Inapproximability
CSCI 532

Test 3 Logistics

1. During class on Thursday 11/20.
2. You can bring your book and any notes you would like, but no electronic devices.
3. You may assume anything proven in class or on homework.
4. Three questions (10 points):
   1) Approximation algorithm (5 points).
   2) Special case (2 points).
   3) Tightness example (3 points).

Project Logistics

1. Report due 12/2 (first day of presentations).
2. Presentation schedule posted.
3. 15 minutes <u>maximum</u> presentation time. Leave a couple minutes for questions.
4. Attend other peoples’ presentations. If attendance plummets, I may have to factor attendance into project grades.

Project Logistics

1. Report due 12/2 (first day of presentations).
2. Presentation schedule posted.
3. 15 minutes maximum presentation time. Leave a couple minutes for questions.
4. Attend other peoples' presentations. If attendance plummets, I may have to factor attendance into project grades.

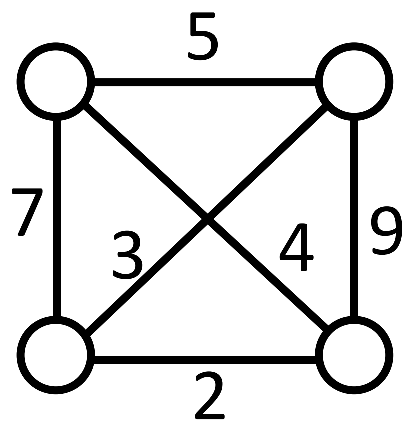
Also, please do the course evaluation:
https://faculty.campuslabs.com/eval-home/direct/8521795

Travelling Salesman Problem

TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?

$\in$ NP-Complete

Example:

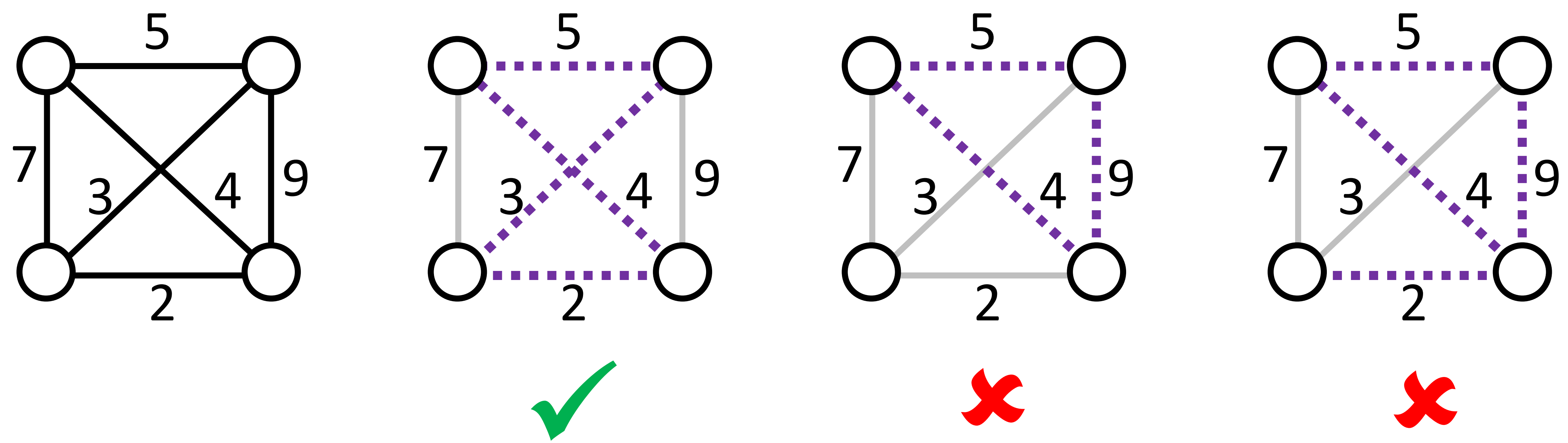


Travelling Salesman Problem

TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?

