Optimalisering / BK1, college 12

Note Title 13-11-2010

Approximatiealgoritmen

Als beslissingsversie Π van optimaliseringsprobleem $\mathcal{P}$ NP-volledig is $\Rightarrow$

we kunnen niet verwachten dat $\mathcal{P}$ oplosbaar is in polyn. tijd.

Welke mogelijkheden hebben we?

1. Optimaal oplossen: bv. branch-and-bound als $\mathcal{P}$ een IP-probleem is. Kan traag zijn, maar geeft wel optimum.

2. Benaderen:
   a) Polynomiaal met kwaliteitsgarantie: "Approximatiealgoritmen"

   b) Anders, bv. "local search", heuristieken

Definitie: Een algoritme A is een ρ-approximatiealgoritme voor probleem $\mathcal{P}$ als A voor iedere toegelaten instantie I van $\mathcal{P}$ een oplossing produceert, in polynomiale tijd, van waarde $z_A(I)$ zdd

$$z_A(I) \leq \rho \cdot z_{OPT}(I) \quad \text{(min-proleem)}$$

of $$z_A(I) \geq \rho \cdot z_{OPT} \quad \text{(max-probleem)}$$

NB! voor minprobleem: $\rho \geq 1$
Als $\rho = 1 \Rightarrow$ optimaal oplossen in polynomiale tijd $\Rightarrow$ probleem $\mathcal{P} \in P$

voor maxprobleem: $\rho \leq 1$

ρ wordt prestatiegarantie genoemd.
hoeft niet een konstante te zijn.

Vertex cover

Gegeven een ongerichte graaf $G = (V, E)$ met kosten c_v op de punten $v \in V$.
Een vertex cover is een deelverz. $C \subseteq V$ van de punten zdd voor elke edge $\{u, v\}$ geldt dat $C \cap \{u, v\} \neq \emptyset$. Bepaal een vertex cover met minimale kosten.

Greedy alg. voor vertex cover

stel $c_v = 1 \ \forall \ v \in V$

Input: $G = (V, E)$
Output: Toegel. $C \subseteq V$
$V' := V$, $E' := E$, $C := \emptyset$
while $E' \neq \emptyset$
kies een punt $v \in V'$ met de grootste graad mbt E' (kies willekeurig bij gelijke graad)
$V' := V' \setminus \{v\}$

$C := C \cup \{v\}$
$E' := E' \setminus \{$alle kanten die aan v grenzen$\}$
end while

Hoe goed/slecht is de output C van greedy?

Neem de volgende bipartite graaf:

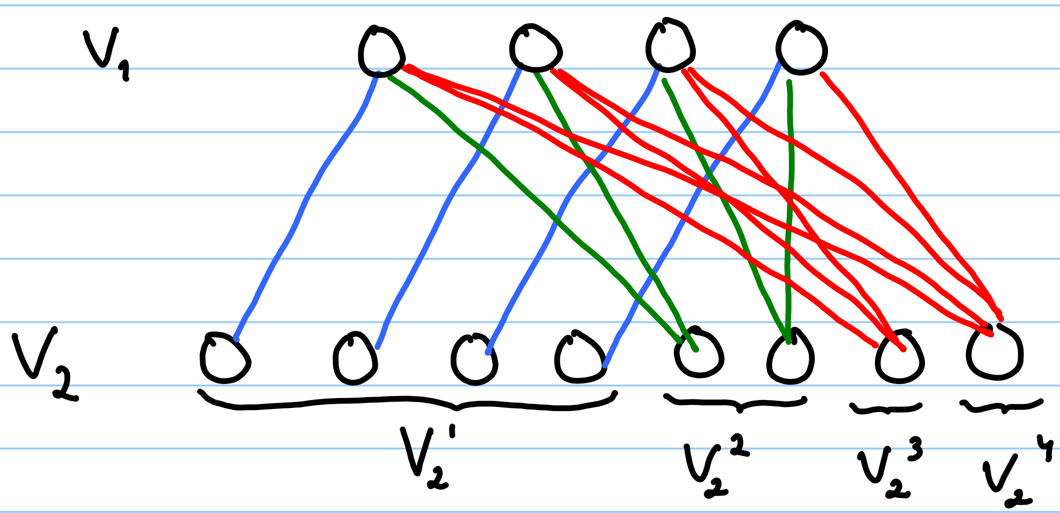
$G = ((V_1, V_2), E)$ met V_1, V_2 zdd

$|V_1| = n$

$V_2 = V_2^1 \cup V_2^2 \cup \dots \cup V_2^n$

$|V_2^1| = n$, $|V_2^2| = \lfloor n/2 \rfloor$, $|V_2^3| = \lfloor n/3 \rfloor$, -----

