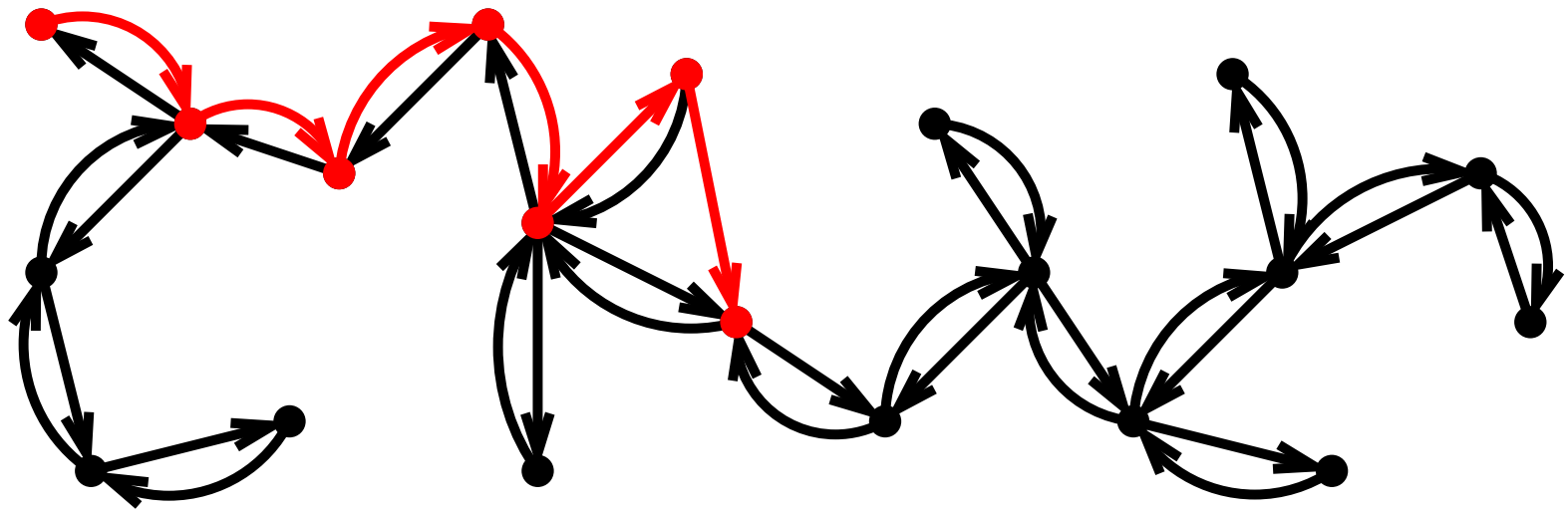


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

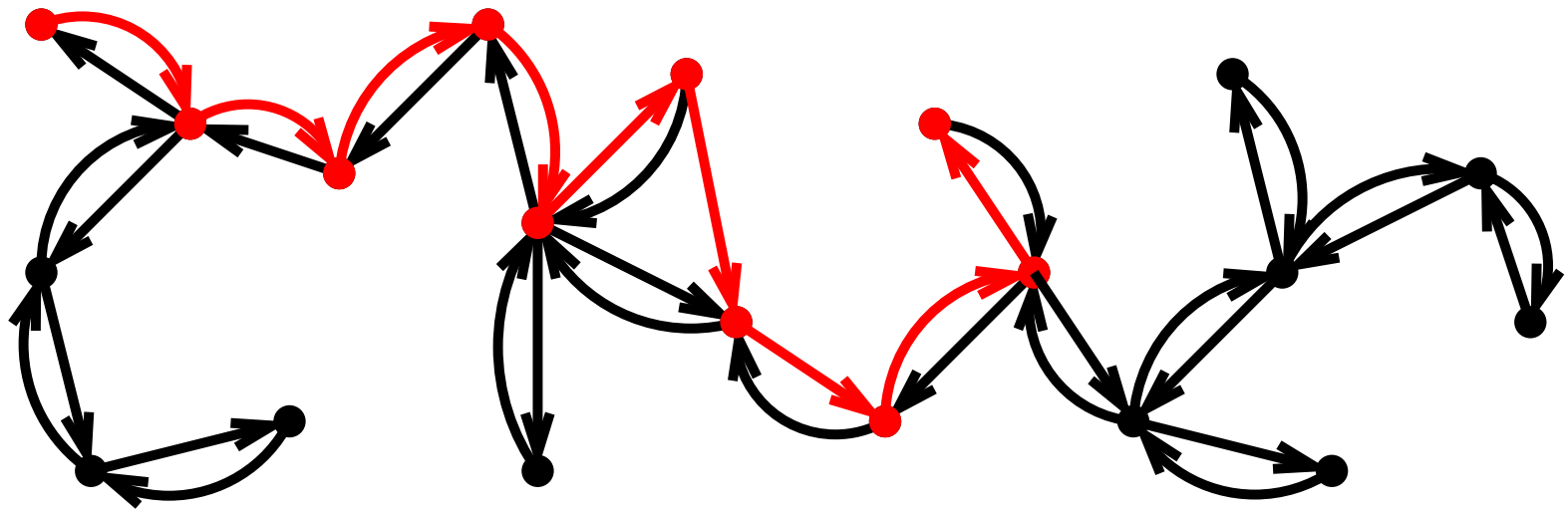


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

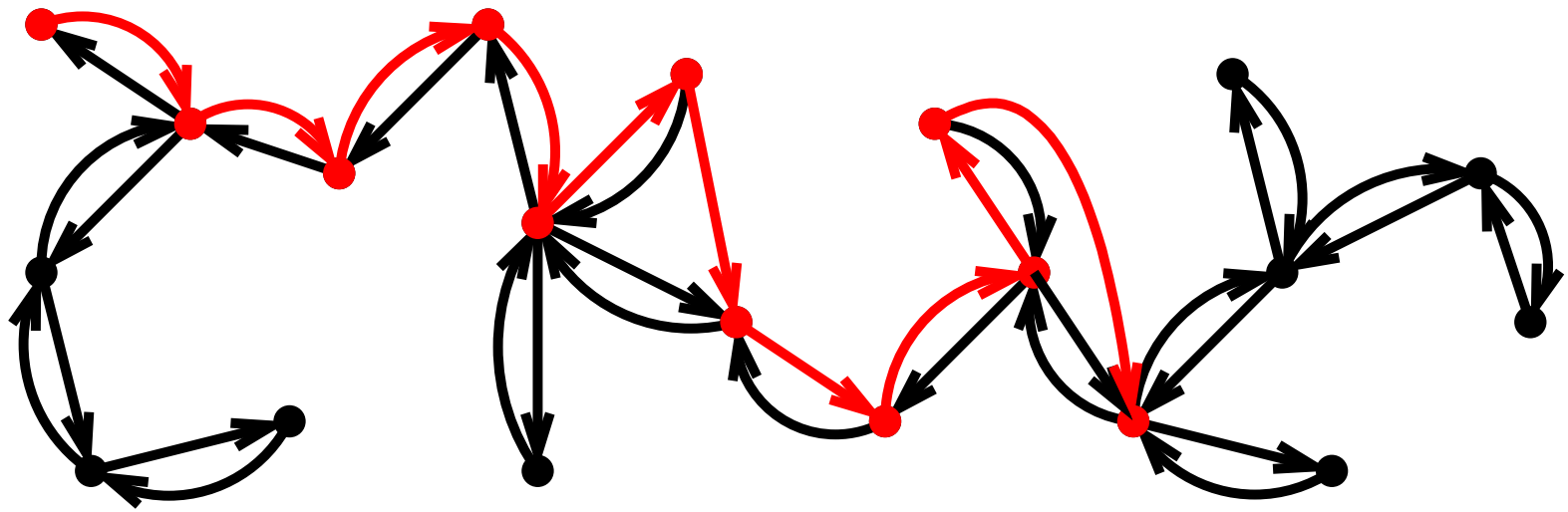


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa
exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