∈ NP-Complete

Example:

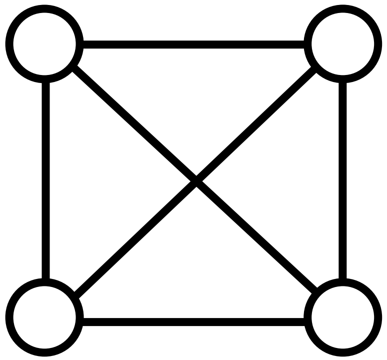


Hamiltonian Cycle Problem

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

$\in$ NP-Complete

Example:

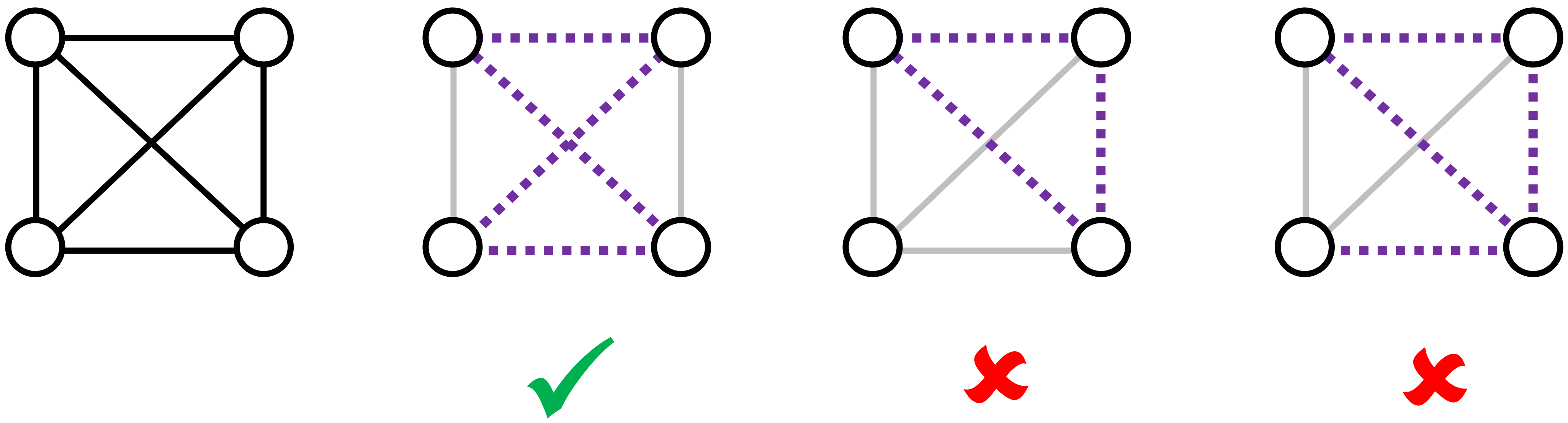


Hamiltonian Cycle Problem

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

∈ NP-Complete

Example:



TSP $\in$ NP-Complete

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?

TSP ∈ NP-Complete

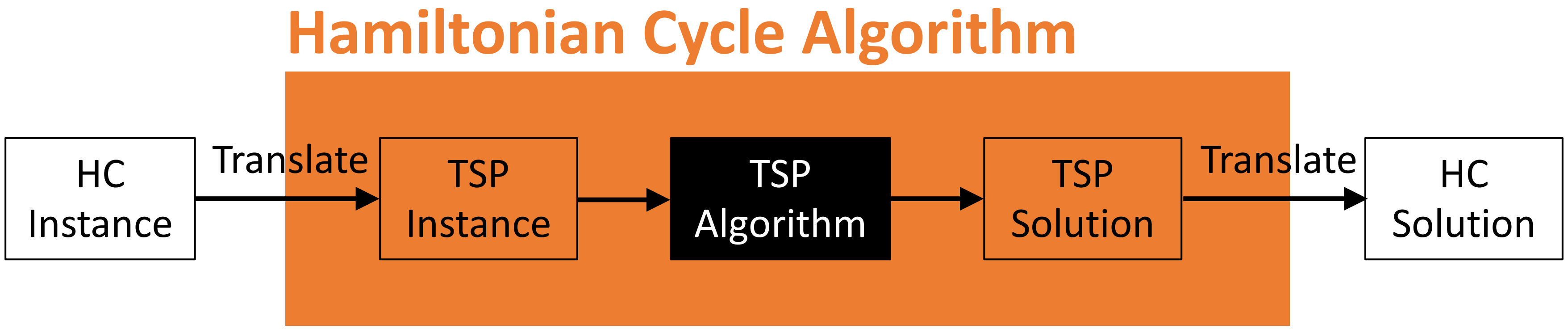
Hamiltonian Cycle: Given a **graph**, find a cycle that visits each vertex exactly once.