--- $|V_2^n| = n/n = 1$



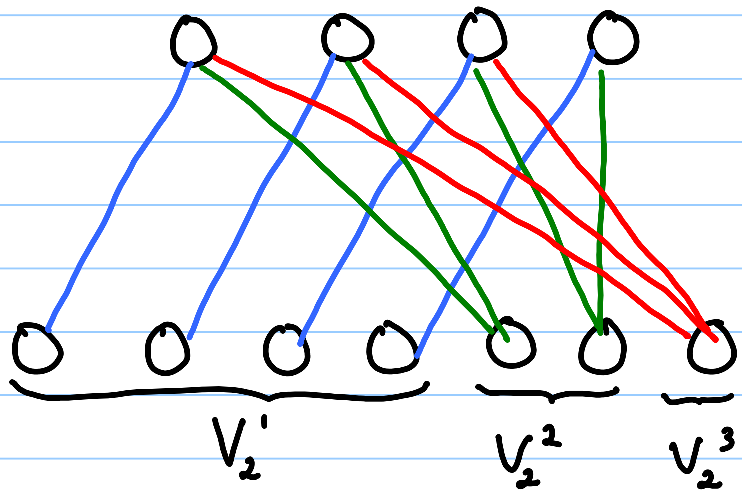
Elk punt in V_2^i is verbonden met i punten in V_1.

$$\Rightarrow d(v) \leq n \ \forall \ v \in V_1, \quad d(v) \leq i \ \forall \ v \in V_2^i$$

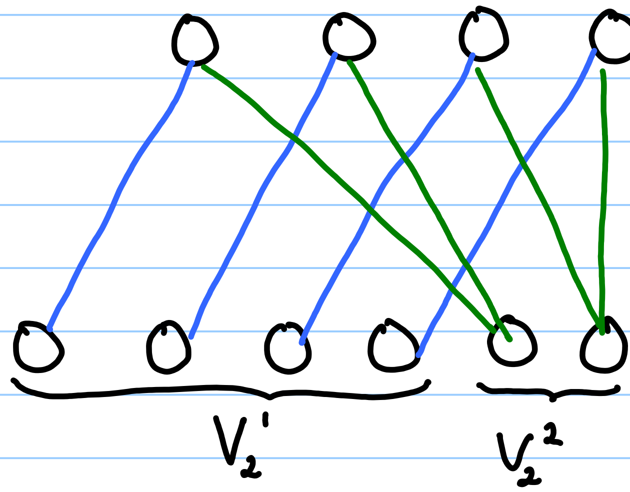
Het algoritme kan steeds een punt in V_2 kiezen als er een "willekeurige keuze" kan worden gemaakt.
Hier:

Kies eerst V_2^4:

$\Rightarrow$



daarna, V_2^3



daarna de punten in V_2^2 en daarna de punten in V_2^1.

Nadat alle punten in V_2 in de cover C zitten, zijn alle kanten gecovered, en het algoritme stopt, dus $C = V_2$. Hoeveel punten zijn er in V_2?

$$|V_2| = n + \lfloor n/2 \rfloor + \lfloor n/3 \rfloor + \dots + n/n$$
$$= O(n \cdot \ln(n))$$

Hoeveel punten hebben we nodig om alle kanten te overdekken? n, neem alle punten in V_1.

$\Rightarrow$ voor deze klasse instanties $\bar{I}$ is

$$\frac{z_A(\bar{I})}{z_{OPT}(\bar{I})} \sim \frac{n \cdot \ln n}{n} = \ln n$$

$\Rightarrow \quad \rho \geq \ln n$

Neem nu de IP-formulering van vertex cover:

Laat $x_v = \begin{cases} 1 & \text{als } v \in C \\ 0 & \text{anders} \end{cases}$

$$z_{IP} = \min \sum_{v \in V} c_v x_v$$
$$\text{s.t} \quad x_u + x_v \geq 1, \quad \forall \; e = \{u, v\} \in E$$
$$x_v \in \{0, 1\}, \quad \forall \; v \in V$$

LP-relaxatie:

$$z_{LP} = \min \sum_{v \in V} c_v x_v$$

$$\begin{aligned} \text{s.t} \quad & x_u + x_v \geq 1, && \forall \; e = \{u,v\} \in E \\ & x_v \geq 0, && \forall \; v \in V \\ & x_v \leq 1, && \forall \; v \in V \end{aligned}$$

hoeft niet als $c_v > 0$

Een "afrond"-algoritme:

1. Los LP-relaxatie op $\Rightarrow x^*, z_{LP}$
   polyn. tijd

2. Laat $C := \{v \in V : x_v^* \geq 1/2\}$
   polyn. tijd

<u>Stelling</u> Het "afrond-algoritme" is een 2-approximatiealgoritme.