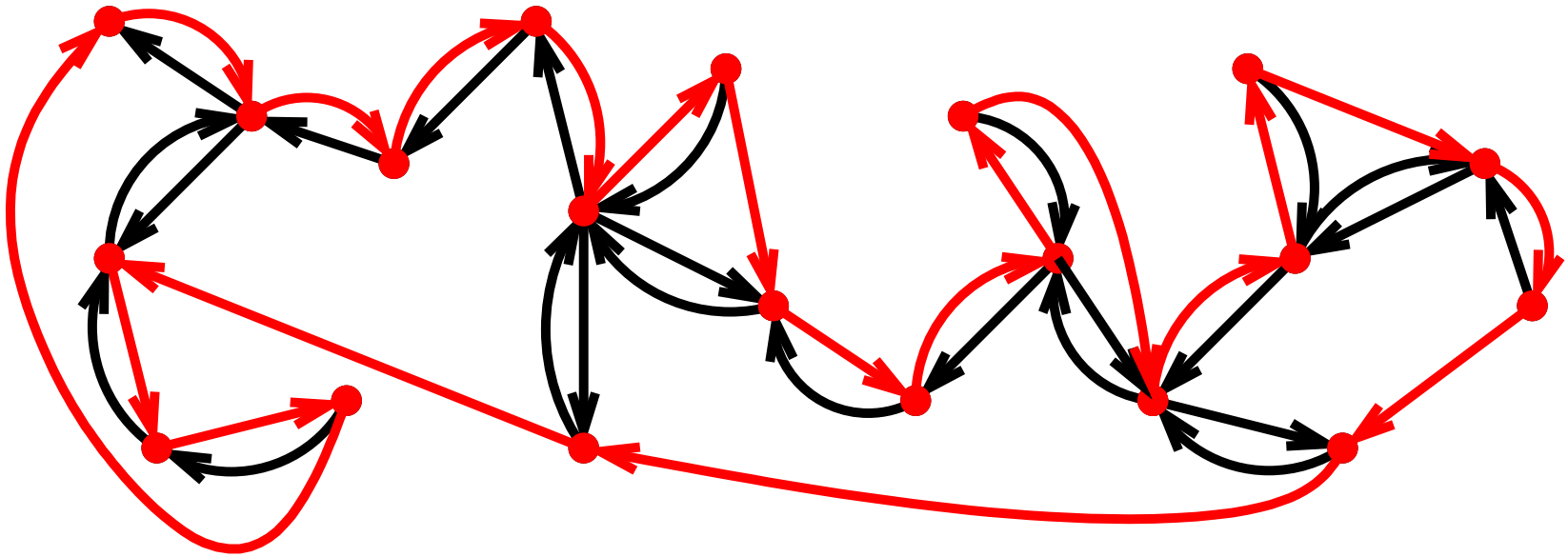


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C

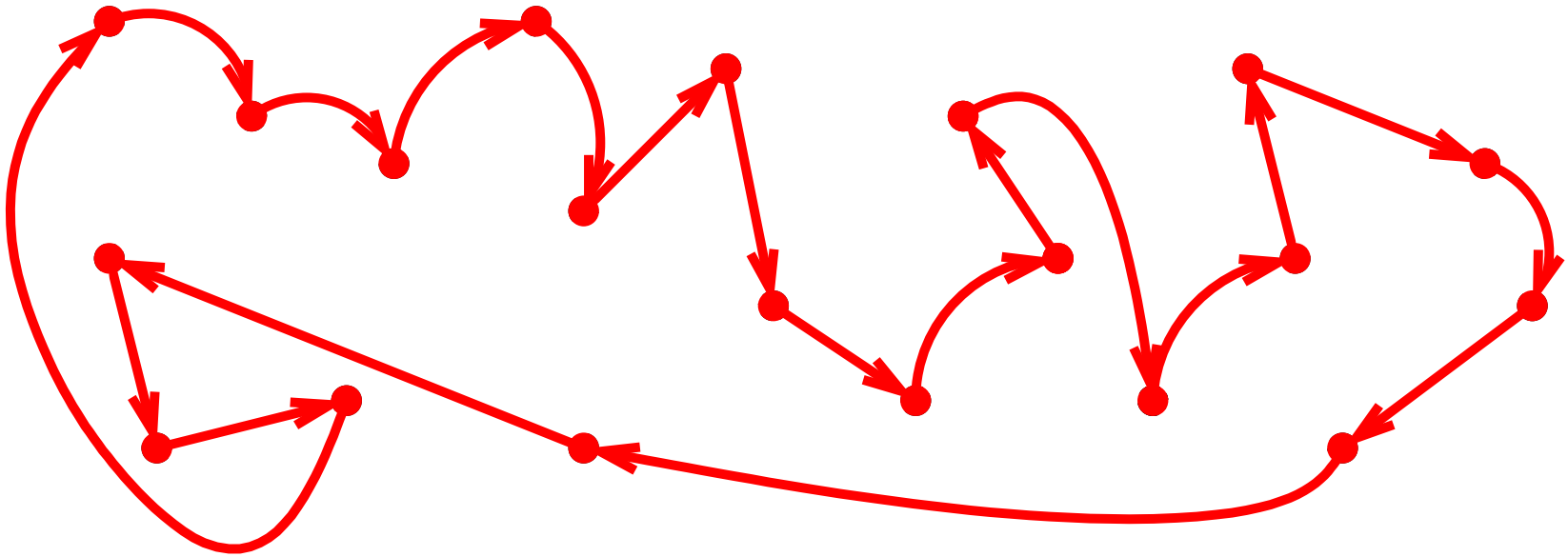


2-aproximação p/o TSP Métrico

(1) Árvore geradora de comprimento mínimo (polinomial)
(2) Dobre as arestas da árvore (todos os vértices tem grau par)
(3) Obtenha ciclo euleriano P (polinomial)

ciclo euleriano: passeio fechado que passa exatamente uma vez por cada aresta

(4) Obtenha de P circuito hamiltoniano C
(5) Devolva C



2-Aproximação p/o TSP Métrico

Algoritmo TSPM-Rosenkrantz-Stearn-Lewis (G, l)

$T \leftarrow \mathsf{MST}(G, l)$

$T' \leftarrow T + T$

$P \leftarrow \mathsf{EULER}(T')$

$C \leftarrow \mathsf{ATALHO}(P)$

devolve C

2-Aproximação p/o TSP Métrico

Algoritmo TSPM-Rosenkrantz-Stearn-Lewis (G, l)

$T \leftarrow \text{MST}(G, l)$

$T' \leftarrow T + T$

$P \leftarrow \text{EULER}(T')$

$C \leftarrow \text{ATALHO}(P)$

devolve C

Tempo de execução: $m \log n$

n: o número de vértices de G
m: o número de arestas de G

2-Aproximação p/o TSP Métrico

Delimitação inferior: $\mathrm{OPT} \geq l(T)$

onde T é árvore geradora de comprimento mínimo

2-Aproximação p/o TSP Métrico

Delimitação inferior: $\mathrm{OPT} \geq l(T)$

onde T é árvore geradora de comprimento mínimo

Prova:

C^*: circuito hamiltoniano de comprimento mínimo

e: aresta de C^*

$C^* - e$ é árvore geradora

2-Aproximação p/o TSP Métrico

Delimitação inferior: $\mathrm{OPT} \geq l(T)$

onde T é árvore geradora de comprimento mínimo

Prova:

C^*: circuito hamiltoniano de comprimento mínimo

e: aresta de C^*

$C^* - e$ é árvore geradora

$$\mathrm{OPT} = l(C^*) \geq l(C^* - e) \geq l(T). \quad \square$$

2-Aproximação p/o TSP Métrico

Delimitação inferior: $\mathrm{OPT} \geq l(T)$

onde T é árvore geradora de comprimento mínimo

Prova:

C^*: circuito hamiltoniano de comprimento mínimo

e: aresta de C^*

$C^* - e$ é árvore geradora

$$\mathrm{OPT} = l(C^*) \geq l(C^* - e) \geq l(T). \quad \square$$

Teorema: TSPM-Rosenkrantz-Stearn-Lewis é uma 2-aproximação polinomial para o TSP métrico.

2-Aproximação p/o TSP Métrico

Delimitação inferior: $\mathrm{OPT} \geq l(T)$

onde T é árvore geradora de comprimento mínimo

Prova:

C^*: circuito hamiltoniano de comprimento mínimo

e: aresta de C^*

$C^* - e$ é árvore geradora

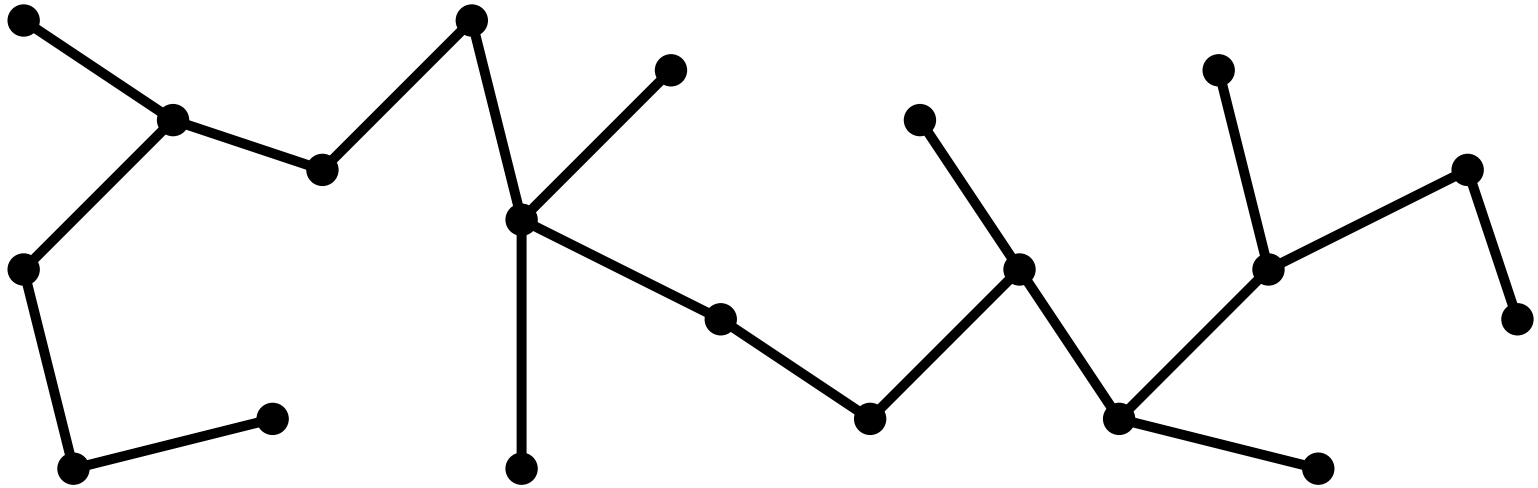
$$\mathrm{OPT} = l(C^*) \geq l(C^* - e) \geq l(T). \quad \square$$

Teorema: TSPM-Rosenkrantz-Stearn-Lewis é uma 2-aproximação polinomial para o TSP métrico.

