# VI. Approximation Algorithms: Travelling Salesman Problem

Thomas Sauerwald

Easter 2017

UNIVERSITY OF CAMBRIDGE

# Outline

Introduction

General TSP

Metric TSP

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*

## The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*

Formal Definition

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*

Formal Definition

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$

# The Traveling Salesman Problem (TSP)

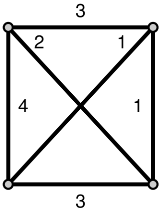
*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*

Formal Definition

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




**Formal Definition**

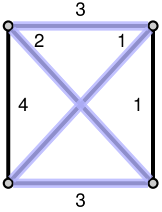
- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*

**Formal Definition**

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.



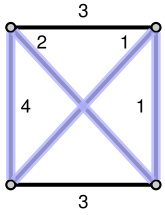

$3 + 2 + 1 + 3 = 9$

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*

**Formal Definition**

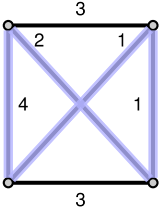
- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.




$2 + 4 + 1 + 1 = 8$

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




$2 + 4 + 1 + 1 = 8$

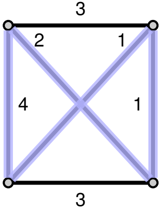
**Formal Definition**

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

Solution space consists of at most $n!$ possible tours!

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




$2 + 4 + 1 + 1 = 8$

Formal Definition

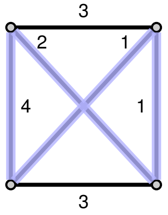
- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

Solution space consists of at most $n!$ possible tours!

Actually the right number is $(n-1)!/2$

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




$2 + 4 + 1 + 1 = 8$

Formal Definition

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

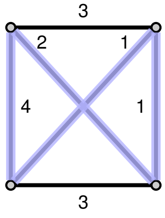
Solution space consists of at most $n!$ possible tours!

Actually the right number is $(n - 1)!/2$

Special Instances

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




$2 + 4 + 1 + 1 = 8$

**Formal Definition**

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

Solution space consists of at most $n!$ possible tours!

Actually the right number is $(n-1)!/2$

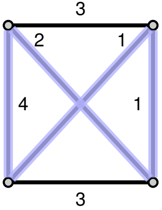
**Special Instances**

- Metric TSP: costs satisfy triangle inequality:

$$\forall u, v, w \in V: \qquad c(u, w) \leq c(u, v) + c(v, w).$$

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




$2 + 4 + 1 + 1 = 8$

**Formal Definition**

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

Solution space consists of at most $n!$ possible tours!

Actually the right number is $(n-1)!/2$

**Special Instances**

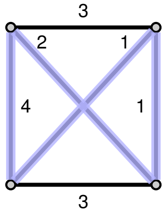
- Metric TSP: costs satisfy triangle inequality:

$$\forall u, v, w \in V: \qquad c(u, w) \leq c(u, v) + c(v, w).$$

- Euclidean TSP: cities are points in the Euclidean space, costs are equal to their (rounded) Euclidean distance

# The Traveling Salesman Problem (TSP)

*Given a set of cities along with the cost of travel between them, find the cheapest route visiting all cities and returning to your starting point.*




$2 + 4 + 1 + 1 = 8$

**Formal Definition**

- Given: A complete undirected graph $G = (V, E)$ with nonnegative integer cost $c(u, v)$ for each edge $(u, v) \in E$
- Goal: Find a hamiltonian cycle of $G$ with minimum cost.

Solution space consists of at most $n!$ possible tours!

Actually the right number is $(n-1)!/2$

**Special Instances**

- Metric TSP: costs satisfy triangle inequality:

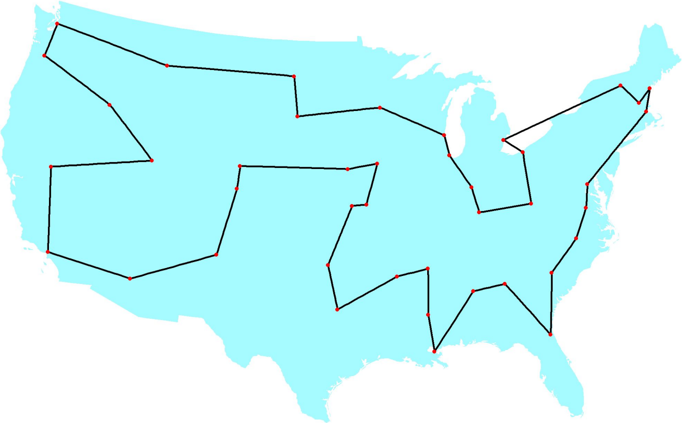
Even this version is NP hard (Ex. 35.2-2)

$$\forall u, v, w \in V: \qquad c(u, w) \leq c(u, v) + c(v, w).$$

- Euclidean TSP: cities are points in the Euclidean space, costs are equal to their (rounded) Euclidean distance

# History of the TSP problem (1954)

Dantzig, Fulkerson and Johnson found an optimal tour through 42 cities.



**http://www.math.uwaterloo.ca/tsp/history/img/dantzig_big.html**

# The Dantzig-Fulkerson-Johnson Method

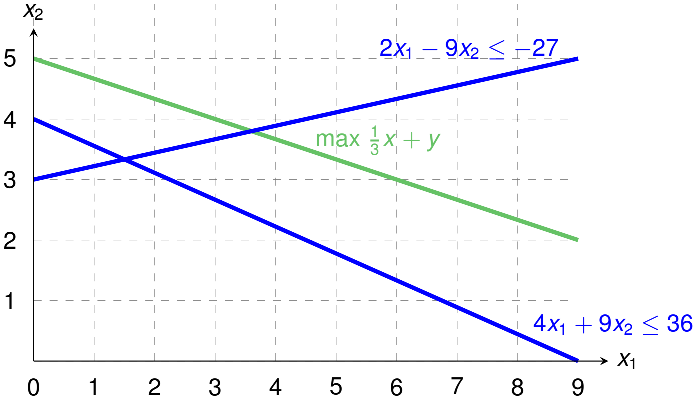
1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)

## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)

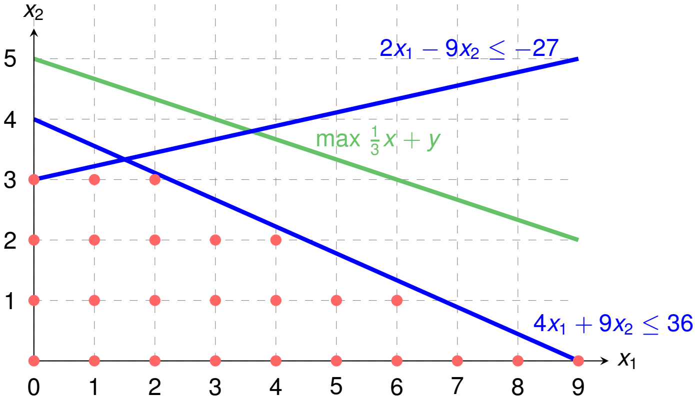
## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



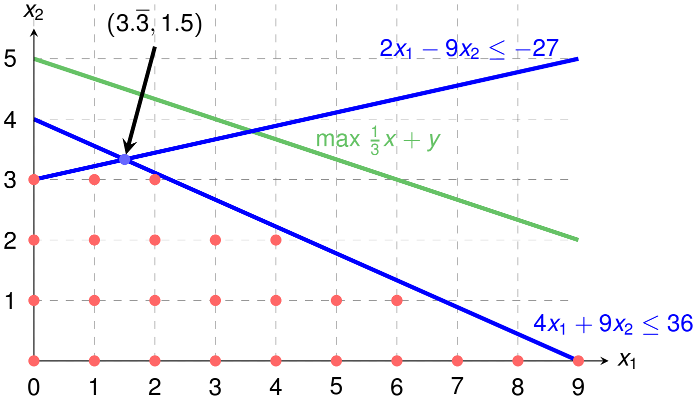
# The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



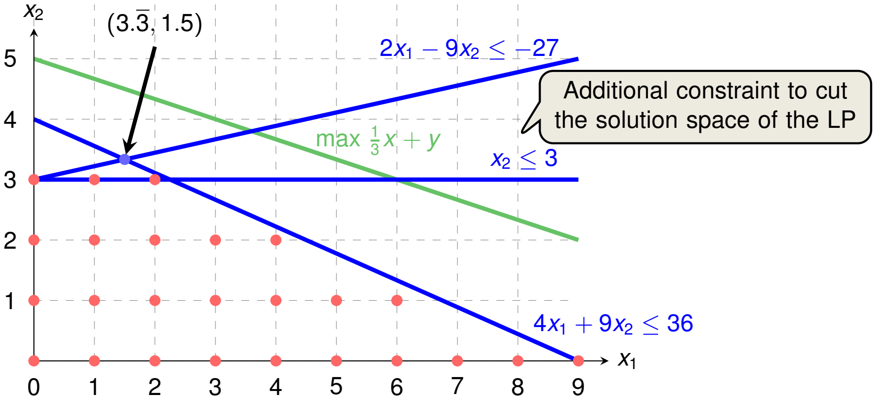
## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



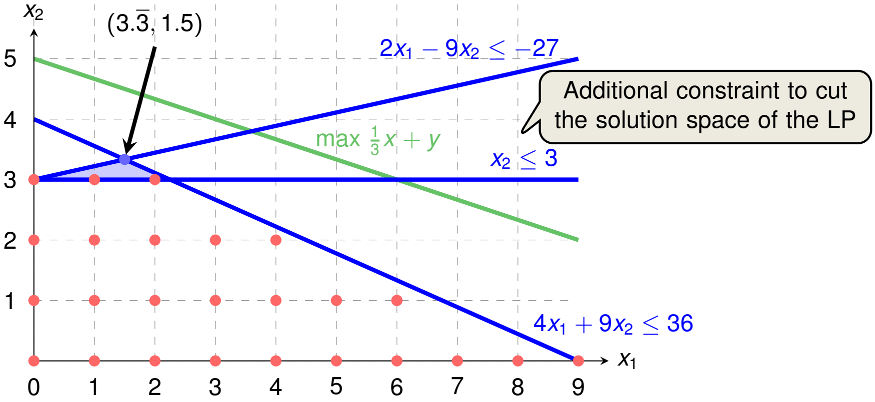
## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



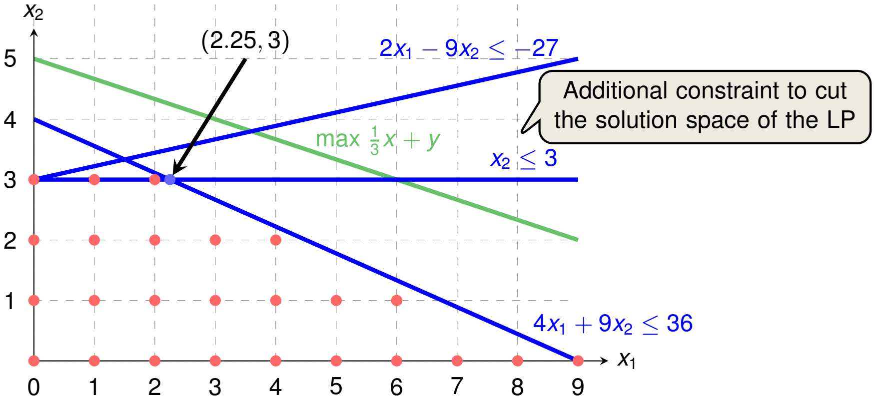
## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