<u>Bewijs</u>

a) Polyn. rekentijd. ok

b) Is de verkregen verz. C een toegel. vertex cover?
Ja, want tenminste één van de var. x_u, x_v heeft waarde $\geq 1/2$ in x^*
$\Rightarrow$ tenminste een van de punten in de kant $\{u,v\}$ zit in C, en dat geldt

voor iedere $\{u,v\} \in E$.

c) Bepaal ρ_A:

$$z(C) = \sum_{\{v : x_v^* \geq \frac{1}{2}\}} c_v \leq 2 \sum_{v \in V} c_v x_v^* = 2 z_{LP}$$

in de "worst case" is $x_v^* = \frac{1}{2}$ $\forall v \in V$
en ronden we alle van. naar boven af.

$$\Rightarrow z(C) \leq 2 \cdot z_{LP} \leq 2 z_{IP} \Rightarrow \frac{z(C)}{z_{IP}} \leq 2 \quad \blacksquare$$

Definitie TSP: Gegeven gehele getallen L en n, en een $n \times n$ symmetrische matrix $[d_{ij}]$, bepaal of er een cyclische permutatie ("tour") τ bestaat zodanig dat

$$\sum_{j=1}^{n} d_{i\tau(j)} \leq L$$

De tour is een kring in de volledige graaf met n punten waarbij d_{ij} de lengte is van kant $\{i,j\}$. In de kring komt elk punt in de graaf precies een keer voor.

Stelling: Er bestaat geen ε-approximatie-algoritme voor TSP voor enige $\varepsilon > 1$ tenzij $P = NP$.

Bewijs: Zie werkcollege.

Vraag: Kunnen we beter doen voor speciale gevallen van TSP? JA!

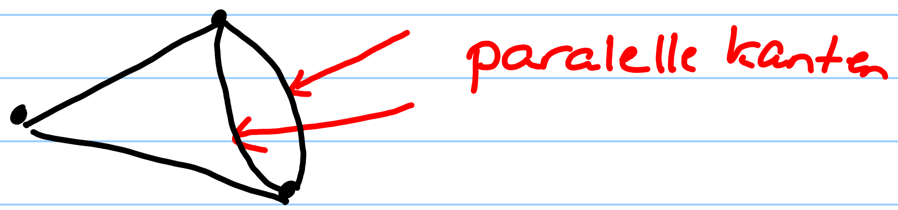
Def. Een afstandenmatrix $[d_{ij}]$ voldoet aan de driehoeksongelijkheid als

$$d_{ik} \leq d_{ij} + d_{jk} \quad \forall\ 1 \leq i,j,k \leq n.$$

Def. De driehoeksongelijkeidsversie van TSP (ΔTSP) is het TSP beperkt tot afstandenmatrices die voldoen aan de driehoeksongelijkheid.

Stelling: ΔTSP is NP-volledig.

Def: Een multigraaf is een graaf waar repetities van kanten, oftewel paralelle kanten zijn toegestaan.



Def. Een multigraaf is Euleriaans als er een gesloten wandeling bestaat in de graaf waarbij elk punt ten minste één keer voorkomt, en elke kant precies een keer voorkomt.

Stelling: Een multigraaf $G=(V,E)$ is Euleriaans dan en slechts dan als

(i) G samenhangend is

(ii) de graad van elk punt $v \in V$ even is.