Prova: $l(C) \leq l(P) = l(T') = 2\,l(T) \leq 2\,\mathrm{OPT}. \quad \square$

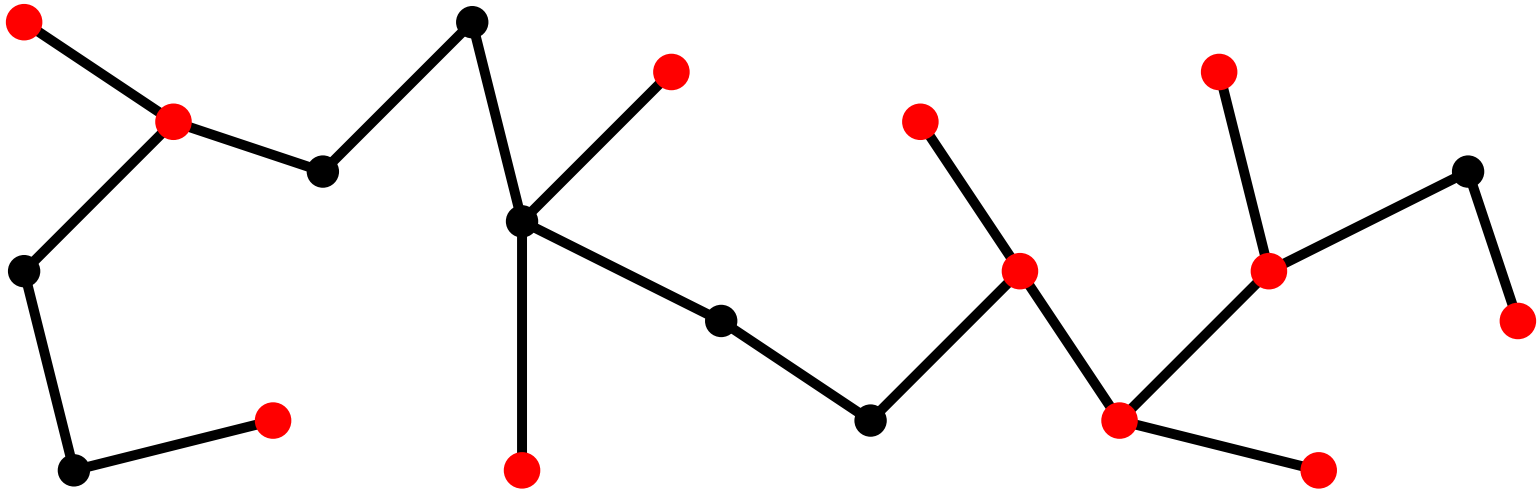
Algoritmo de Christofides

(1) Árvore geradora de comprimento mínimo



Algoritmo de Christofides

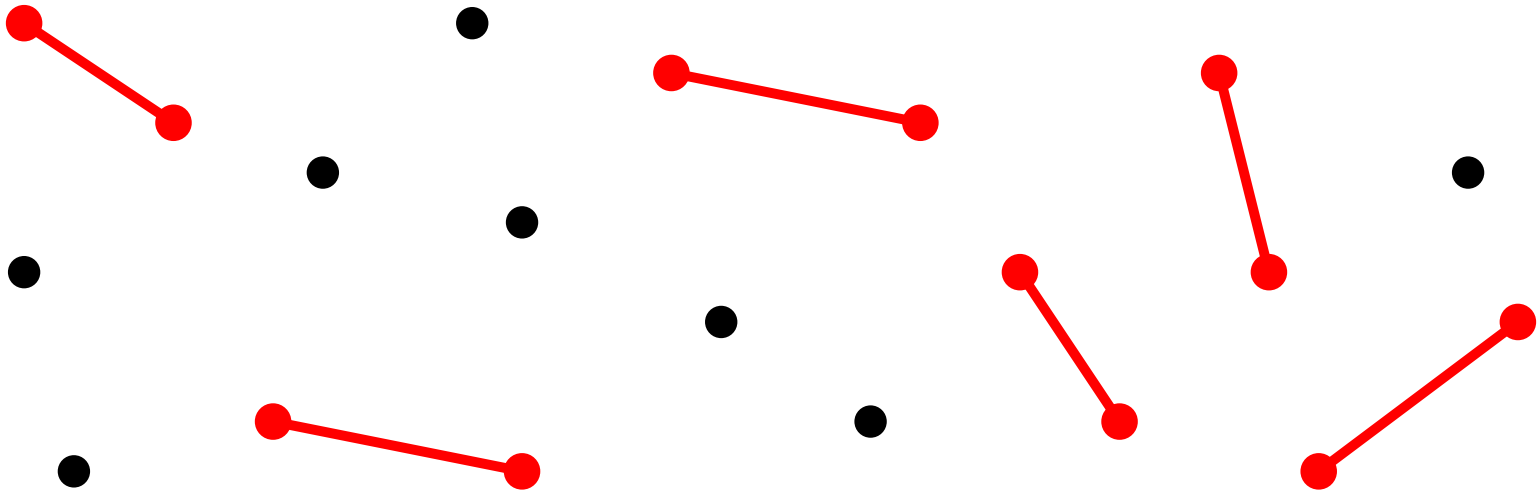
(1) Árvore geradora de comprimento mínimo



vértices de grau ímpar

Algoritmo de Christofides

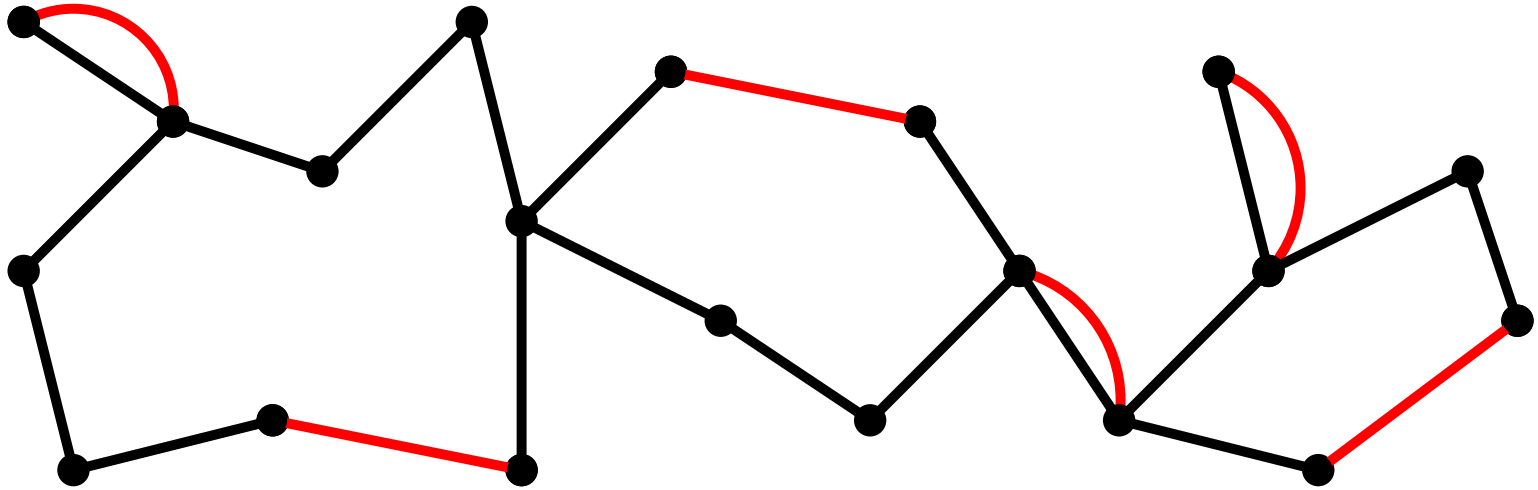
(1) Árvore geradora de comprimento mínimo



(2) Emparelhamento perfeito de comprimento mínimo no subgrafo induzido pelos vértices de grau ímpar (polinomial)

Algoritmo de Christofides

(1) Árvore geradora de comprimento mínimo

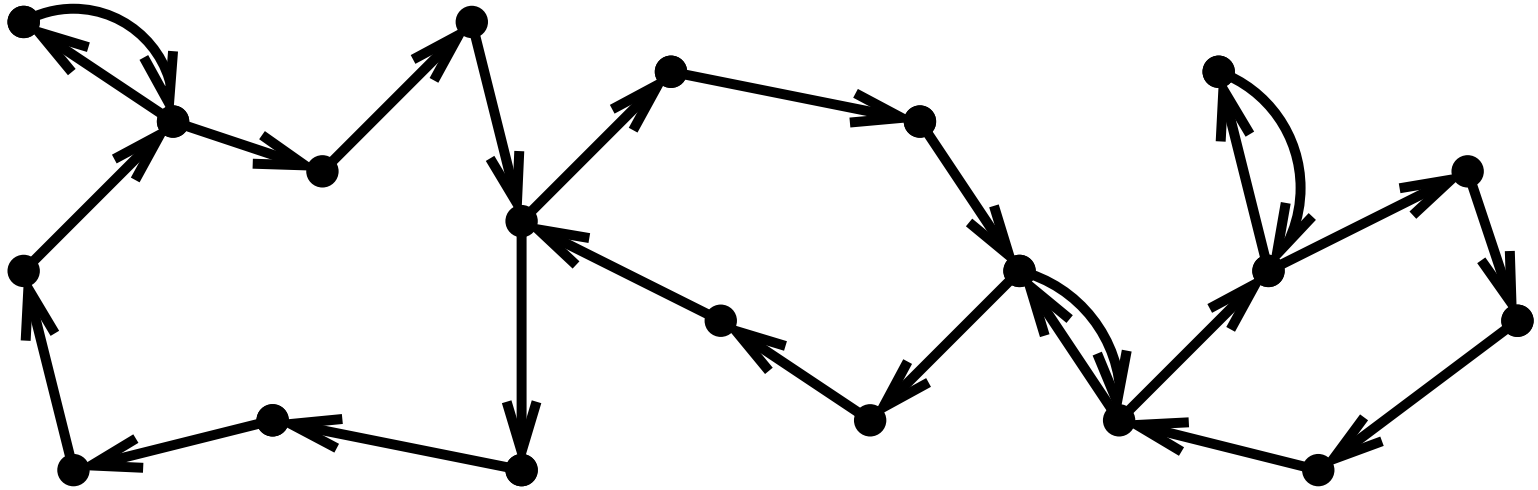


(2) Emparelhamento perfeito de comprimento mínimo no subgrafo induzido pelos vértices de grau ímpar (polinomial)