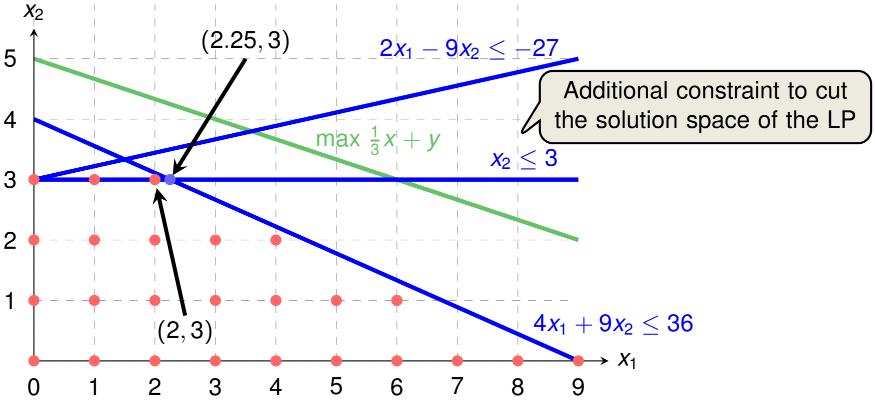
# The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



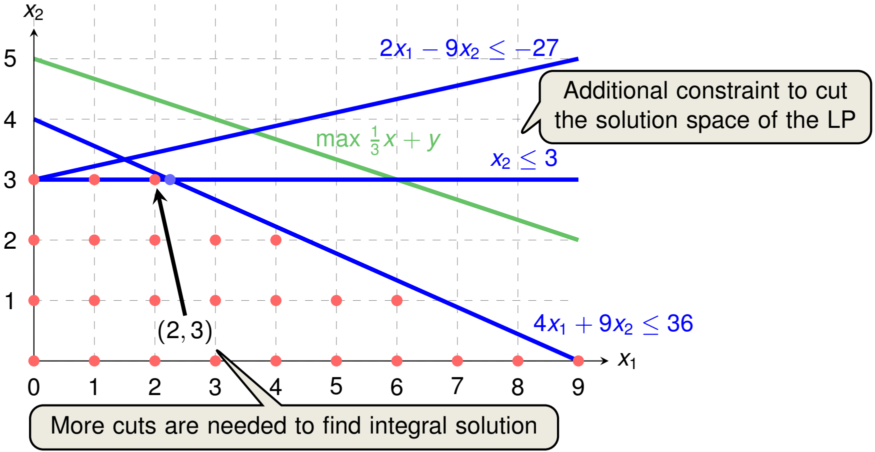
## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



## The Dantzig-Fulkerson-Johnson Method

1. Create a linear program (variable $x(u, v) = 1$ iff tour goes between $u$ and $v$)
2. Solve the linear program. If the solution is integral and forms a tour, stop. Otherwise find a new constraint to add (cutting plane)



# Outline

Introduction

General TSP

Metric TSP

# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

## Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Proof:

# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Proof:

Idea: Reduction from the hamiltonian-cycle problem.

**Theorem 35.3**

If $P \neq NP$, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem

## Hardness of Approximation

**Theorem 35.3**

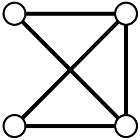
If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem

$G = (V, E)$



## Hardness of Approximation

**Theorem 35.3**

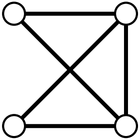
If $\mathrm{P} \neq \mathrm{NP}$, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$G = (V, E)$



# Hardness of Approximation

**Theorem 35.3**

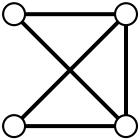
If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

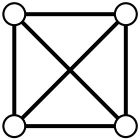
Proof:

Idea: Reduction from the hamiltonian-cycle problem.

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$G = (V, E)$





$G' = (V, E')$

# Hardness of Approximation

Theorem 35.3

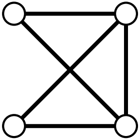
If $P \neq NP$, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

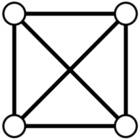
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

$G = (V, E)$ 

 $G' = (V, E')$

# Hardness of Approximation

**Theorem 35.3**

If $P \neq NP$, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

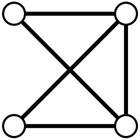
Idea: Reduction from the hamiltonian-cycle problem.

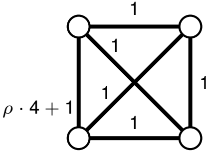
Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

$G = (V, E)$






$G' = (V, E')$

# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

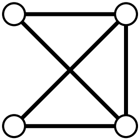
Proof:

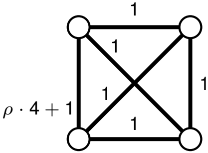
- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

Large weight will render this edge useless!

$G = (V, E)$






$G' = (V, E')$

## Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

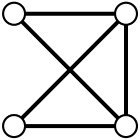
Proof:

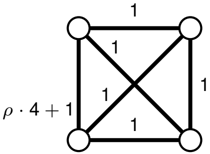
- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

Can create representations of $G'$ and $c$ in time polynomial in $|V|$ and $|E|$!

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

$G = (V, E)$






$G' = (V, E')$

# Hardness of Approximation

**Theorem 35.3**

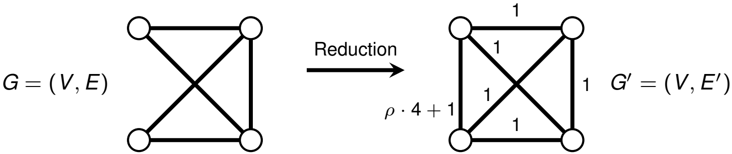
If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$



# Hardness of Approximation

**Theorem 35.3**

If $\mathrm{P} \neq \mathrm{NP}$, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

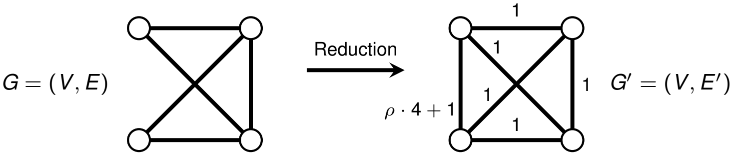
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$



## Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

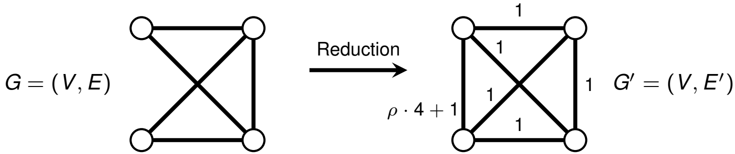
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$



## Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

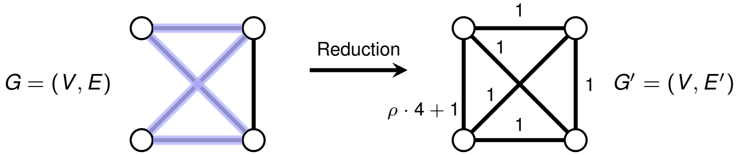
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$



**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

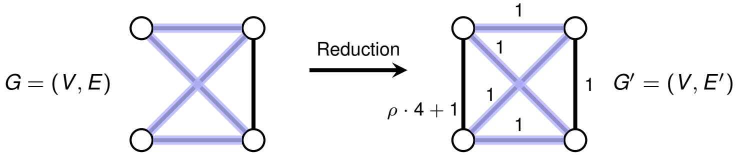
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$



# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

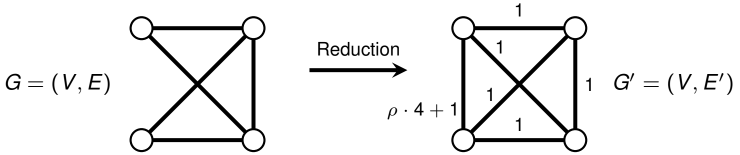
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,



# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

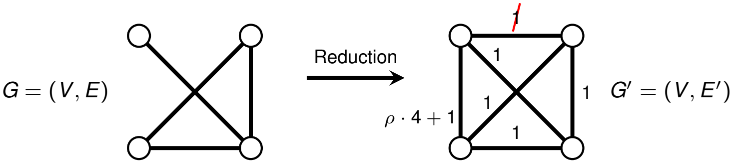
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,



# Hardness of Approximation

**Theorem 35.3**

If $\mathrm{P} \neq \mathrm{NP}$, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

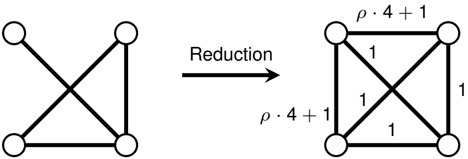
Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$G = (V, E)$




$G' = (V, E')$

# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

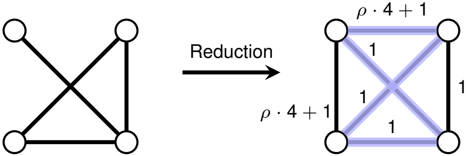
Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$G = (V, E)$




$G' = (V, E')$

## Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

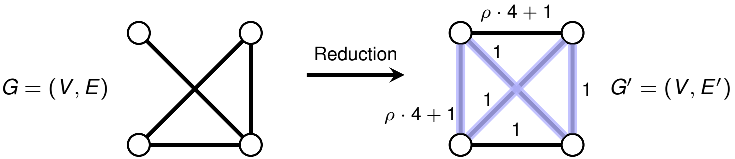
Idea: Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,



## Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

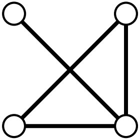
Proof:

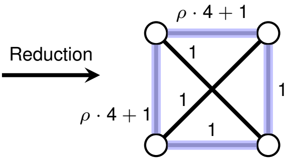
- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$G = (V, E)$






$G' = (V, E')$

# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

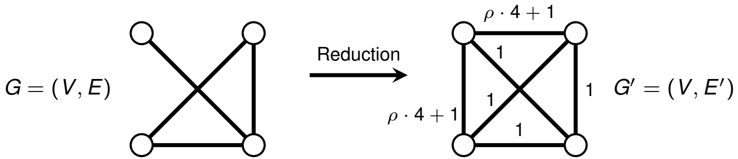
Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$$\Rightarrow \qquad c(T) \geq (\rho|V| + 1) + (|V| - 1)$$



# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

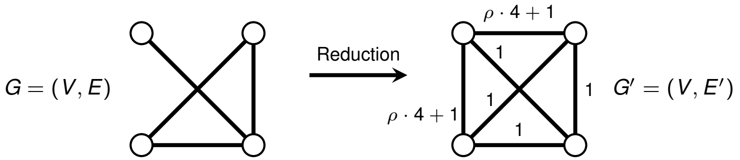
Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$$\Rightarrow \qquad c(T) \geq (\rho|V| + 1) + (|V| - 1) = (\rho + 1)|V|.$$



# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

Proof:

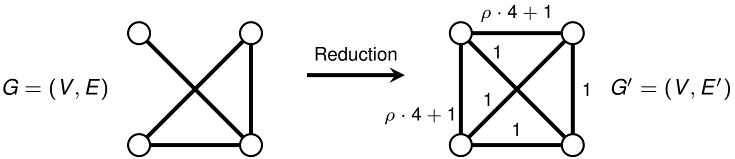
- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$$\Rightarrow \qquad c(T) \geq (\rho|V| + 1) + (|V| - 1) = (\rho + 1)|V|.$$

- Gap of $\rho + 1$ between tours which are using only edges in $G$ and those which don't



# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

Idea: Reduction from the hamiltonian-cycle problem.