Beschouw het volgende "boomalgoritme" voor ΔTSP:

Input: $n \times n$ matrix $[d_{ij}]$, symmetrisch Δ-ongelijkheid.
$G=(V,E)$, $|V|=n$, lengte van kant $e=\{v_i,v_j\}$ is d_{ij}, $d_{ii}=0$

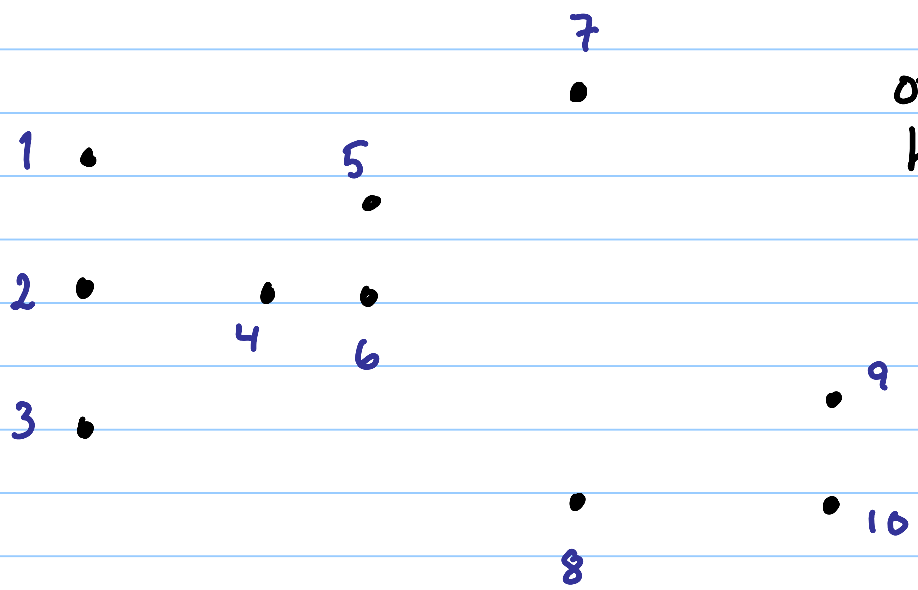
1. Bepaal de kortste opspannende boom $G_T=(V,T)$ in G. ($O(|V|^2)$ tijd met alg. van Prim, zie college 8.)

2. Verdubbel de kanten in T $\Rightarrow$ multigraaf waarbij elk punt even graad heeft, samenhangend.

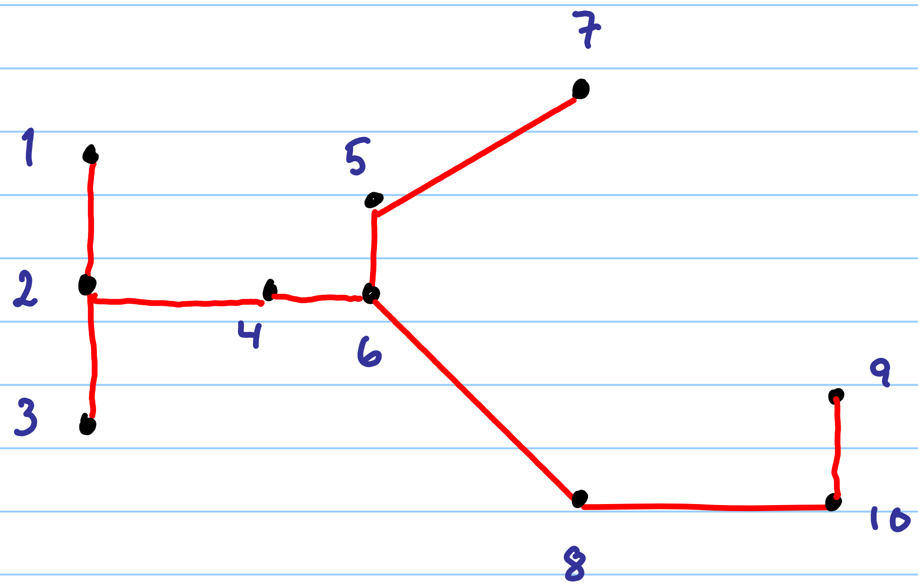
3. Bepaal een Euleriaanse wandeling in de multigraaf. Volg deze wandeling, maar sla punten over als ze al bezocht zijn. ($O(|E|)$ tijd.)