TSP: Given a list of cities and the **distances between each pair** of cities, what is the shortest possible route that visits each city once and returns to the origin city?

TSP ∈ NP-Complete

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

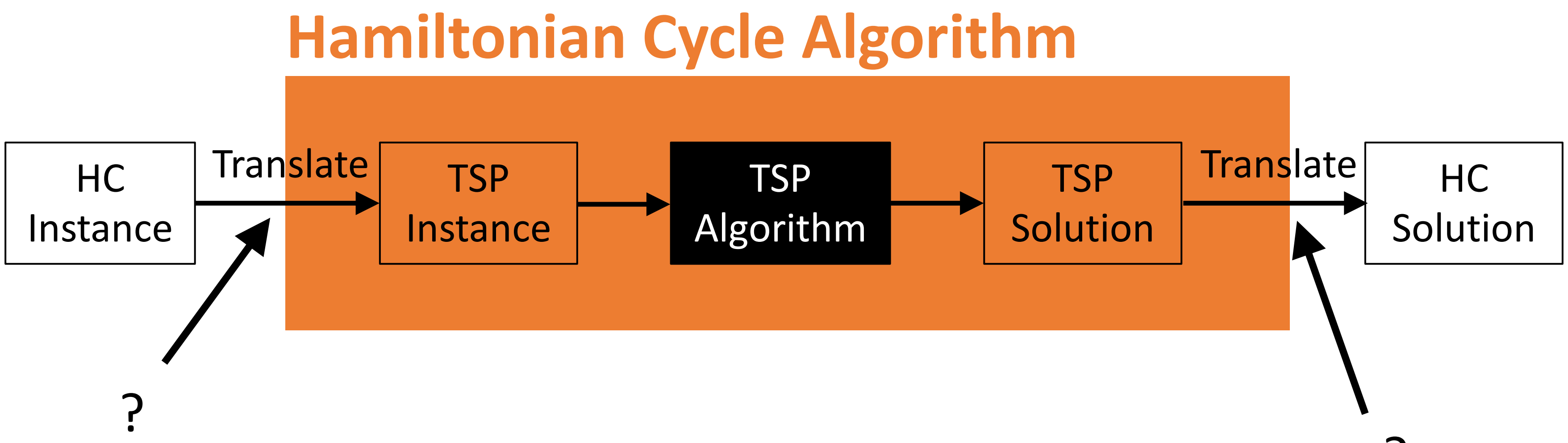
TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?



TSP ∈ NP-Complete

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

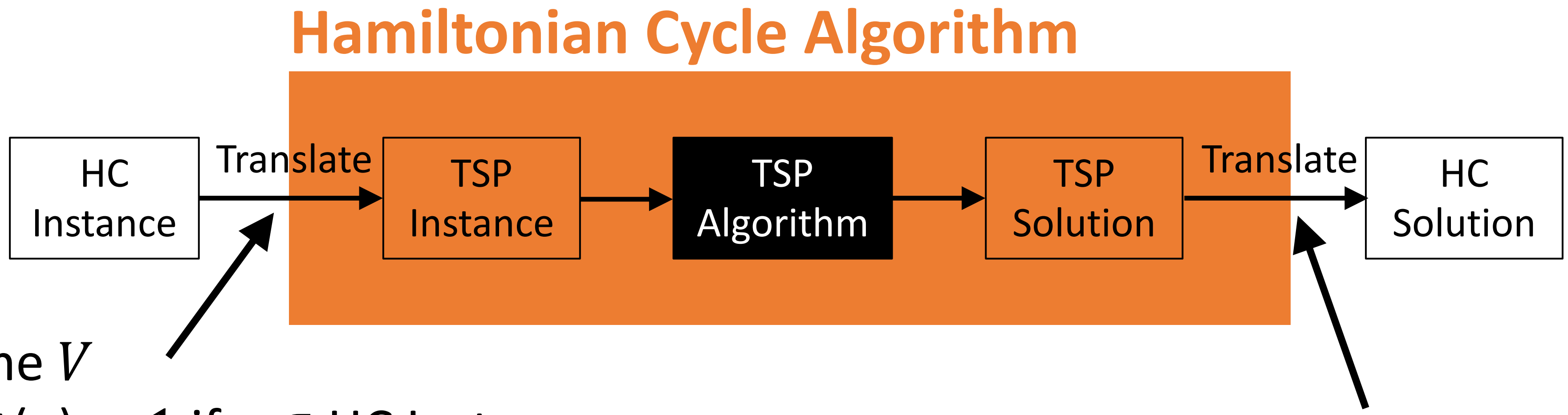
TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?