Proof:

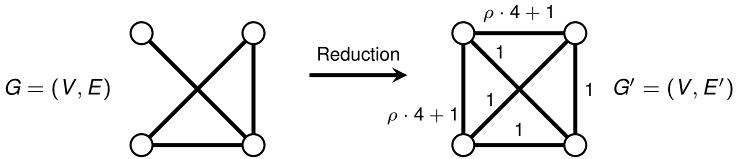
- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$$\Rightarrow \qquad c(T) \geq (\rho|V| + 1) + (|V| - 1) = (\rho + 1)|V|.$$

- Gap of $\rho + 1$ between tours which are using only edges in $G$ and those which don't
- $\rho$-Approximation of TSP in $G'$ computes hamiltonian cycle in $G$ (if one exists)



# Hardness of Approximation

**Theorem 35.3**

If P $\neq$ NP, then for any constant $\rho \geq 1$, there is no polynomial-time approximation algorithm with approximation ratio $\rho$ for the general TSP.

**Idea:** Reduction from the hamiltonian-cycle problem.

Proof:

- Let $G = (V, E)$ be an instance of the hamiltonian-cycle problem
- Let $G' = (V, E')$ be a complete graph with costs for each $(u, v) \in E'$:

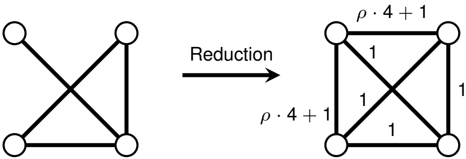
$$c(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E, \\ \rho|V| + 1 & \text{otherwise.} \end{cases}$$

- If $G$ has a hamiltonian cycle $H$, then $(G', c)$ contains a tour of cost $|V|$
- If $G$ does not have a hamiltonian cycle, then any tour $T$ must use some edge $\notin E$,

$$\Rightarrow \qquad c(T) \geq (\rho|V| + 1) + (|V| - 1) = (\rho + 1)|V|.$$

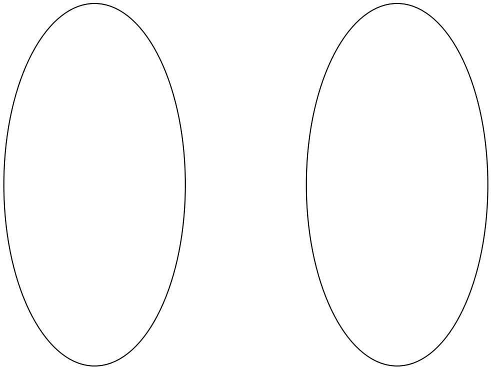
- Gap of $\rho + 1$ between tours which are using only edges in $G$ and those which don't
- $\rho$-Approximation of TSP in $G'$ computes hamiltonian cycle in $G$ (if one exists) □

$G = (V, E)$



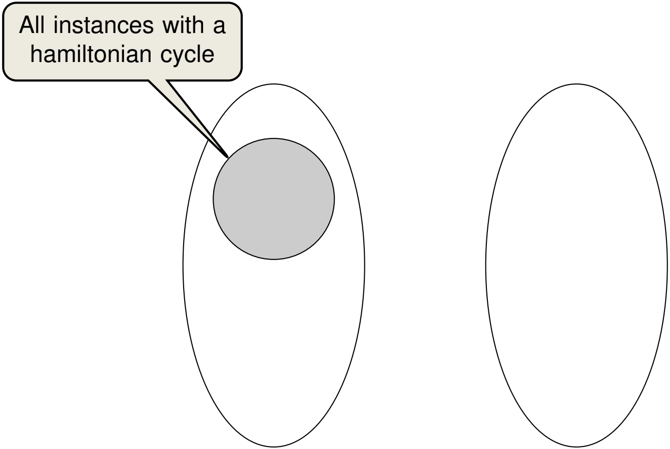

$G' = (V, E')$

# Proof of Theorem 35.3 from a higher perspective



instances of Hamilton          instances of TSP

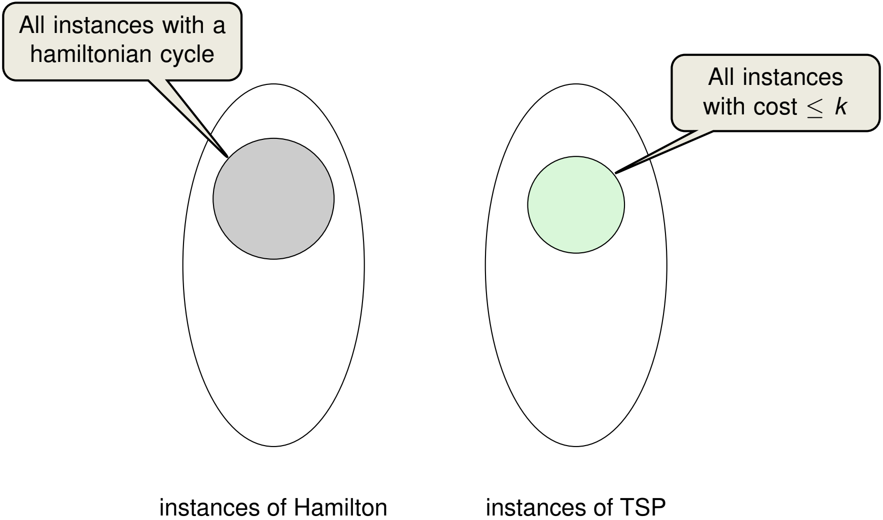
# Proof of Theorem 35.3 from a higher perspective



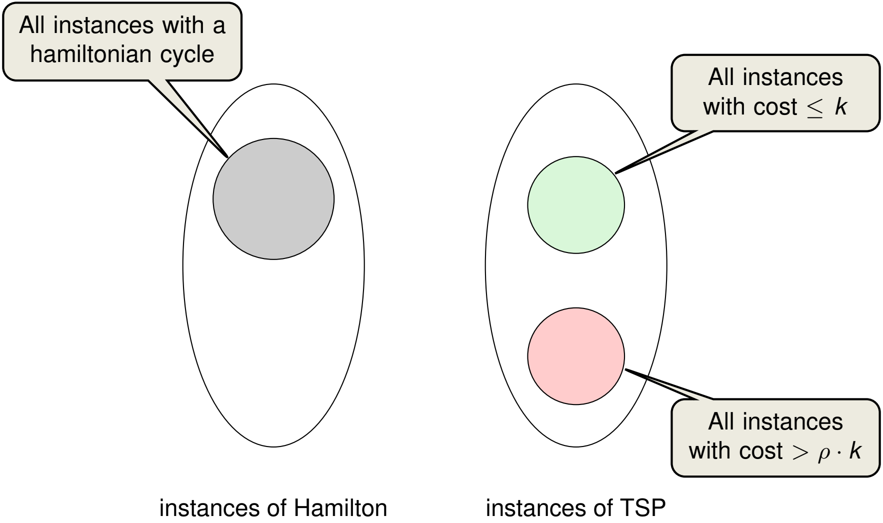

instances of Hamilton

instances of TSP

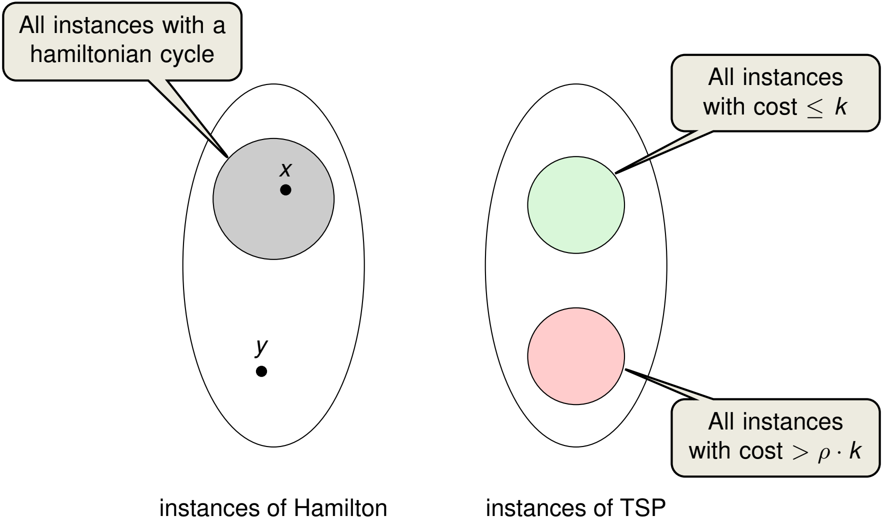
# Proof of Theorem 35.3 from a higher perspective



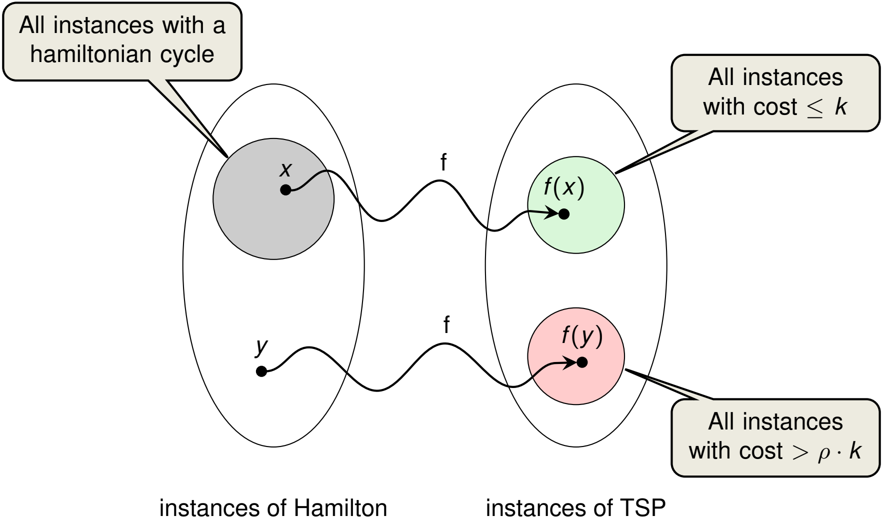
# Proof of Theorem 35.3 from a higher perspective



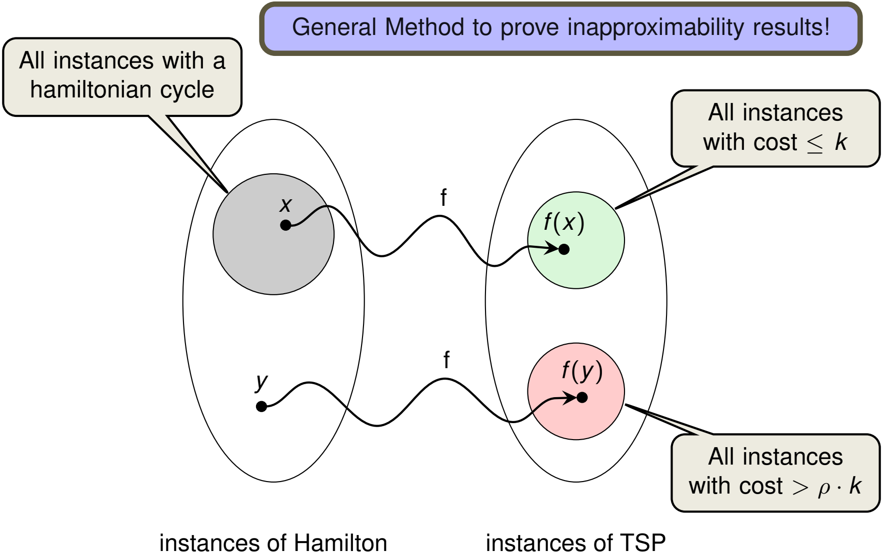
# Proof of Theorem 35.3 from a higher perspective



## Proof of Theorem 35.3 from a higher perspective



# Proof of Theorem 35.3 from a higher perspective



# Outline

Introduction

General TSP

Metric TSP

# Metric TSP (TSP Problem with the Triangle Inequality)

Idea: First compute an MST, and then create a tour based on the tree.