(3) Junte os dois obtendo T' euleriano

Algoritmo de Christofides

(1) Árvore geradora de comprimento mínimo



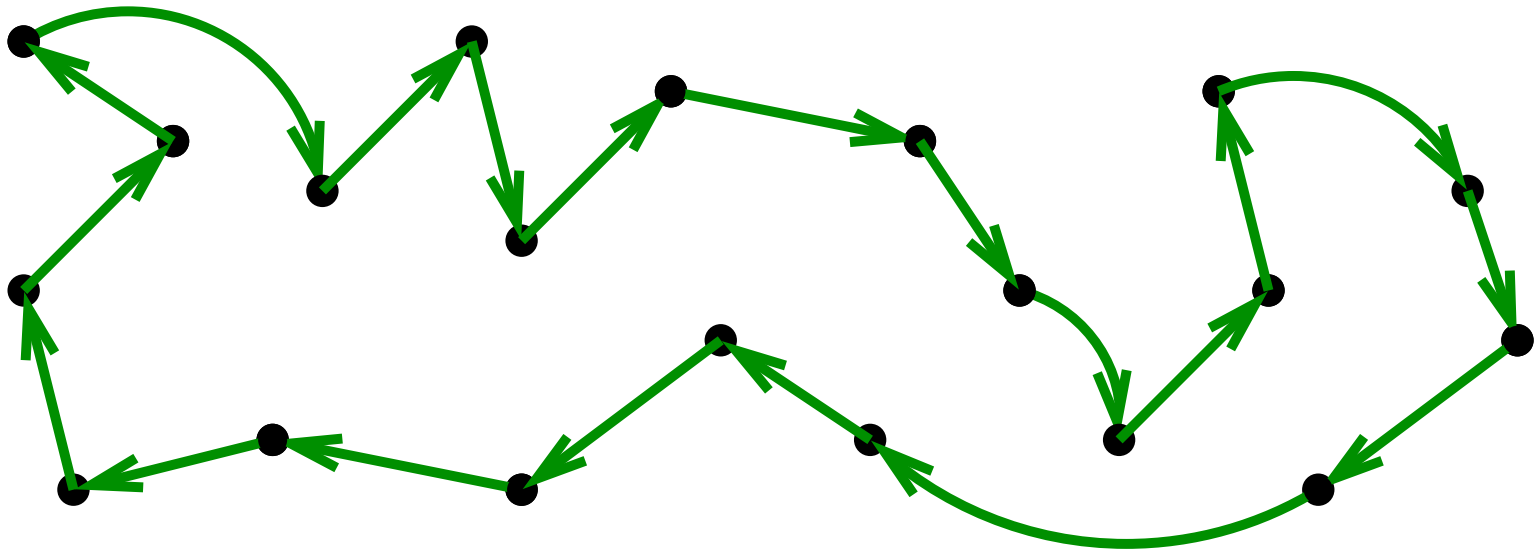
(2) Emparelhamento perfeito de comprimento mínimo no subgrafo induzido pelos vértices de grau ímpar (polinomial)

(3) Junte os dois obtendo T' euleriano

(4) Obtenha ciclo euleriano P de T'

Algoritmo de Christofides

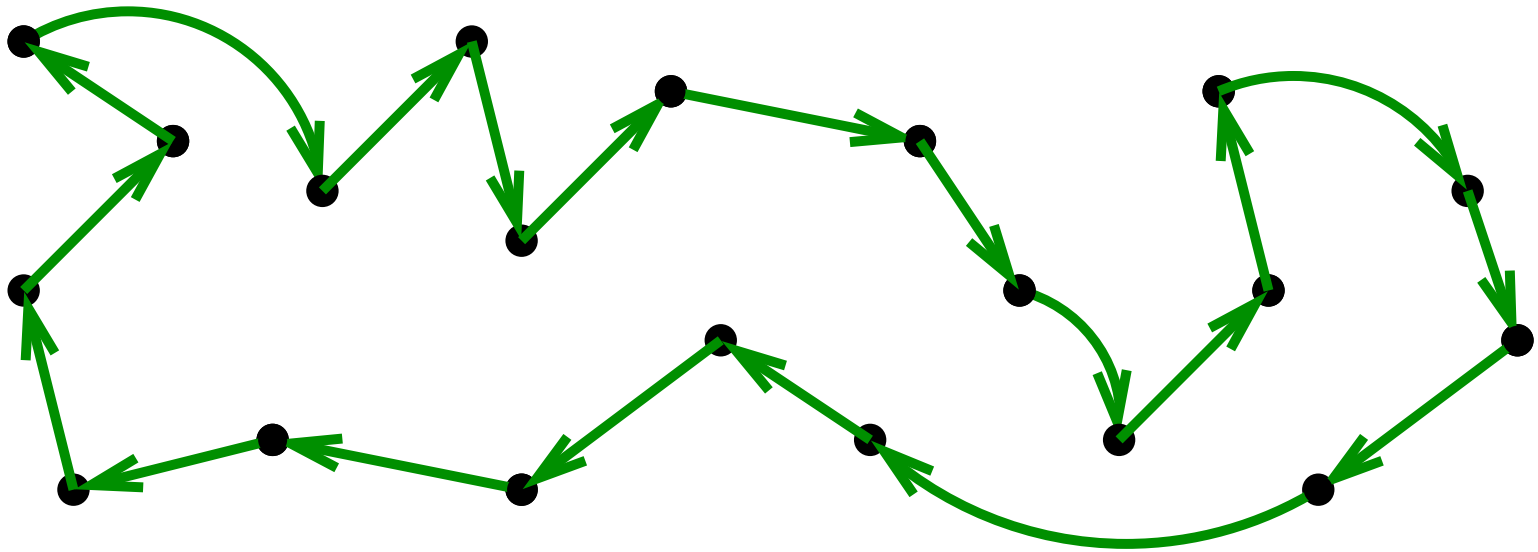
(1) Árvore geradora de comprimento mínimo



(2) Emparelhamento perfeito de comprimento mínimo no subgrafo induzido pelos vértices de grau ímpar (polinomial)
(3) Junte os dois obtendo T' euleriano
(4) Obtenha ciclo euleriano P de T'
(5) Obtenha de P circuito hamiltoniano C

Algoritmo de Christofides

(1) Árvore geradora de comprimento mínimo



(2) Emparelhamento perfeito de comprimento mínimo no subgrafo induzido pelos vértices de grau ímpar (polinomial)
(3) Junte os dois obtendo T' euleriano
(4) Obtenha ciclo euleriano P de T'
(5) Obtenha de P circuito hamiltoniano C
(6) Devolva C

Algoritmo de Christofides

Algoritmo TSPM-Christofides (G, l)

$T \leftarrow \text{MST}(G, l)$
$I \leftarrow$ conjunto de vértices de grau ímpar de T
$M \leftarrow \text{EDMONDS}(G[I], l)$
$T' \leftarrow T + M$
$P \leftarrow \text{EULER}(T')$
$C \leftarrow \text{ATALHO}(P)$
devolve C

($G[I]$: subgrafo de G induzido por I)

Algoritmo de Christofides

Algoritmo TSPM-Christofides (G, l)

$T \leftarrow \text{MST}(G, l)$
$I \leftarrow$ conjunto de vértices de grau ímpar de T
$M \leftarrow \text{EDMONDS}(G[I], l)$
$T' \leftarrow T + M$
$P \leftarrow \text{EULER}(T')$
$C \leftarrow \text{ATALHO}(P)$
devolve C

($G[I]$: subgrafo de G induzido por I)

Tempo de execução: n^3
(n: o número de vértices de G)

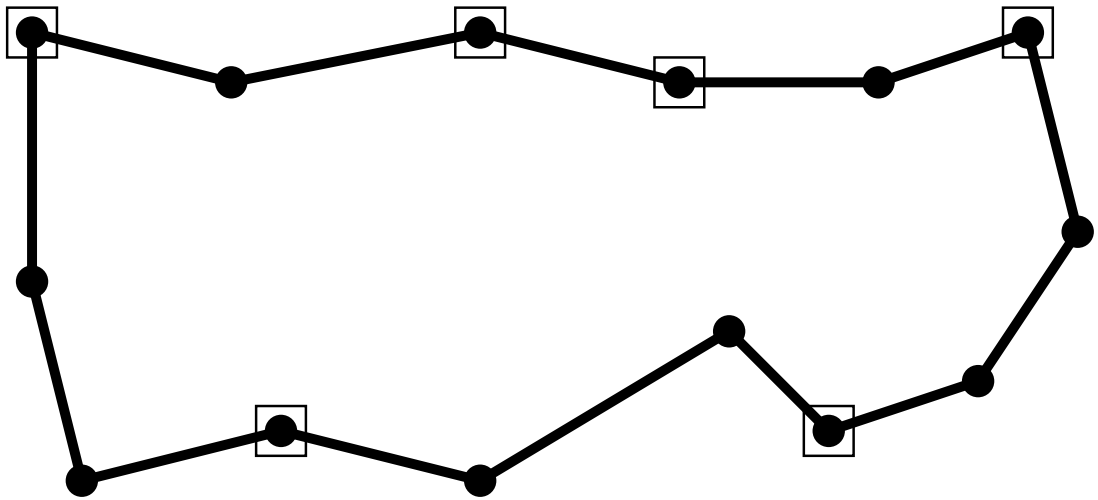
Algoritmo de Christofides

Uma segunda delimitação inferior: $\mathrm{OPT} \geq 2\,l(M)$
onde M é e. p. de comprimento mínimo em $G[I]$