Voorbeeld:

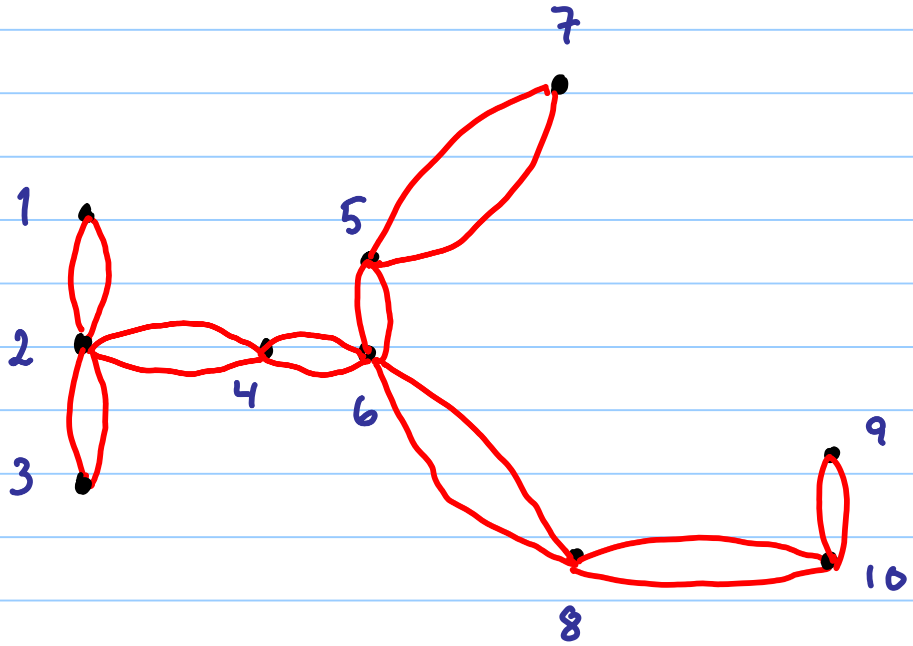
(De kanten zijn voor de overzichtelijkheid weggelaten)



Stap 1: Bepaal kortste opspannende boom in de graaf:



Stap 2: Verdubbel de kanten



Stap 3: Bepaal een Euleriaanse wandeling en een bijbehorende TSP-tour.

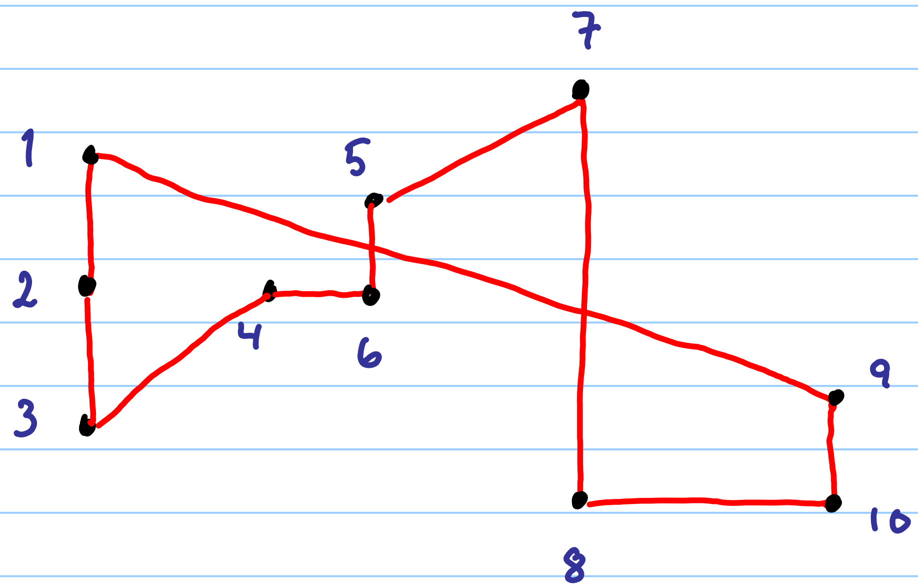
Euleriaanse wandeling:

$\{v_1, v_2, v_3, v_2, v_4, v_6, v_5, v_7, v_5, v_6, v_8, v_{10}, v_9, v_{10}, v_8, v_6, v_4, v_2, v_1\}$