# Metric TSP (TSP Problem with the Triangle Inequality)

Idea: First compute an MST, and then create a tour based on the tree.

```
APPROX-TSP-TOUR(G, c)
select a vertex r ∈ G.V to be a "root" vertex
compute a minimum spanning tree T for G from root r
    using MST-PRIM(G, c, r)
let H be a list of vertices, ordered according to when they are first visited
    in a preorder tree walk of T
return the hamiltonian cycle H
```

# Metric TSP (TSP Problem with the Triangle Inequality)

Idea: First compute an MST, and then create a tour based on the tree.

APPROX-TSP-TOUR$(G, c)$

1 select a vertex $r \in G.V$ to be a "root" vertex
2 compute a minimum spanning tree $T$ for $G$ from root $r$
  using MST-PRIM$(G, c, r)$
3 let $H$ be a list of vertices, ordered according to when they are first visited
  in a preorder tree walk of $T$
4 **return** the hamiltonian cycle $H$

Runtime is dominated by MST-PRIM, which is $\Theta(V^2)$.

# Metric TSP (TSP Problem with the Triangle Inequality)

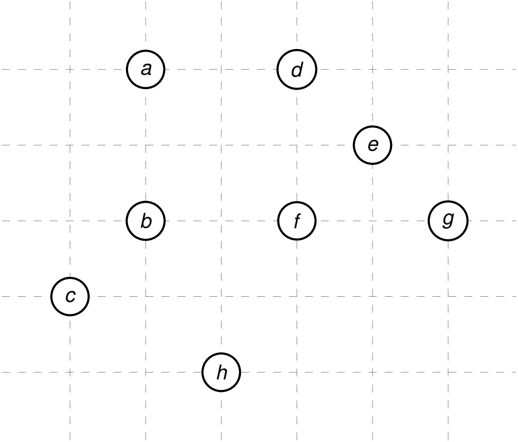
> Idea: First compute an MST, and then create a tour based on the tree.

```
APPROX-TSP-TOUR(G, c)
    select a vertex r ∈ G.V to be a "root" vertex
    compute a minimum spanning tree T for G from root r
        using MST-PRIM(G, c, r)
    let H be a list of vertices, ordered according to when they are first visited
        in a preorder tree walk of T
    return the hamiltonian cycle H
```

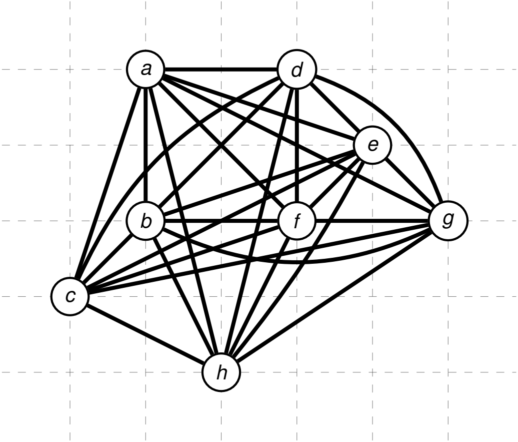
Runtime is dominated by MST-PRIM, which is $\Theta(V^2)$.

Remember: In the Metric-TSP problem, $G$ is a complete graph.