Algoritmo de Christofides

Uma segunda delimitação inferior: $\mathrm{OPT} \geq 2\,l(M)$
onde M é e. p. de comprimento mínimo em $G[I]$

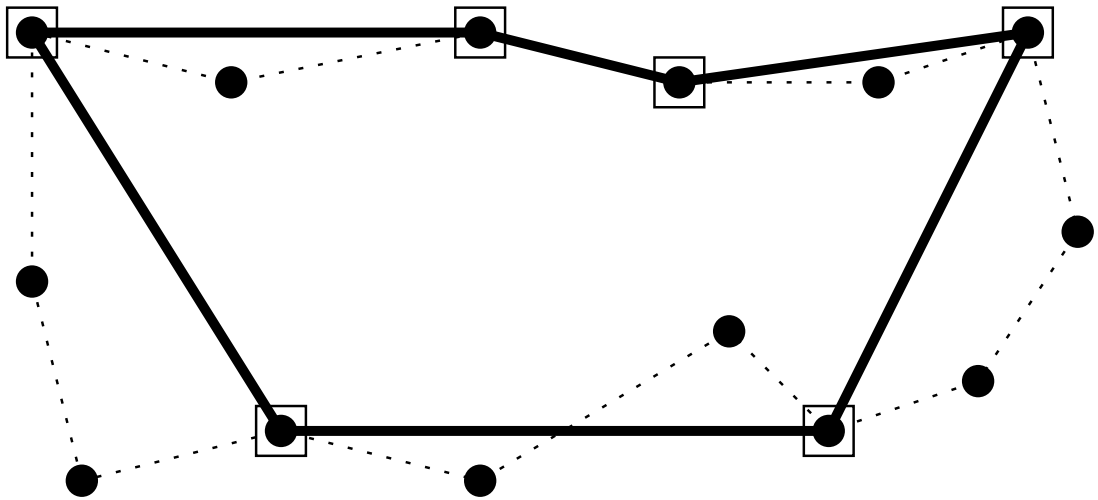
Pr: C^*: circuito hamiltoniano de comprimento mínimo
I: conj. vért. de grau ímpar de T ($|I|$ é par)



Algoritmo de Christofides

Uma segunda delimitação inferior: $\mathrm{OPT} \geq 2\,l(M)$
onde M é e. p. de comprimento mínimo em $G[I]$

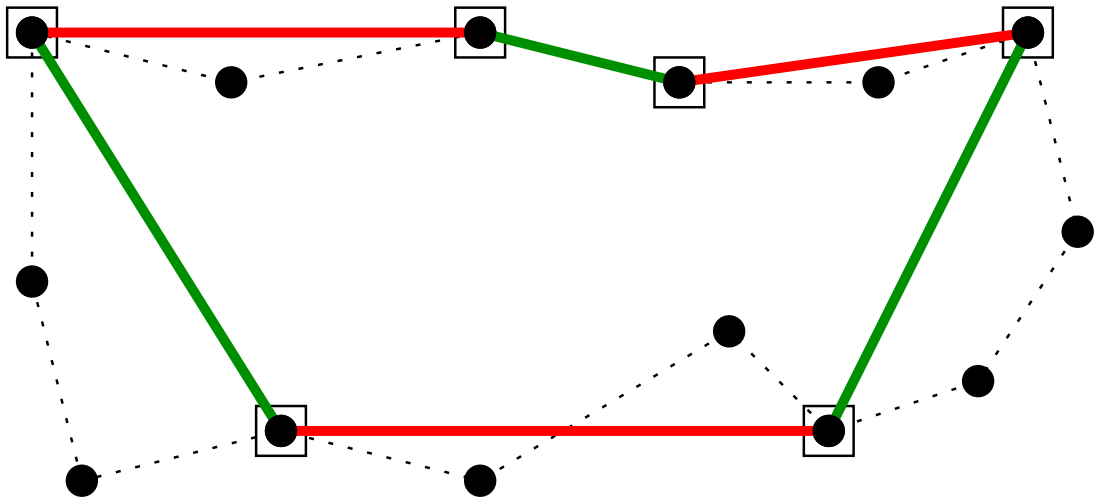
Pr: C^*: circuito hamiltoniano de comprimento mínimo
I: conj. vért. de grau ímpar de T ($|I|$ é par)
C': circuito induzido por C^* em I



Algoritmo de Christofides

Uma segunda delimitação inferior: $\mathrm{OPT} \geq 2\,l(M)$
onde M é e. p. de comprimento mínimo em $G[I]$

Pr: C^*: circuito hamiltoniano de comprimento mínimo
I: conj. vért. de grau ímpar de T ($|I|$ é par)
C': circuito induzido por C^* em I



C' determina dois e. p. em $G[I]$: M_1 e M_2

Algoritmo de Christofides

Uma segunda delimitação inferior: $\mathrm{OPT} \geq 2\,l(M)$
onde M é e. p. de comprimento mínimo em $G[I]$.

Prova:

$$\begin{aligned} 2\,l(M) &\leq l(M_1) + l(M_2) \\ &= l(C') \\ &\leq l(C^*) \qquad \text{(pela desigualdade triangular)} \\ &= \mathrm{OPT}\,. \end{aligned}$$

□

Algoritmo de Christofides

Teorema: TSPM-Christofides é uma $1{,}5$-aproximação polinomial para o TSP métrico.

Algoritmo de Christofides

Teorema: TSPM-Christofides é uma $1{,}5$-aproximação polinomial para o TSP métrico.

Prova:

$$\begin{aligned} l(C) &\leq l(P) \\ &= l(T') \\ &= l(T) + l(M) \\ &\leq \mathrm{OPT} + \frac{1}{2}\,\mathrm{OPT} \\ &= \frac{3}{2}\,\mathrm{OPT}\,. \end{aligned}$$

□

Algoritmo de Christofides

Teorema: TSPM-Christofides é uma $1{,}5$-aproximação polinomial para o TSP métrico.

Prova:

$$\begin{aligned} l(C) &\leq l(P) \\ &= l(T') \\ &= l(T) + l(M) \\ &\leq \mathrm{OPT} + \frac{1}{2}\,\mathrm{OPT} \\ &= \frac{3}{2}\,\mathrm{OPT}\,. \end{aligned}$$

□

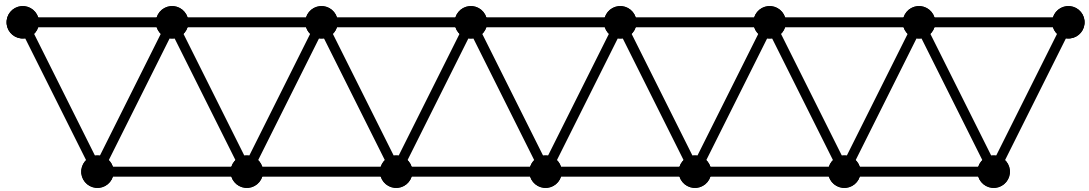
Melhor algoritmo de aproximação conhecido para o TSP métrico.

Algoritmo de Christofides

A análise é justa.

Algoritmo de Christofides

A análise é justa.

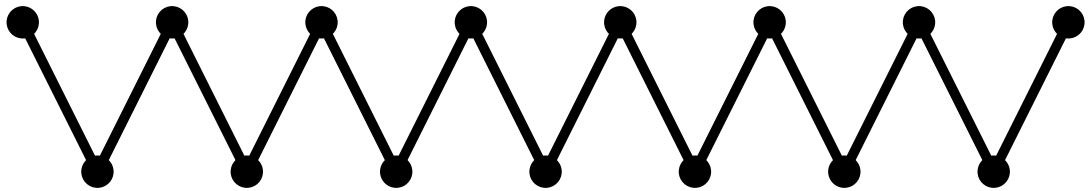


$2n + 1$ vértices

Algoritmo de Christofides

A análise é justa.

Christofides

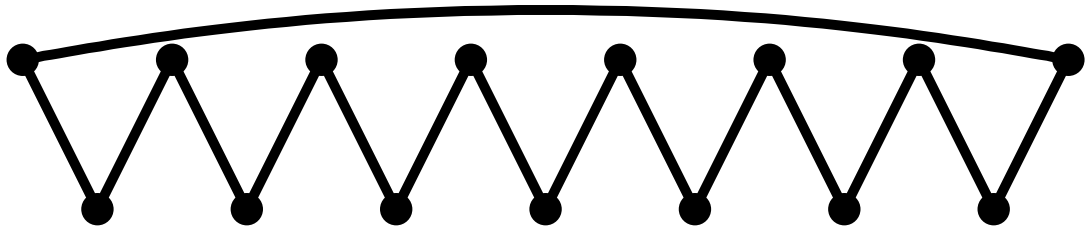


$2n + 1$ vértices

Algoritmo de Christofides

A análise é justa.