TSP ∈ NP-Complete

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?



Same V
$\text{cost}(e) = 1$ if $e \in$ HC Instance
$\text{cost}(e) = \infty$ if $e \notin$ HC Instance

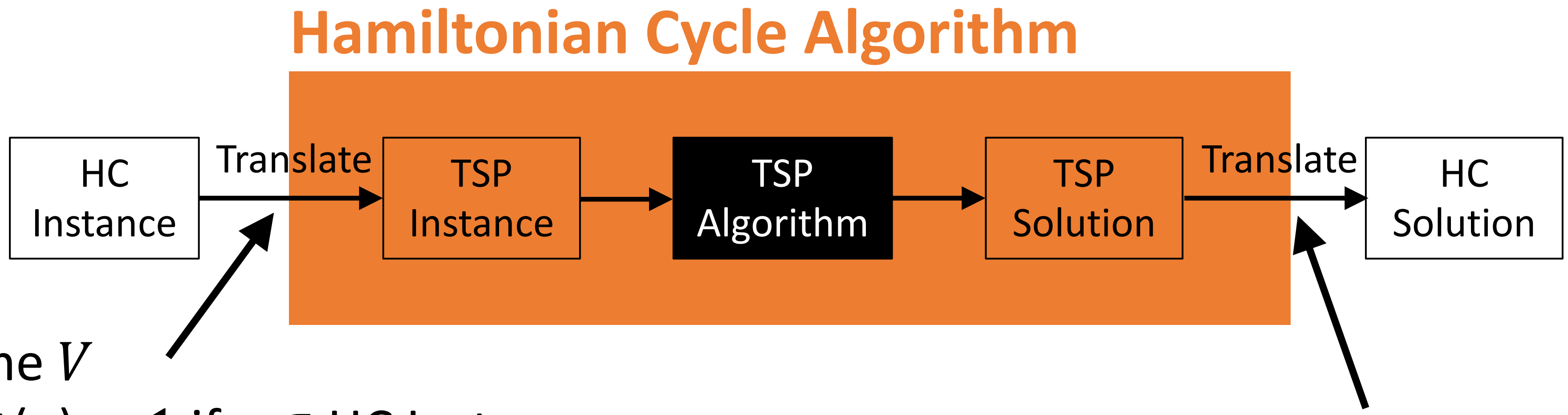
HC $\Longleftrightarrow$ TSP

Same selected edges

TSP ∈ NP-Complete

Hamiltonian Cycle: Given a graph, find a cycle that visits each vertex exactly once.

TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?



Same V
$\text{cost}(e) = 1$ if $e \in$ HC Instance
$\text{cost}(e) = 1 + \beta, \beta \neq 0$ if $e \notin$ HC Instance

Same selected edges

TSP Approximation Algorithm

Translation: cost$(e) = 1$ if $e \in$ HC

cost$(e) = 1 + \beta, \beta \neq 0$ if $e \notin$ HC

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} =$?

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if e ∉ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} =$?

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if e ∉ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if $e \in$ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\,|V|$ be input to TSP.

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if $e \in$ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\,|V|$ be input to TSP.

What happens when A runs on G', β...

If G has a Hamiltonian Cycle?

$\text{ALG}_A \leq$?

If G does not have a Hamiltonian Cycle?

$\text{ALG}_A \geq$?

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if $e \in$ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\,|V|$ be input to TSP.
What happens when A runs on G', β...
If G has a Hamiltonian Cycle?
$\text{ALG}_A \leq \alpha \text{ OPT} = \alpha\,|V|$
If G does not have a Hamiltonian Cycle?
$\text{ALG}_A \geq$?

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if e ∉ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\ |V|$ be input to TSP.

What happens when A runs on G', β...

If G has a Hamiltonian Cycle?

$\text{ALG}_A \leq \alpha\ \text{OPT} = \alpha\ |V|$

If G does not have a Hamiltonian Cycle?