# Run of APPROX-TSP-TOUR

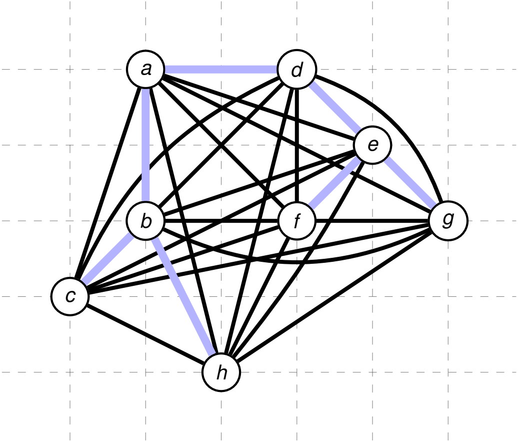


# Run of APPROX-TSP-TOUR



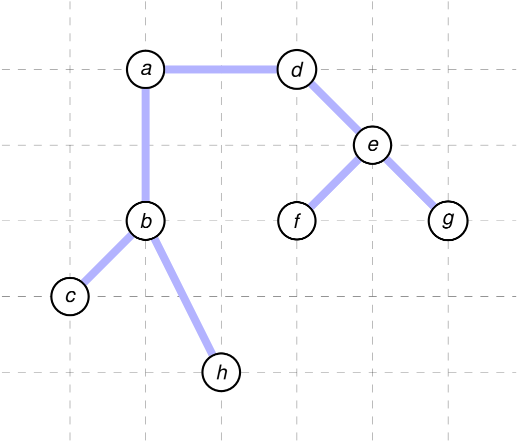

1. Compute MST

# Run of APPROX-TSP-TOUR



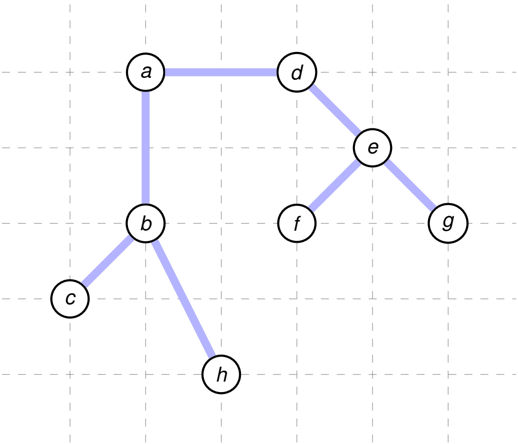

1. Compute MST

# Run of APPROX-TSP-TOUR



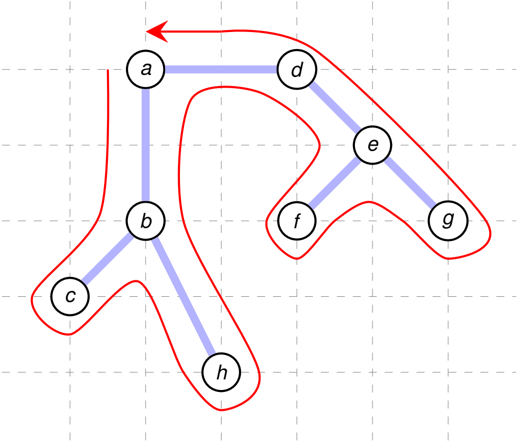

1. Compute MST ✓

# Run of APPROX-TSP-TOUR



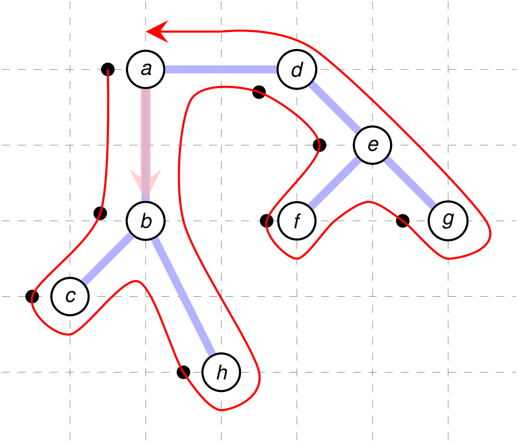

1. Compute MST ✓
2. Perform preorder walk on MST

# Run of APPROX-TSP-TOUR



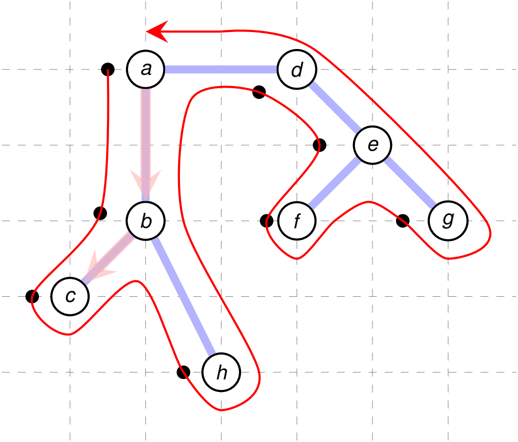

1. Compute MST ✓
2. Perform preorder walk on MST ✓

# Run of APPROX-TSP-TOUR



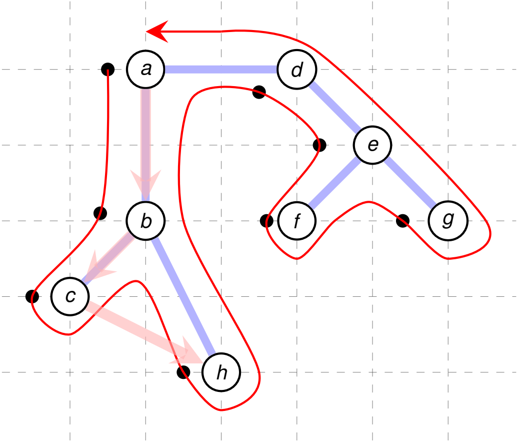

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

# Run of APPROX-TSP-TOUR



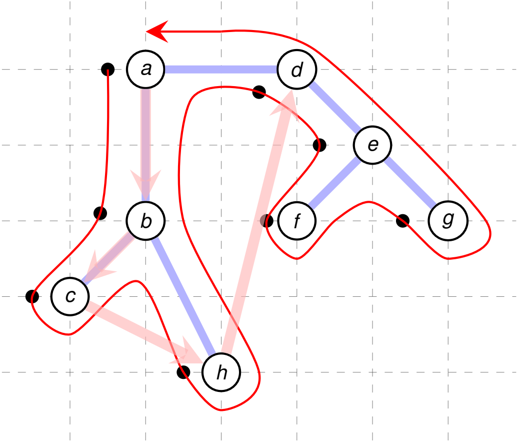

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

# Run of APPROX-TSP-TOUR



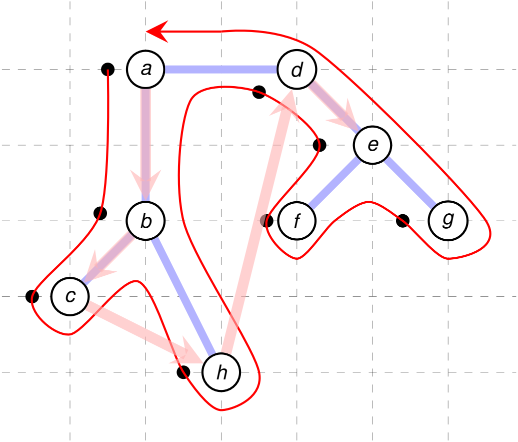

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

# Run of APPROX-TSP-TOUR



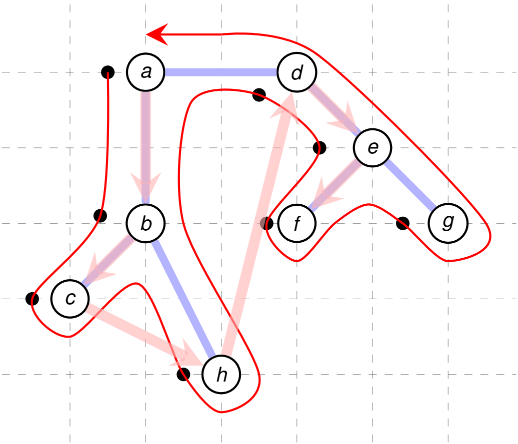

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

# Run of APPROX-TSP-TOUR



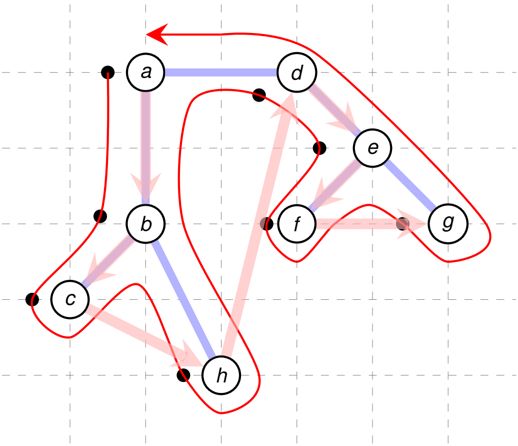

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

## Run of APPROX-TSP-TOUR



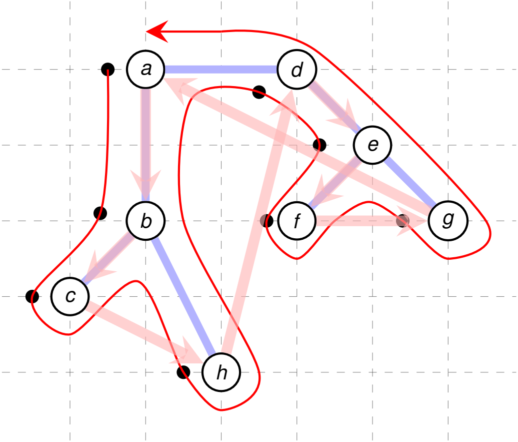

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

## Run of APPROX-TSP-TOUR



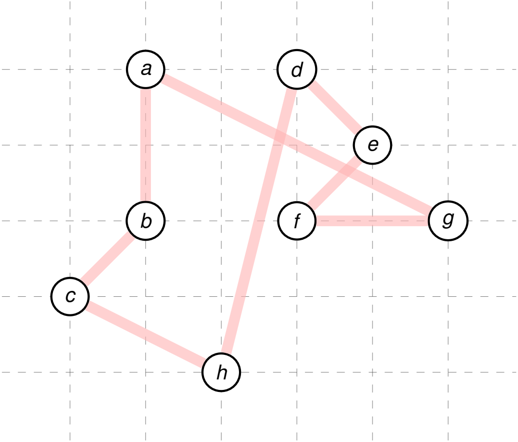

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

# Run of APPROX-TSP-TOUR



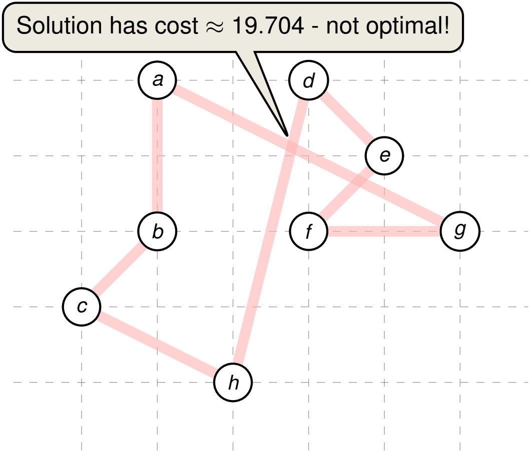

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk

# Run of APPROX-TSP-TOUR



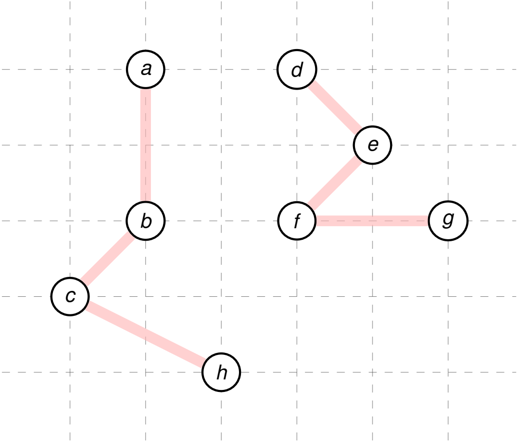

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk ✓

# Run of APPROX-TSP-TOUR



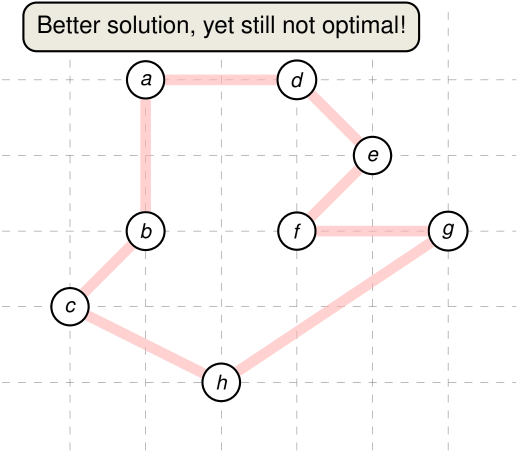

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk ✓

# Run of APPROX-TSP-TOUR



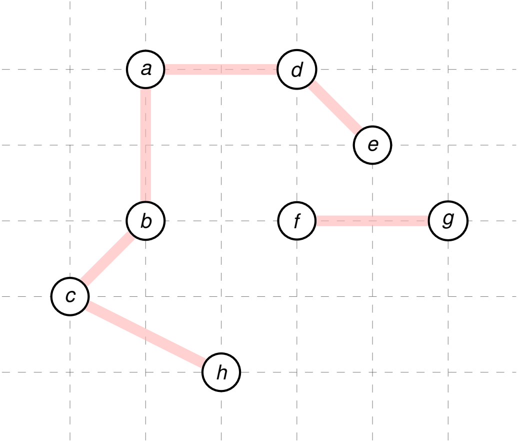

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk ✓

# Run of APPROX-TSP-TOUR



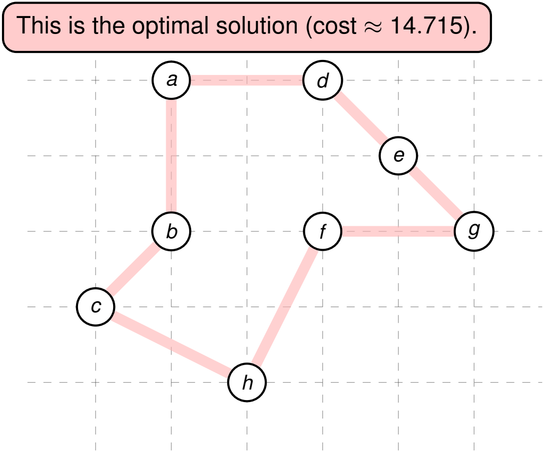

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk ✓

## Run of APPROX-TSP-TOUR



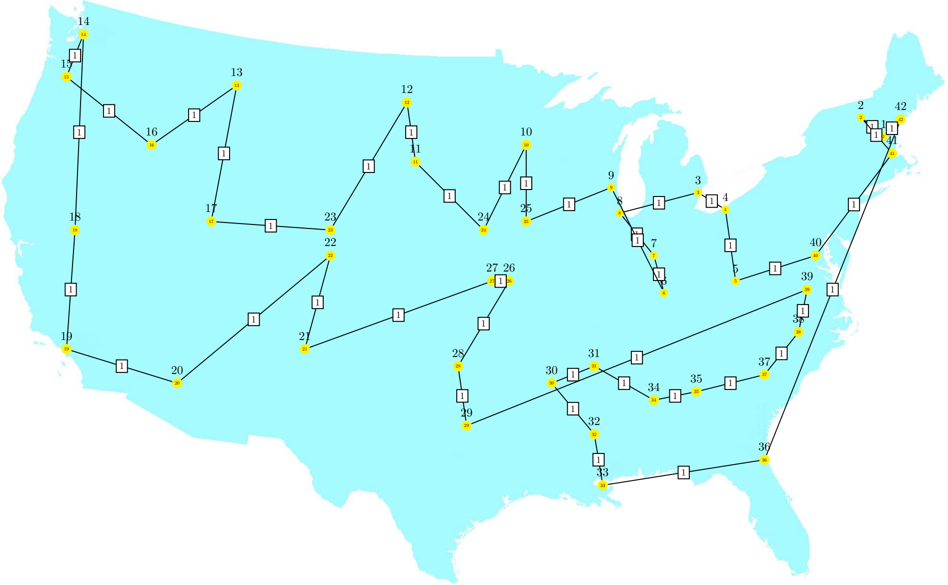

1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk ✓

# Run of APPROX-TSP-TOUR

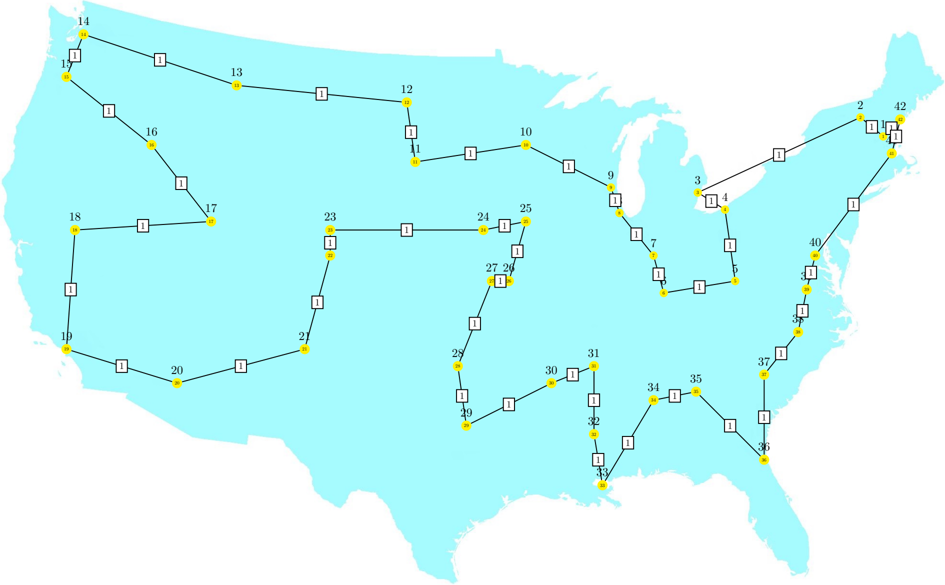



1. Compute MST ✓
2. Perform preorder walk on MST ✓
3. Return list of vertices according to the preorder tree walk ✓

# Approximate Solution: Objective 921



# Optimal Solution: Objective 699



## Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

## Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

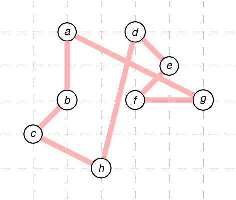
Proof:

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:



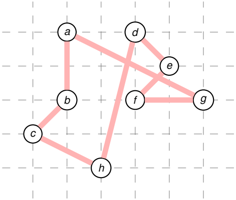

solution $H$ of APPROX-TSP

## Proof of the Approximation Ratio

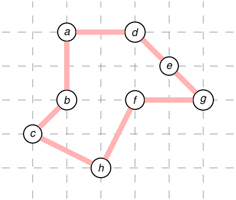
**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:




solution $H$ of APPROX-TSP




optimal solution $H^*$

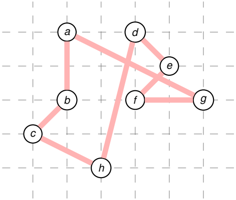
# Proof of the Approximation Ratio

**Theorem 35.2**

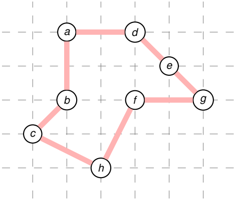
APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge




solution $H$ of APPROX-TSP




optimal solution $H^*$

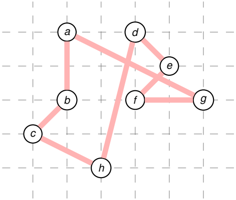
# Proof of the Approximation Ratio

**Theorem 35.2**

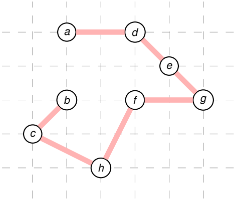
APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge



solution $H$ of APPROX-TSP



spanning tree $T$ as a subset of $H^*$

# Proof of the Approximation Ratio

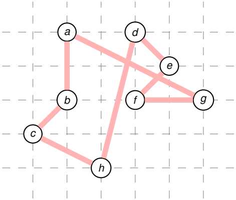
**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

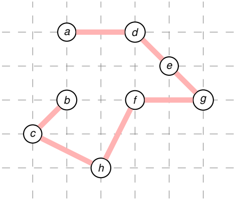
Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and




solution $H$ of APPROX-TSP




spanning tree $T$ as a subset of $H^*$

# Proof of the Approximation Ratio

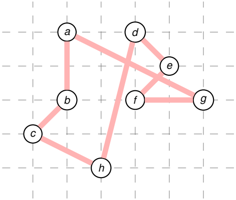
**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

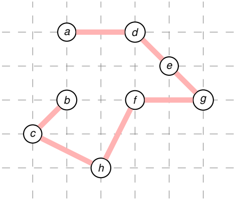
Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$




solution $H$ of APPROX-TSP




spanning tree $T$ as a subset of $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

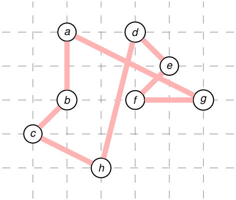
APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

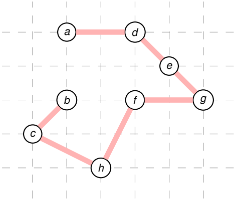
- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

exploiting that all edge costs are non-negative!




solution $H$ of APPROX-TSP




spanning tree $T$ as a subset of $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

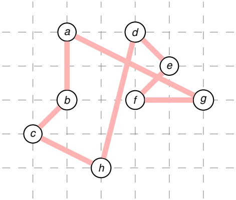
APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

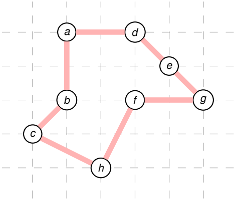
- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)




solution $H$ of APPROX-TSP




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

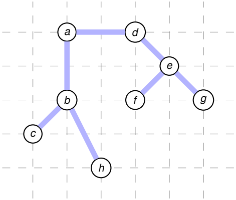
APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

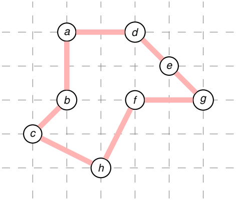
- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)




minimum spanning tree $T_{\min}$




optimal solution $H^*$

## Proof of the Approximation Ratio

**Theorem 35.2**

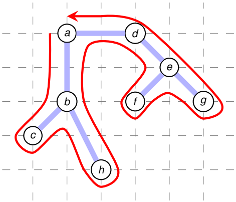
APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

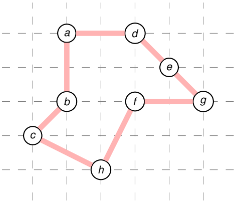
- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)




Walk $W = (a, b, c, b, h, b, a, d, e, f, e, g, e, d, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

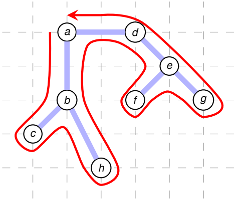
Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

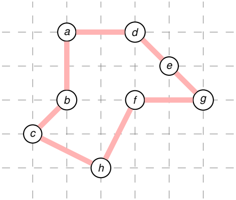
$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

$\Rightarrow$ Full walk traverses every edge exactly twice, so




Walk $W = (a, b, c, b, h, b, a, d, e, f, e, g, e, d, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

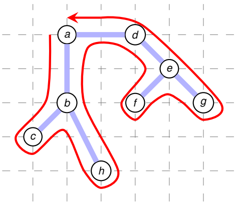
- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

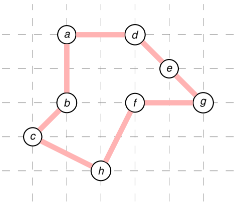
- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min})$$




Walk $W = (a, b, c, b, h, b, a, d, e, f, e, g, e, d, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

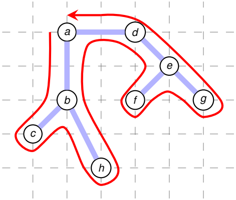
- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

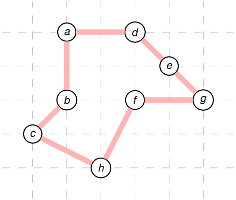
- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$




Walk $W = (a, b, c, b, h, b, a, d, e, f, e, g, e, d, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

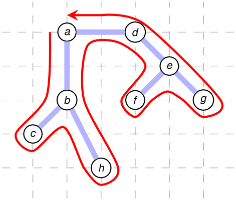
$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

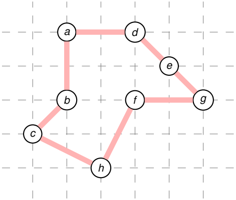
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

- Deleting duplicate vertices from $W$ yields a tour $H$




Walk $W = (a, b, c, b, h, b, a, d, e, f, e, g, e, d, a)$




optimal solution $H^*$

## Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

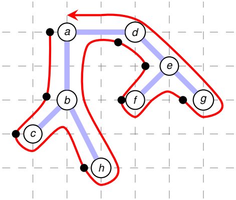
$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

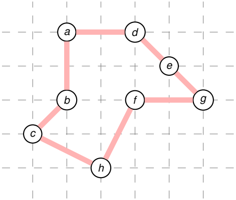
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

- Deleting duplicate vertices from $W$ yields a tour $H$




Walk $W = (a, b, c, b, h, b, a, d, e, f, e, g, e, d, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

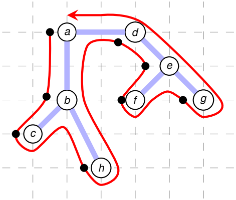
$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

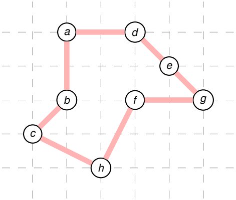
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

- Deleting duplicate vertices from $W$ yields a tour $H$




Walk $W = (a, b, c, \cancel{b}, h, \cancel{b}, \cancel{a}, d, e, f, \cancel{e}, g, \cancel{e}, \cancel{d}, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

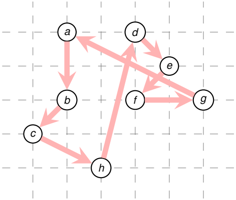
$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

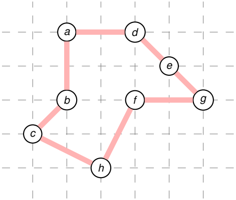
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

- Deleting duplicate vertices from $W$ yields a tour $H$




Tour $H = (a, b, c, h, d, e, f, g, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

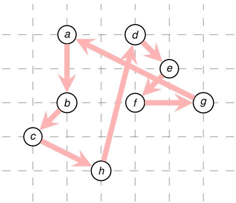
- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

$\Rightarrow$ Full walk traverses every edge exactly twice, so

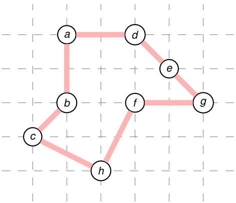
$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

exploiting triangle inequality!

- Deleting duplicate vertices from $W$ yields a tour $H$ with smaller cost:




Tour $H = (a, b, c, h, d, e, f, g, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

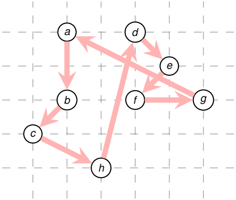
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

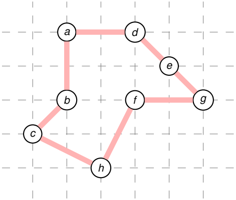
exploiting triangle inequality!

- Deleting duplicate vertices from $W$ yields a tour $H$ with smaller cost:

$$c(H) \leq c(W)$$




Tour $H = (a, b, c, h, d, e, f, g, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

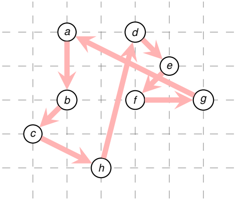
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

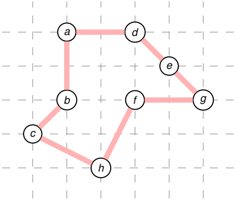
exploiting triangle inequality!