Christofides

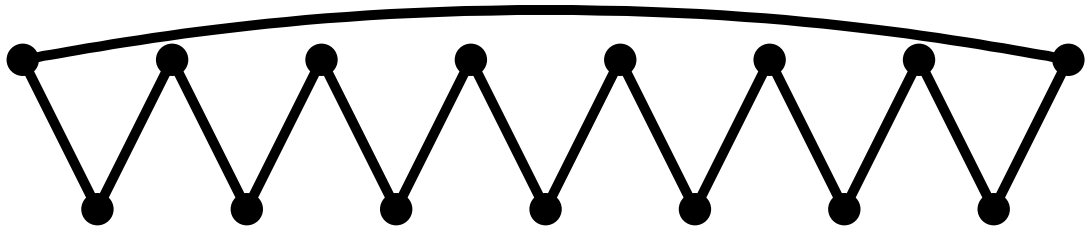


$2n + 1$ vértices
Comprimento: $3n$

Algoritmo de Christofides

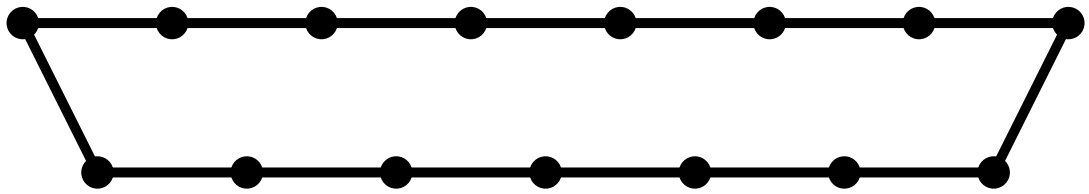
A análise é justa.

Christofides



$2n + 1$ vértices
Comprimento: $3n$

Circuito ótimo



Comprimento: $2n + 1$

TSP Euclideano

PTAS: esquema de aproximação polinomial

$(1+\epsilon)$-aproximação polinomial para todo $\epsilon > 0$ fixo

Arora'96 e Mitchel'96: PTAS para o TSP euclideano

Ideia:

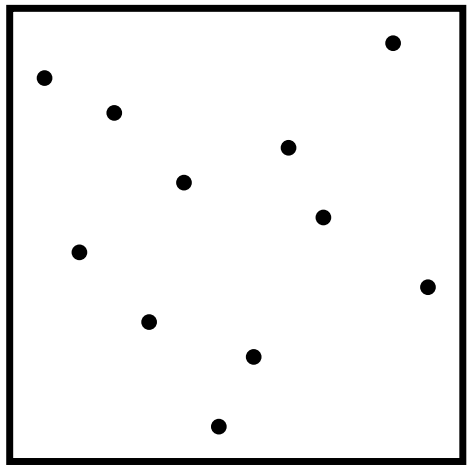
TSP Euclideano

PTAS: esquema de aproximação polinomial

$(1+\epsilon)$-aproximação polinomial para todo $\epsilon > 0$ fixo

Arora'96 e Mitchel'96: PTAS para o TSP euclideano

Ideia:



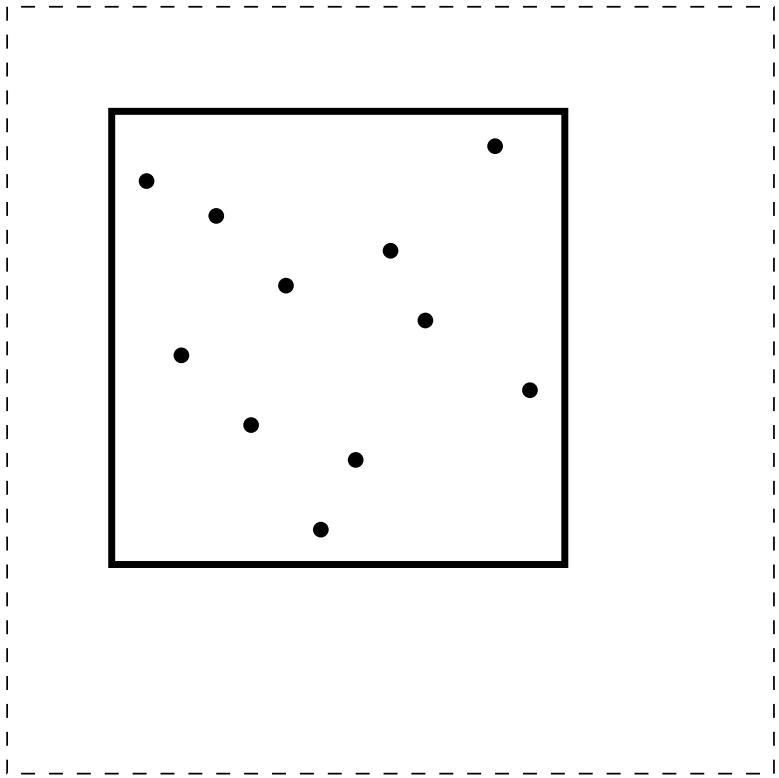
TSP Euclideano

PTAS: esquema de aproximação polinomial

$(1+\epsilon)$-aproximação polinomial para todo $\epsilon > 0$ fixo

Arora'96 e Mitchel'96: PTAS para o TSP euclideano

Ideia:



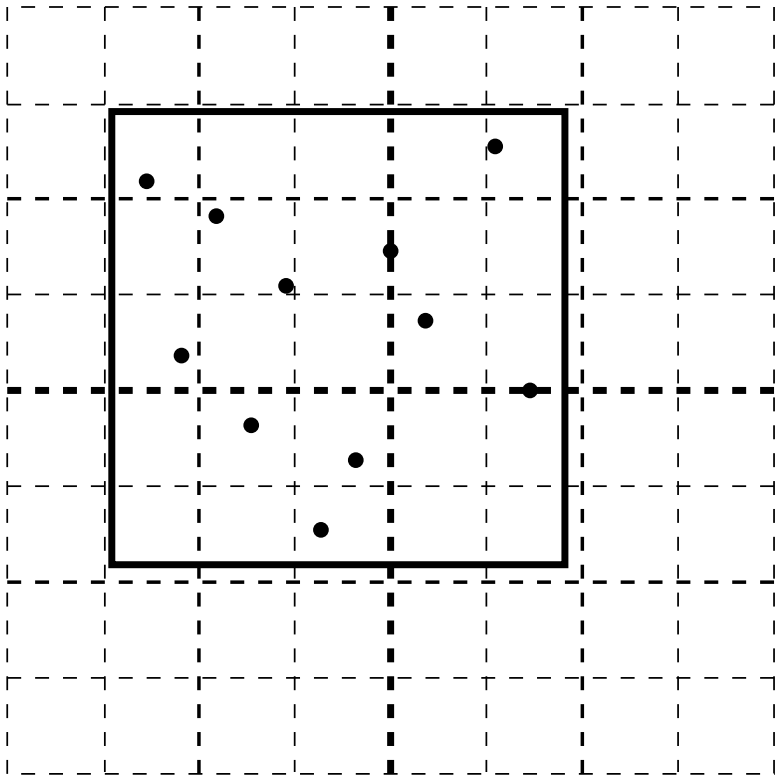
TSP Euclideano

PTAS: esquema de aproximação polinomial

$(1+\epsilon)$-aproximação polinomial para todo $\epsilon > 0$ fixo

Arora'96 e Mitchel'96: PTAS para o TSP euclideano

Ideia:



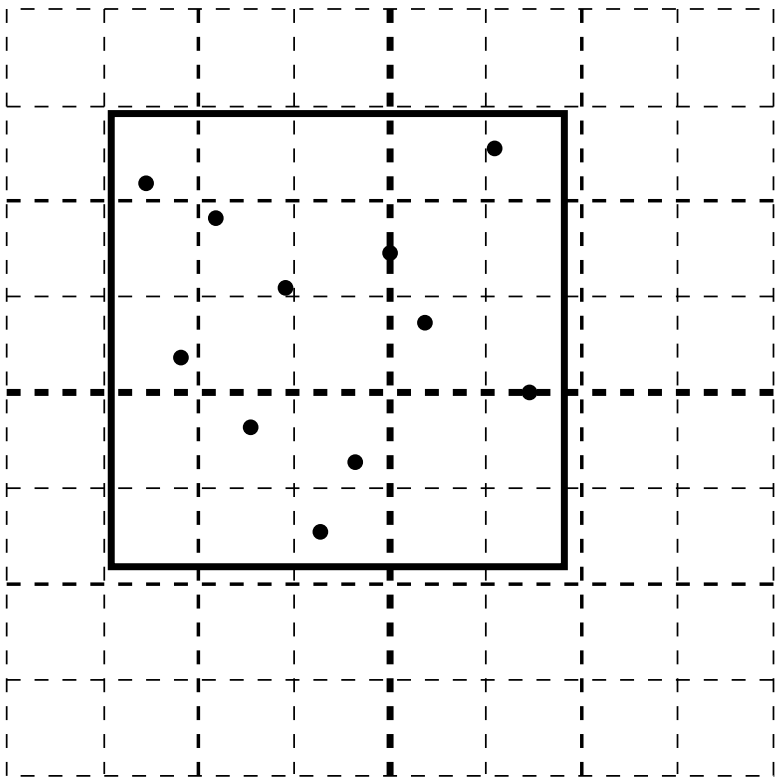
TSP Euclideano

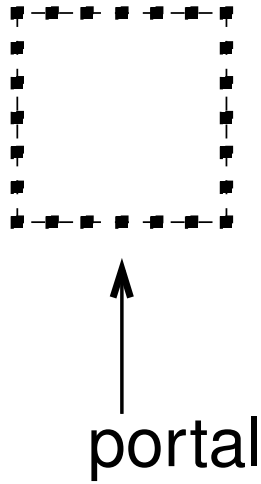
PTAS: esquema de aproximação polinomial

$(1+\epsilon)$-aproximação polinomial para todo $\epsilon > 0$ fixo

Arora'96 e Mitchel'96: PTAS para o TSP euclideano

Ideia:





TSP Euclideano

PTAS: esquema de aproximação polinomial

$(1+\epsilon)$-aproximação polinomial para todo $\epsilon > 0$ fixo

Arora'96 e Mitchel'96: PTAS para o TSP euclideano

Ideia:

Circuito que respeita portal: entra e sai dos quadrados da dissecção através de portais.