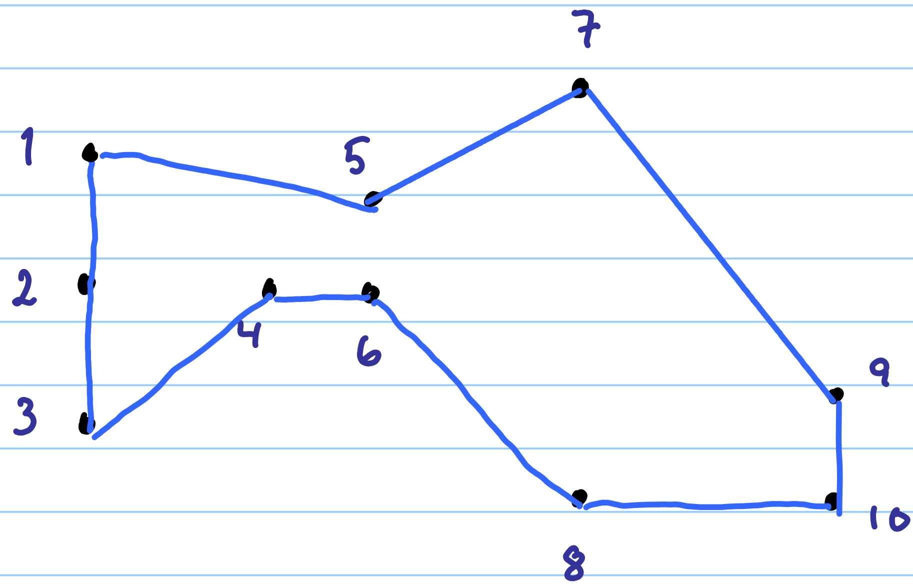
Bijbehorende route: Neem elk punt precies een keer (bv. de eerste keer dat hij in de wandeling voorkomt) en ga terug naar het begin.

$\{v_1, v_2, v_3, v_2, v_4, v_6, v_5, v_7, v_5, v_6, v_8, v_{10}, v_9, v_{10}, v_8, v_6, v_4, v_2, v_1\}$

Verkregen route:

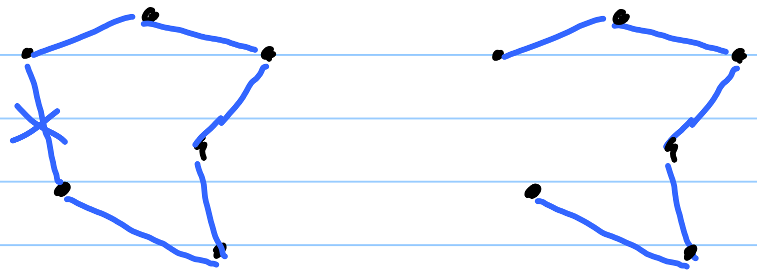


Optimale route:



Stelling: Het "boomalgoritme" is een 2-approximatiealgoritme voor ΔTSP.

Bewijs: Een TSP-tour met één kant weggelaten is een opspannende boom.



Dit geldt ook voor de optimale tour.

$\Rightarrow$ $\underbrace{\text{lengte v. kortste opspannende boom}}_{z_{MST}} \leq$

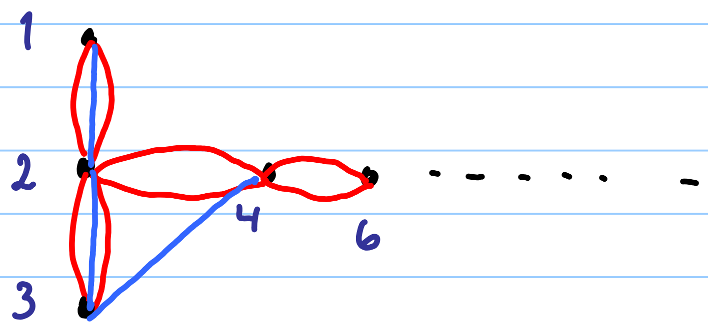
$\underbrace{\text{lengte optimale TSP-tour.}}_{z_{TSP}}$

Kanten van MST verdubbelen :

lengte v/d Euleriaanse wandeling $= 2 \cdot z_{MST}$

$$2 \cdot z_{MST} \leq 2 \cdot z_{TSP}$$

Vanwege de Δ-ongelijkheid is de lengte van de TSP-tour verkregen uit de Euleriaanse wandeling, z_{TSPE}, nooit langer dan deze wandeling, want we slaan punten over.



$d_{34} \leq d_{32} + d_{24}$

We kunnen daarom concluderen:

$$z_{TSPE} \leq 2 \cdot z_{MST} - 2 \cdot z_{TSP}$$

$$\Rightarrow \frac{z_{TSPE}}{z_{TSP}} \leq 2$$

Het algoritme is polynomiaal:
Uitrekenen kortste opspannende boom kan in $O(|V|^2)$ tijd.
Uitrekenen Euleriaanse wandeling kan in $O(|E|)$ tijd.

$\blacksquare$