- Deleting duplicate vertices from $W$ yields a tour $H$ with smaller cost:

$$c(H) \leq c(W) \leq 2c(H^*)$$




Tour $H = (a, b, c, h, d, e, f, g, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\text{min}}$ (including repeated visits)

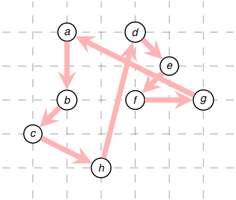
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\text{min}}) \leq 2c(T) \leq 2c(H^*)$$

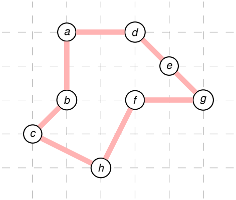
exploiting triangle inequality!

- Deleting duplicate vertices from $W$ yields a tour $H$ with smaller cost:

$$c(H) \leq c(W) \leq 2c(H^*)$$




Tour $H = (a, b, c, h, d, e, f, g, a)$




optimal solution $H^*$

# Proof of the Approximation Ratio

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Proof:

- Consider the optimal tour $H^*$ and remove an arbitrary edge

$\Rightarrow$ yields a spanning tree $T$ and $c(T) \leq c(H^*)$

- Let $W$ be the full walk of the minimum spanning tree $T_{\min}$ (including repeated visits)

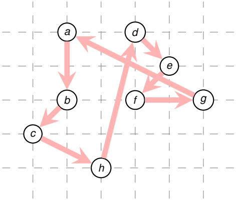
$\Rightarrow$ Full walk traverses every edge exactly twice, so

$$c(W) = 2c(T_{\min}) \leq 2c(T) \leq 2c(H^*)$$

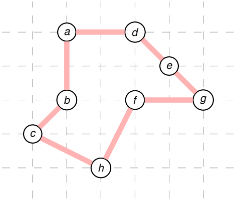
exploiting triangle inequality!

- Deleting duplicate vertices from $W$ yields a tour $H$ with smaller cost:

$$c(H) \leq c(W) \leq 2c(H^*) \qquad \square$$




Tour $H = (a, b, c, h, d, e, f, g, a)$




optimal solution $H^*$

# Christofides Algorithm

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

# Christofides Algorithm

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Can we get a better approximation ratio?

# Christofides Algorithm

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

Can we get a better approximation ratio?

CHRISTOFIDES$(G, c)$
select a vertex $r \in G.V$ to be a “root” vertex
compute a minimum spanning tree $T$ for $G$ from root $r$
    using MST-PRIM$(G, c, r)$
compute a perfect matching $M$ with minimum weight in the complete graph
    over the odd-degree vertices in $T$
let $H$ be a list of vertices, ordered according to when they are first visited
    in a Eulearian circuit of $T \cup M$
**return** $H$

# Christofides Algorithm

**Theorem 35.2**

APPROX-TSP-TOUR is a polynomial-time 2-approximation for the traveling-salesman problem with the triangle inequality.

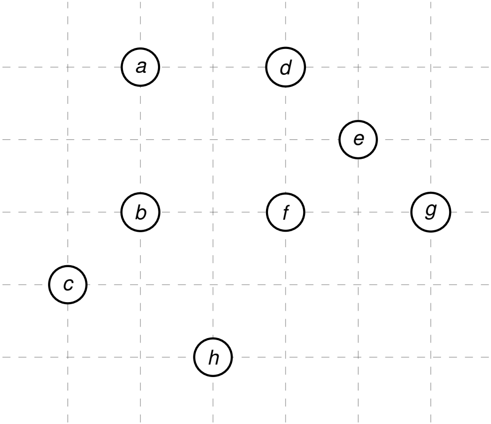
Can we get a better approximation ratio?

```
CHRISTOFIDES(G, c)
select a vertex r ∈ G.V to be a "root" vertex
compute a minimum spanning tree T for G from root r
        using MST-PRIM(G, c, r)
compute a perfect matching M with minimum weight in the complete graph
        over the odd-degree vertices in T
let H be a list of vertices, ordered according to when they are first visited
        in a Eulearian circuit of T ∪ M
return H
```

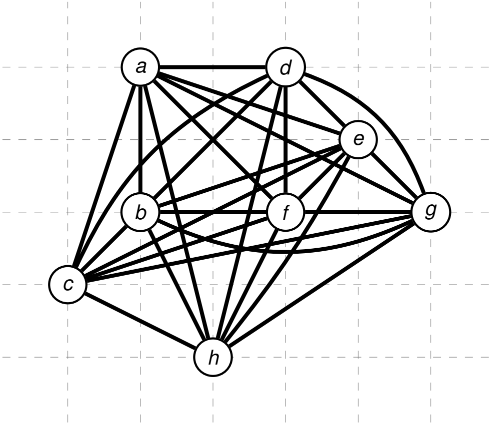
**Theorem (Christofides'76)**

There is a polynomial-time $\frac{3}{2}$-approximation algorithm for the travelling salesman problem with the triangle inequality.

# Run of CHRISTOFIDES

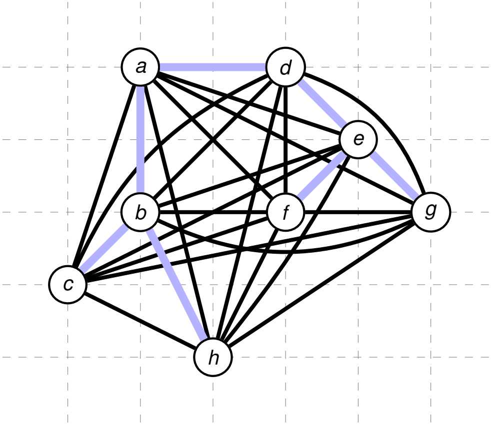


# Run of CHRISTOFIDES



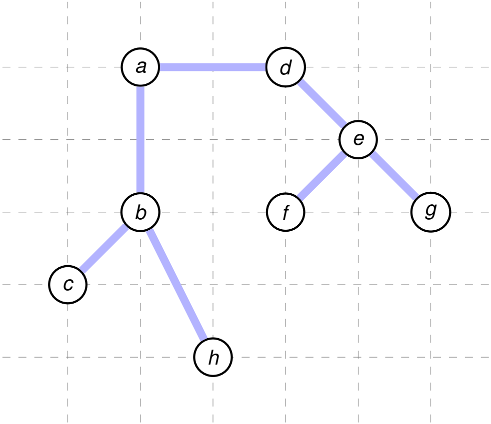

1. Compute MST

## Run of CHRISTOFIDES



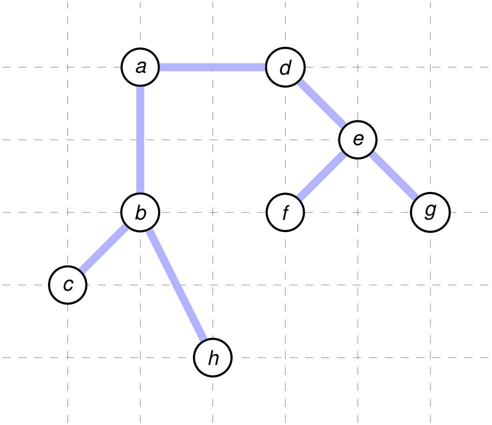

1. Compute MST

# Run of CHRISTOFIDES



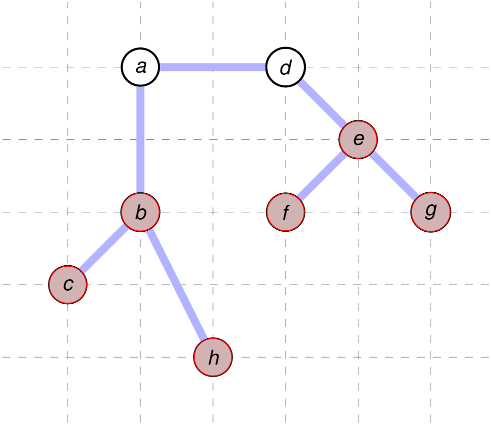

1. Compute MST ✓

## Run of CHRISTOFIDES



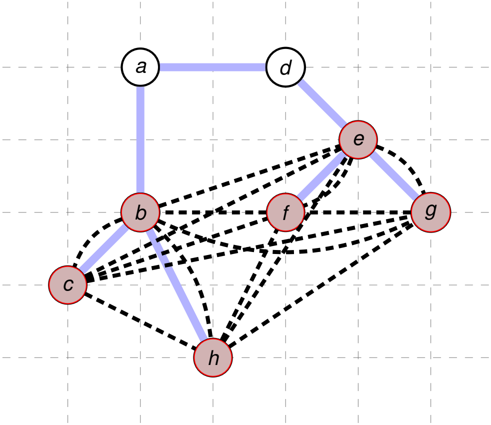

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$

# Run of CHRISTOFIDES



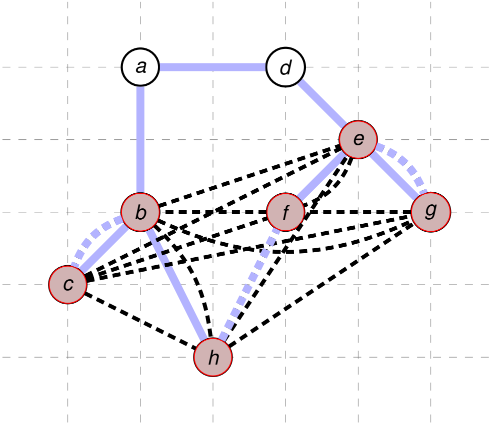

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$

## Run of CHRISTOFIDES



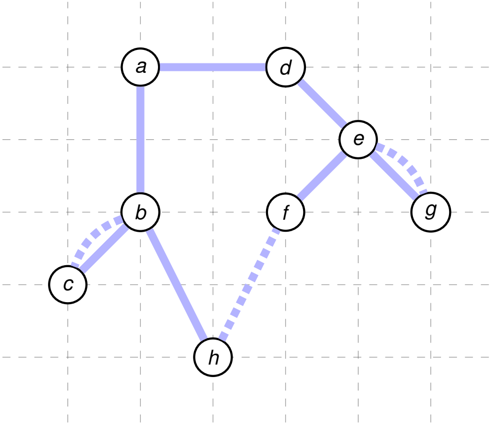

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$

# Run of CHRISTOFIDES



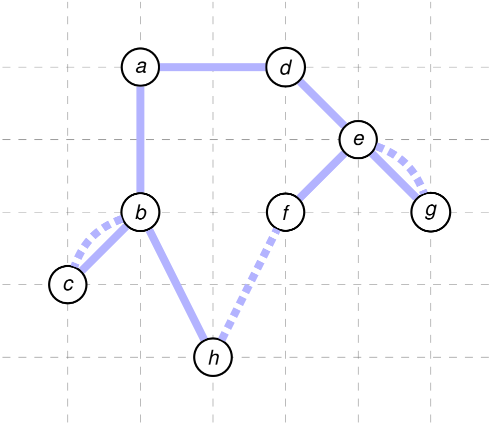

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$

## Run of CHRISTOFIDES




1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓

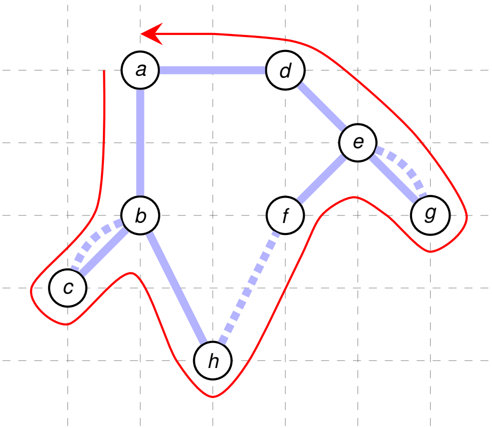
# Run of CHRISTOFIDES




1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit

All vertices in $T \cup M$ have even degree!