Algoritmo: Encontra um circuito hamiltoniano mais curto que respeita os portais por programação dinâmica.

Tal circuito está tão próximo quando se queira de um TSP tour.

Resultado de Inaproximabilidade

$\alpha : \mathbb{N} \to \mathbb{N}$ função polinomialmente computável

Teorema (Sahni e Gonzalez '76): Se existir $\alpha(n)$-aproximação polinomial p/o TSP(G, l), onde $n := |V_G|$, então P=NP.

Resultado de Inaproximabilidade

$\alpha : \mathbb{N} \to \mathbb{N}$ função polinomialmente computável

Teorema (Sahni e Gonzalez '76): Se existir $\alpha(n)$-aproximação polinomial p/o TSP(G, l), onde $n := |V_G|$, então P=NP.

Problema HC: Dado G, decidir se G tem ou não um circuito hamiltoniano.

- NP-completo [K72]

Resultado de Inaproximabilidade

Teorema (Sahni e Gonzalez '76): Se existir $\alpha(n)$-aproximação polinomial p/o TSP(G, l), onde $n := |V_G|$, então P=NP.

Pr:

$\alpha(n)$-aproximação polinomial p/o TSP

$\Downarrow$

algoritmo polinomial p/o HC

Resultado de Inaproximabilidade

Teorema (Sahni e Gonzalez '76): Se existir $\alpha(n)$-aproximação polinomial p/o TSP(G, l), onde $n := |V_G|$, então P=NP.

Pr:

$\alpha(n)$-aproximação polinomial p/o TSP $\leftarrow A$

$\Downarrow$

algoritmo polinomial p/o HC $\leftarrow B$

Resultado de Inaproximabilidade

Teorema (Sahni e Gonzalez '76): Se existir $\alpha(n)$-aproximação polinomial p/o TSP(G, l), onde $n := |V_G|$, então P=NP.

Pr:

$\alpha(n)$-aproximação polinomial p/o TSP $\leftarrow A$

$\Downarrow$

algoritmo polinomial p/o HC $\leftarrow B$

Algoritmo B (G)

$H \leftarrow$ grafo completo em V_G

para cada e em E_G faça $l_e \leftarrow 1$

para cada e em $E_H \setminus E_G$ faça $l_e \leftarrow \alpha(|V_G|)|V_G| + 1$

$C \leftarrow A(H, l)$

se $l(C) \leq \alpha(|V_G|)|V_G|$

então devolva "Sim"

senão devolva "Não"

Resultado de Inaproximabilidade

A polinomial $\Rightarrow$ B polinomial

Resultado de Inaproximabilidade

A polinomial $\Rightarrow$ B polinomial

se existe circuito hamiltoniano em G,
então $\mathrm{OPT} = |V_G|$ e

$$l(C) \leq \alpha(|V_G|)\,\mathrm{OPT} = \alpha(|V_G|)|V_G|.$$

Resultado de Inaproximabilidade

A polinomial $\Rightarrow$ B polinomial

se existe circuito hamiltoniano em G,
então $\mathrm{OPT} = |V_G|$ e

$$l(C) \leq \alpha(|V_G|)\,\mathrm{OPT} = \alpha(|V_G|)|V_G|.$$

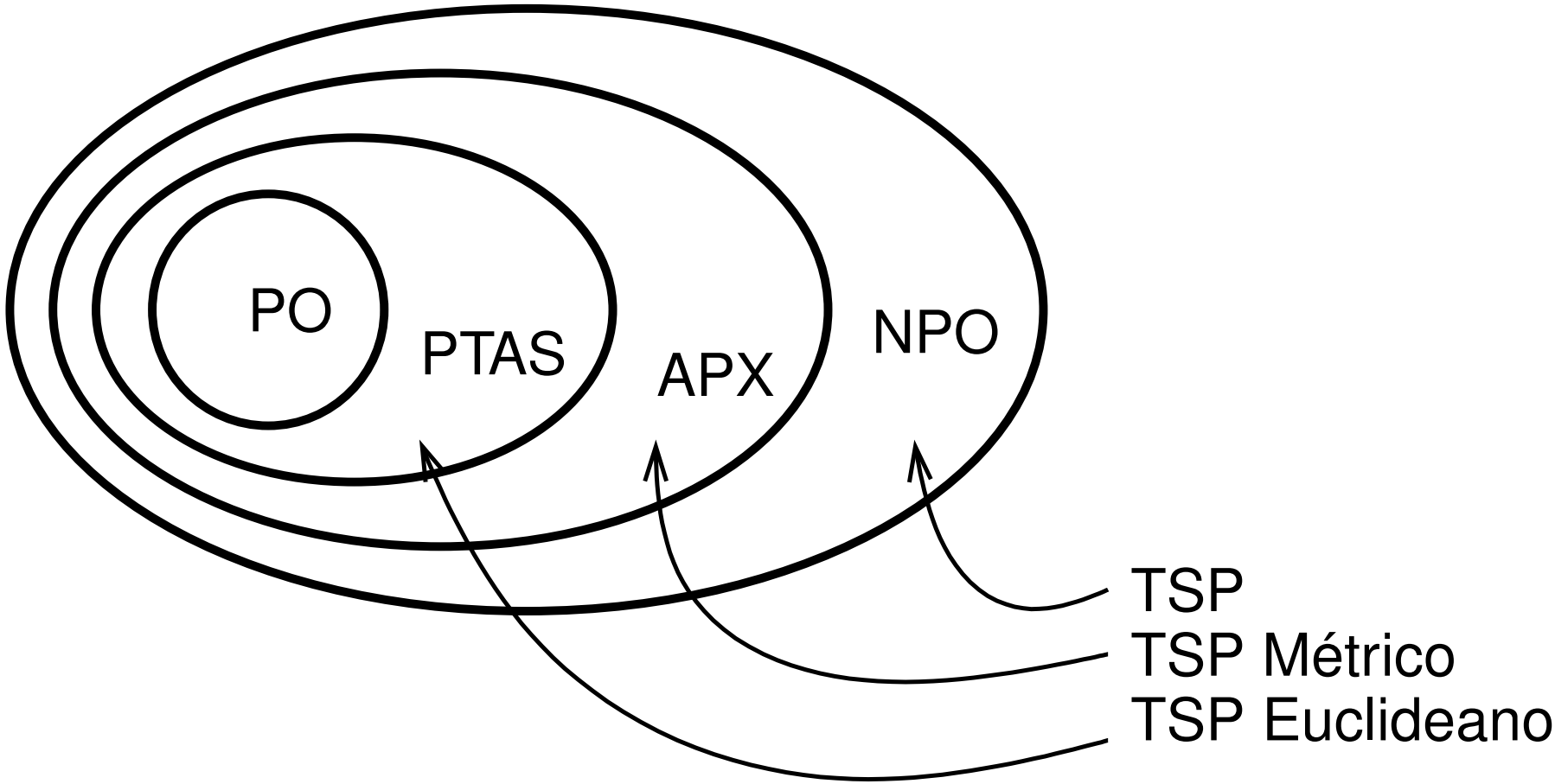
se não existe circuito hamiltoniano em G,
então

$$l(C) \geq \alpha(|V_G|)|V_G| + 1$$

pois qq circ. hamilt. usa $e \notin E_G$
(i.e., $l_e = \alpha(|V_G|)|V_G| + 1$).

□

Para completar...



Cobertura por conjuntos

Instância:

- conjunto base finito E
- coleção $\mathcal{S} = \{S_1, \ldots, S_m\}$ de subconjuntos de E
- custo $c_j > 0$ para cada S_j em $\mathcal{S}$

Cobertura por conjuntos

Instância:

- conjunto base finito E
- coleção $\mathcal{S} = \{S_1, \ldots, S_m\}$ de subconjuntos de E
- custo $c_j > 0$ para cada S_j em $\mathcal{S}$

Objetivo: encontrar cobertura $\mathcal{S}' \subseteq \mathcal{S}$ de E de custo mínimo.

Cobertura por conjuntos

Instância:

- conjunto base finito E
- coleção $\mathcal{S} = \{S_1, \ldots, S_m\}$ de subconjuntos de E
- custo $c_j > 0$ para cada S_j em $\mathcal{S}$

Objetivo: encontrar cobertura $\mathcal{S}' \subseteq \mathcal{S}$ de E de custo mínimo.

Relaxação linear: Dados E, $\mathcal{S}$ e c, encontrar x que

minimize $\sum_j c_j\, x_j$

sujeito a $\sum_{j:e \in S_j} x_j \geq 1$ para cada e em E

$x_j \geq 0$ para $j = 1, \ldots, m$

Cobertura por conjuntos

Instância:

- conjunto base finito E
- coleção $\mathcal{S} = \{S_1, \ldots, S_m\}$ de subconjuntos de E
- custo $c_j > 0$ para cada S_j em $\mathcal{S}$

Objetivo: encontrar cobertura $\mathcal{S}' \subseteq \mathcal{S}$ de E de custo mínimo.

Relaxação linear: Dados E, $\mathcal{S}$ e c, encontrar x que

minimize $\sum_j c_j\, x_j$

sujeito a $\sum_{j:e \in S_j} x_j \geq 1$ para cada e em E

$x_j \geq 0$ para $j = 1, \ldots, m$

Este é o programa primal, que denotaremos por (P).