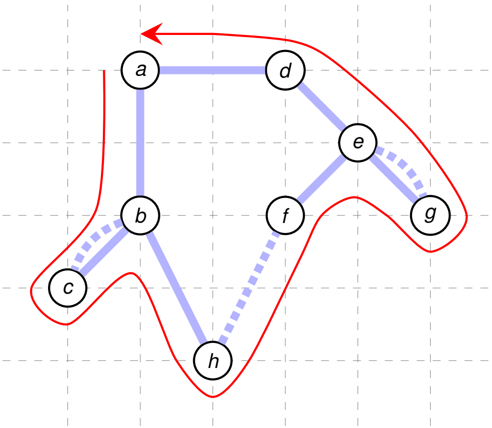
# Run of CHRISTOFIDES




1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓

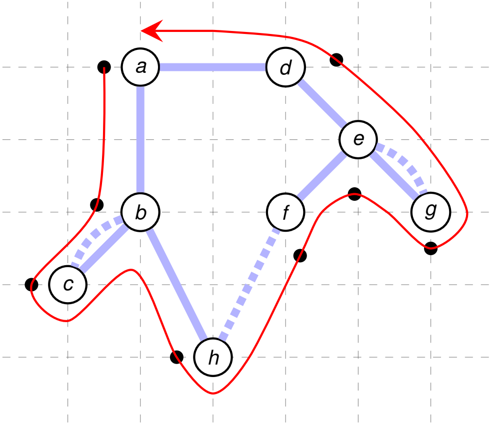
All vertices in $T \cup M$ have even degree!

# Run of CHRISTOFIDES



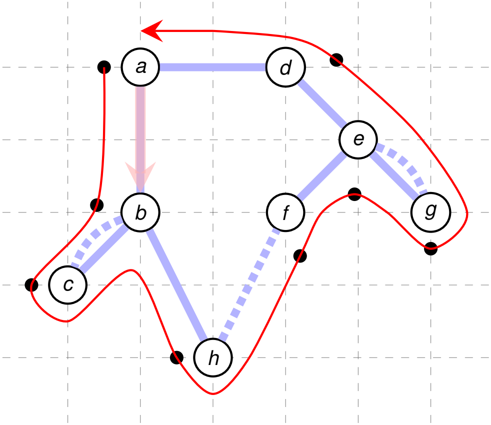

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES



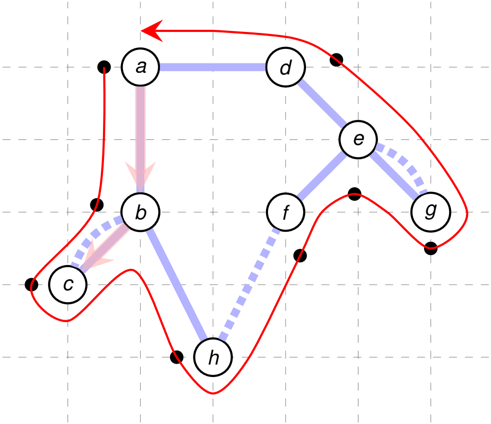

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES



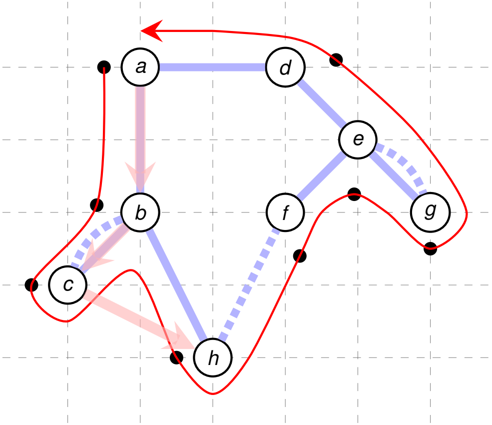

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES



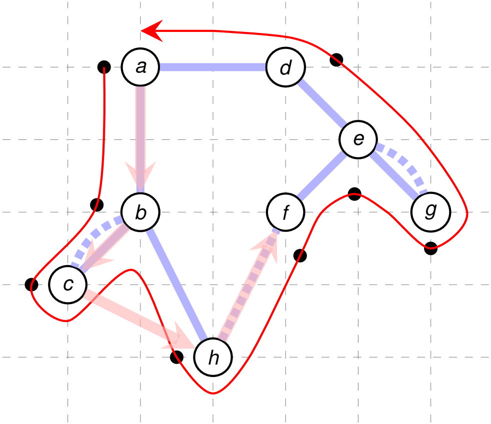

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES



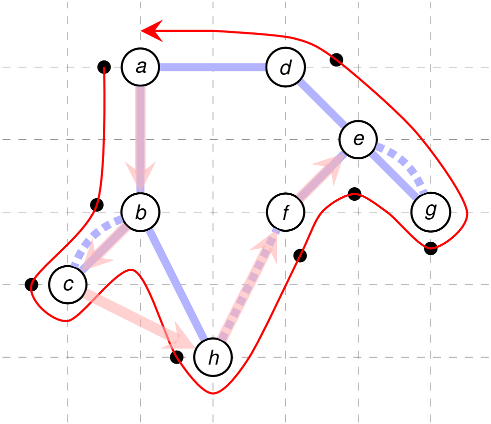

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES



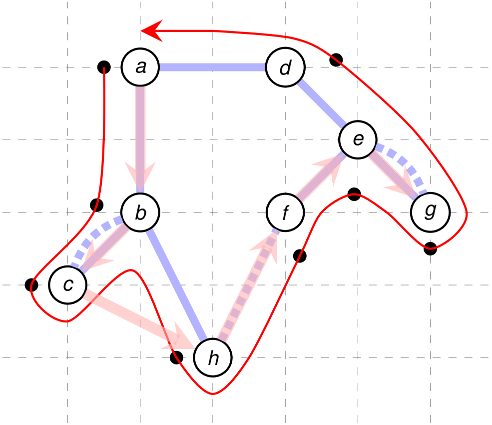

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES



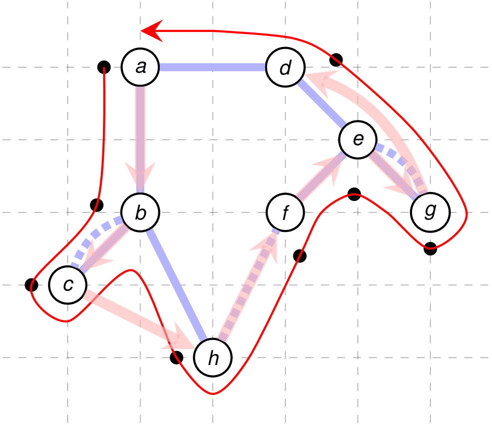

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

## Run of CHRISTOFIDES



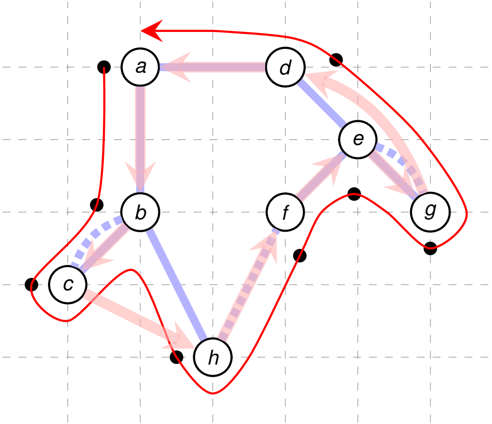

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

## Run of Christofides



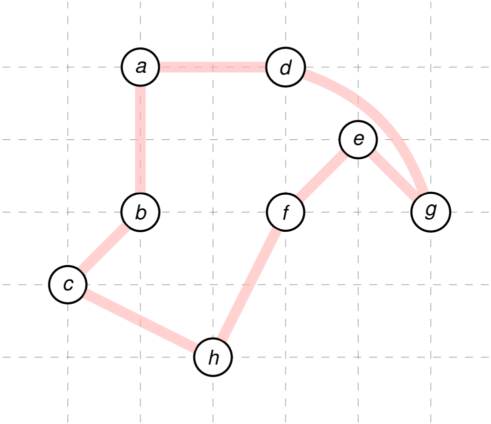

1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

# Run of CHRISTOFIDES




1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle

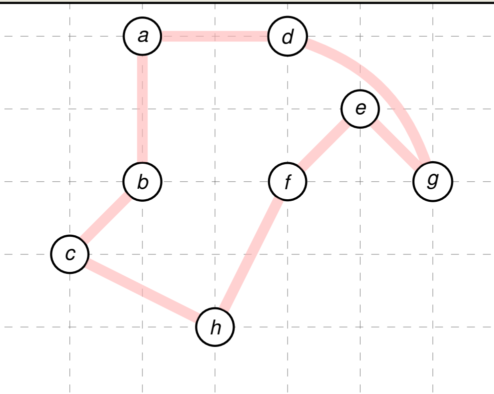
## Run of CHRISTOFIDES




1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle ✓

# Run of CHRISTOFIDES

Solution has cost $\approx$ 15.54 - within 10% of the optimum!




1. Compute MST ✓
2. Add a minimum-weight perfect matching $M$ of the odd vertices in $T$ ✓
3. Find an Eulerian Circuit ✓
4. Transform the Circuit into a Hamiltonian Cycle ✓

# Concluding Remarks

**Theorem (Christofides'76)**

There is a polynomial-time $\frac{3}{2}$-approximation algorithm for the travelling salesman problem with the triangle inequality.

## Concluding Remarks

**Theorem (Christofides'76)**

There is a polynomial-time $\frac{3}{2}$-approximation algorithm for the travelling salesman problem with the triangle inequality.

**Theorem (Arora'96, Mitchell'96)**

There is a PTAS for the Euclidean TSP Problem.

## Concluding Remarks

**Theorem (Christofides'76)**

There is a polynomial-time $\frac{3}{2}$-approximation algorithm for the travelling salesman problem with the triangle inequality.

Both received the Gödel Award 2010

**Theorem (Arora'96, Mitchell'96)**

There is a PTAS for the Euclidean TSP Problem.

## Concluding Remarks

**Theorem (Christofides'76)**

There is a polynomial-time $\frac{3}{2}$-approximation algorithm for the travelling salesman problem with the triangle inequality.

Both received the Gödel Award 2010

**Theorem (Arora'96, Mitchell'96)**

There is a PTAS for the Euclidean TSP Problem.

*"Christos Papadimitriou told me that the traveling salesman problem is not a problem. It's an addiction."*

Jon Bentley 1991

# Concluding Remarks

**Theorem (Christofides'76)**

There is a polynomial-time $\frac{3}{2}$-approximation algorithm for the travelling salesman problem with the triangle inequality.

Both received the Gödel Award 2010

**Theorem (Arora'96, Mitchell'96)**

There is a PTAS for the Euclidean TSP Problem.

*"Christos Papadimitriou told me that the traveling salesman problem is not a problem. It's an addiction."*

Jon Bentley 1991