Arredondamento determinístico

ARREDDET $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $x^* \leftarrow$ solução da relaxação linear (P)
2 **para** cada e em E **faça**
3 $\quad f_e \leftarrow |\{j : e \in S_j\}|$
4 $f \leftarrow \max\{f_e : e \in E\}$
5 $I \leftarrow \{j : x^*_j \geq 1/f\}$
6 **devolva** I

Arredondamento determinístico

ARREDDET $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $x^* \leftarrow$ solução da relaxação linear (P)
2 **para** cada e em E **faça**
3 $\quad f_e \leftarrow |\{j : e \in S_j\}|$
4 $f \leftarrow \max\{f_e : e \in E\}$
5 $I \leftarrow \{j : x^*_j \geq 1/f\}$
6 **devolva** I

Como (P) pode ser resolvido em tempo polinomial,
o consumo de tempo do algoritmo é polinomial.

Arredondamento determinístico

ARREDDET $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $x^* \leftarrow$ solução da relaxação linear (P)
2 **para** cada e em E **faça**
3 $\quad f_e \leftarrow |\{j : e \in S_j\}|$
4 $f \leftarrow \max\{f_e : e \in E\}$
5 $I \leftarrow \{j : x^*_j \geq 1/f\}$
6 **devolva** I

Como (P) pode ser resolvido em tempo polinomial, o consumo de tempo do algoritmo é polinomial.

Mas I é mesmo uma cobertura?

Arredondamento determinístico

ARREDDET $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $x^* \leftarrow$ solução da relaxação linear (P)
2 **para** cada e em E **faça**
3 $\quad f_e \leftarrow |\{j : e \in S_j\}|$
4 $f \leftarrow \max\{f_e : e \in E\}$
5 $I \leftarrow \{j : x^*_j \geq 1/f\}$
6 **devolva** I

Como (P) pode ser resolvido em tempo polinomial, o consumo de tempo do algoritmo é polinomial.

Mas I é mesmo uma cobertura?

Como $\sum_{j:e\in S_j} x^*_j \geq 1$ tem exatamente $f_e \leq f$ termos, um deles tem que valer pelo menos $1/f$.

Arredondamento determinístico

ARREDDET $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $x^* \leftarrow$ solução da relaxação linear (P)
2 **para** cada e em E **faça**
3 $\quad f_e \leftarrow |\{j : e \in S_j\}|$
4 $f \leftarrow \max\{f_e : e \in E\}$
5 $I \leftarrow \{j : x^*_j \geq 1/f\}$
6 **devolva** I

Lema: I é uma cobertura.

Prova: Como $\sum_{j:e\in S_j} x^*_j \geq 1$ tem exatamente $f_e \leq f$ termos, um deles tem que valer pelo menos $1/f$. ∎

Arredondamento determinístico

ARREDDET $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $x^* \leftarrow$ solução da relaxação linear (P)
2 **para** cada e em E **faça**
3 $\quad f_e \leftarrow |\{j : e \in S_j\}|$
4 $f \leftarrow \max\{f_e : e \in E\}$
5 $I \leftarrow \{j : x^*_j \geq 1/f\}$
6 **devolva** I

Lema: I é uma cobertura.

Prova: Como $\sum_{j:e\in S_j} x^*_j \geq 1$ tem exatamente $f_e \leq f$ termos, um deles tem que valer pelo menos $1/f$. ∎

Teorema: ARREDDET é uma f-aproximação.

Prova: $c(I) = \sum_{j\in I} c_j \leq \sum_j (c_j\, f\, x^*_j) = f\, z^*_{\mathrm{LP}} \leq f\, \mathrm{OPT}$. ∎

Arredondamento de solução dual

ARREDDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $y^* \leftarrow$ solução do dual da relaxação linear (P)

2 $I' \leftarrow \{j : \sum_{e \in S_j} y^*_e = c_j\}$

3 **devolva** I'

Arredondamento de solução dual

ARREDDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $y^* \leftarrow$ solução do dual da relaxação linear (P)

2 $I' \leftarrow \{j : \sum_{e \in S_j} y^*_e = c_j\}$

3 **devolva** I'

O consumo de tempo é polinomial.

Arredondamento de solução dual

ARREDDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $y^* \leftarrow$ solução do dual da relaxação linear (P)

2 $I' \leftarrow \{j : \sum_{e \in S_j} y^*_e = c_j\}$

3 **devolva** I'

O consumo de tempo é polinomial.

Lema: I' é uma cobertura.

Prova feita na aula.

Arredondamento de solução dual

ARREDDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $y^* \leftarrow$ solução do dual da relaxação linear (P)

2 $I' \leftarrow \{j : \sum_{e \in S_j} y^*_e = c_j\}$

3 **devolva** I'

O consumo de tempo é polinomial.

Lema: I' é uma cobertura.

Prova feita na aula.

Teorema: ARREDDUAL é uma f-aproximação.

Prova feita na aula.

Algoritmo primal-dual

PRIMALDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 **para** cada e em E **faça** $y_e \leftarrow 0$
2 $I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
3 **enquanto** existe e' que não é coberto por I **faça**
4 $\quad \epsilon \leftarrow \min\{c_j - \sum_{e \in S_j} y_e : j$ tal que $e' \in S_j\}$
5 $\quad y_{e'} \leftarrow y_{e'} + \epsilon$
6 $\quad I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
7 **devolva** I

Algoritmo primal-dual

PRIMALDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 **para** cada e em E **faça** $y_e \leftarrow 0$
2 $I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
3 **enquanto** existe e' que não é coberto por I **faça**
4 $\quad \epsilon \leftarrow \min\{c_j - \sum_{e \in S_j} y_e : j$ tal que $e' \in S_j\}$
5 $\quad y_{e'} \leftarrow y_{e'} + \epsilon$
6 $\quad I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
7 **devolva** I

Consumo de tempo polinomial, sem resolução de PL!

Algoritmo primal-dual

PRIMALDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 **para** cada e em E **faça** $y_e \leftarrow 0$
2 $I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
3 **enquanto** existe e' que não é coberto por I **faça**
4 $\quad \epsilon \leftarrow \min\{c_j - \sum_{e \in S_j} y_e : j$ tal que $e' \in S_j\}$
5 $\quad y_{e'} \leftarrow y_{e'} + \epsilon$
6 $\quad I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
7 **devolva** I

Consumo de tempo polinomial, sem resolução de PL!

Claro que I é uma cobertura.

Algoritmo primal-dual

PRIMALDUAL $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 **para** cada e em E **faça** $y_e \leftarrow 0$
2 $I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
3 **enquanto** existe e' que não é coberto por I **faça**
4 $\quad \epsilon \leftarrow \min\{c_j - \sum_{e \in S_j} y_e : j$ tal que $e' \in S_j\}$
5 $\quad y_{e'} \leftarrow y_{e'} + \epsilon$
6 $\quad I \leftarrow \{j : \sum_{e \in S_j} y_e = c_j\}$
7 **devolva** I

Consumo de tempo polinomial, sem resolução de PL!

Claro que I é uma cobertura.

Teorema: PRIMALDUAL é uma f-aproximação.

A mesma prova do ARREDDUAL funciona!

Algoritmo guloso

GULOSO $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $I \leftarrow \emptyset$
2 **para** $j \leftarrow 1$ **até** m **faça** $\hat{S}_j \leftarrow S_j$
3 **enquanto** I não é uma cobertura **faça**
4 $\quad k \leftarrow \arg\min\{\frac{c_j}{|\hat{S}_j|} : j$ é tal que $\hat{S}_j \neq \emptyset\}$
5 $\quad I \leftarrow I \cup \{k\}$
6 $\quad$ **para** $j \leftarrow 1$ **até** m **faça**
7 $\quad\quad \hat{S}_j \leftarrow \hat{S}_j \setminus S_k$
8 **devolva** I

Algoritmo guloso

GULOSO $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $I \leftarrow \emptyset$
2 **para** $j \leftarrow 1$ **até** m **faça** $\hat{S}_j \leftarrow S_j$
3 **enquanto** I não é uma cobertura **faça**
4 $\quad k \leftarrow \arg\min\{\frac{c_j}{|\hat{S}_j|} : j$ é tal que $\hat{S}_j \neq \emptyset\}$
5 $\quad I \leftarrow I \cup \{k\}$
6 $\quad$ **para** $j \leftarrow 1$ **até** m **faça**
7 $\quad\quad \hat{S}_j \leftarrow \hat{S}_j \setminus S_k$
8 **devolva** I

Consumo de tempo polinomial, sem resolução de PL!

Algoritmo guloso

GULOSO $(E, \mathcal{S}, c)$ $\quad \triangleright\ \mathcal{S} = \{S_1, \ldots, S_m\}$

1 $I \leftarrow \emptyset$
2 **para** $j \leftarrow 1$ **até** m **faça** $\hat{S}_j \leftarrow S_j$
3 **enquanto** I não é uma cobertura **faça**
4 $\quad k \leftarrow \arg\min\{\frac{c_j}{|\hat{S}_j|} : j$ é tal que $\hat{S}_j \neq \emptyset\}$
5 $\quad I \leftarrow I \cup \{k\}$
6 $\quad$ **para** $j \leftarrow 1$ **até** m **faça**
7 $\quad\quad \hat{S}_j \leftarrow \hat{S}_j \setminus S_k$
8 **devolva** I

Consumo de tempo polinomial, sem resolução de PL!

Claro que I é uma cobertura.

Algoritmo guloso

```
GULOSO (E, S, c)        ▷ S = {S_1, ..., S_m}
 1   I ← ∅
 2   para j ← 1 até m faça Ŝ_j ← S_j
 3   enquanto I não é uma cobertura faça
 4         k ← arg min{ c_j / |Ŝ_j| : j é tal que Ŝ_j ≠ ∅ }
 5         I ← I ∪ {k}
 6         para j ← 1 até m faça
 7               Ŝ_j ← Ŝ_j \ S_k
 8   devolva I
```

Consumo de tempo polinomial, sem resolução de PL!

Claro que I é uma cobertura.

Teorema: GULOSO é uma H_n-aproximação, onde $n = |E|$.