$\text{ALG}_A \geq \text{OPT}$

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if e ∉ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\,|V|$ be input to TSP.
What happens when A runs on G', β...
If G has a Hamiltonian Cycle?
$\text{ALG}_A \leq \alpha\ \text{OPT} = \alpha\,|V|$
If G does not have a Hamiltonian Cycle?
$\text{ALG}_A \geq \text{OPT} \geq |V| + \beta$

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if $e \in$ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\,|V|$ be input to TSP.

What happens when A runs on G', β...

If G has a Hamiltonian Cycle?

$\text{ALG}_A \leq \alpha\ \text{OPT} = \alpha\,|V|$

If G does not have a Hamiltonian Cycle?

$\text{ALG}_A \geq \text{OPT} \geq |V| + \beta = |V| + \alpha\,|V|$

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if $e \in$ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if $e \notin$ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\,|V|$ be input to TSP.

What happens when A runs on G', β...

If G has a Hamiltonian Cycle?

$\text{ALG}_A \leq \alpha\ \text{OPT} = \alpha\,|V|$

If G does not have a Hamiltonian Cycle?

$\text{ALG}_A \geq \text{OPT} \geq |V| + \beta = |V| + \alpha\,|V| = (1 + \alpha)\,|V|$

TSP Approximation Algorithm

Translation: cost(e) $= 1$ if e ∈ HC
cost(e) $= 1 + \beta, \beta \neq 0$ if e ∉ HC

If $G = (V, E)$ has a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} = |V|$
If $G = (V, E)$ does not have a Hamiltonian Cycle, $\text{OPT}_{\text{TSP}} \geq |V| + \beta$

Let A be an α-approximation algorithm for TSP (i.e. ALG $\leq \alpha$ OPT)
Let $G = (V, E)$ be input to Hamiltonian Cycle.
Let $G', \beta = \alpha\, |V|$ be input to TSP.

What happens when A runs on G', β...

If G has a Hamiltonian Cycle?

$\text{ALG}_A \leq \alpha\ \text{OPT} = \alpha\, |V|$

If G does not have a Hamiltonian Cycle?

$\text{ALG}_A \geq \text{OPT} \geq |V| + \beta = |V| + \alpha\, |V| = (1 + \alpha)\, |V| > \alpha\, |V|$

TSP Approximation Algorithm

Let $G', \beta = \alpha\,|V|$ be input to TSP and run A:

If G has a Hamiltonian Cycle,

$\text{ALG}_A \leq \alpha\,|V|$

If G does not have a Hamiltonian Cycle,

$\text{ALG}_A > \alpha\,|V|$

TSP Approximation Algorithm

Let $G', \beta = \alpha\,|V|$ be input to TSP and run A:

If G has a Hamiltonian Cycle,

$\text{ALG}_A \leq \alpha\,|V|$

If G does not have a Hamiltonian Cycle,

$\text{ALG}_A > \alpha\,|V|$

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α|V| and run A on G′,β
```

TSP Approximation Algorithm

Let $G', \beta = \alpha\,|V|$ be input to TSP and run A:

If G has a Hamiltonian Cycle,

$\text{ALG}_A \leq \alpha\,|V|$

If G does not have a Hamiltonian Cycle,

$\text{ALG}_A > \alpha\,|V|$

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α |V| and run A on G′,β
   if ALG_A ≤ α |V|
      return cycle found
   else
      return false
```

TSP Approximation Algorithm

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α |V| and run A on G′, β
   if ALG_A ≤ α |V|
      return cycle found
   else
      return false
```

Is this a problem?

TSP Approximation Algorithm

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α|V| and run A on G′,β
   if ALG_A ≤ α|V|
      return cycle found
   else
      return false
```

Is this a problem?

Yes! Any approximation algorithm for TSP will solve the NP-Complete Hamiltonian Cycle problem!

TSP Approximation Algorithm

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α |V| and run A on G′, β
   if ALG_A ≤ α |V|
      return cycle found
   else
      return false
```

Is this a problem?

Yes! Any approximation algorithm for TSP will solve the NP-Complete Hamiltonian Cycle problem!

$\therefore \nexists$ poly time approx alg for TSP, unless P = NP

TSP Approximation Algorithm

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α|V| and run A on G′,β
   if ALG_A ≤ α|V|
      return cycle found
   else
      return false
```

Why is this the case?

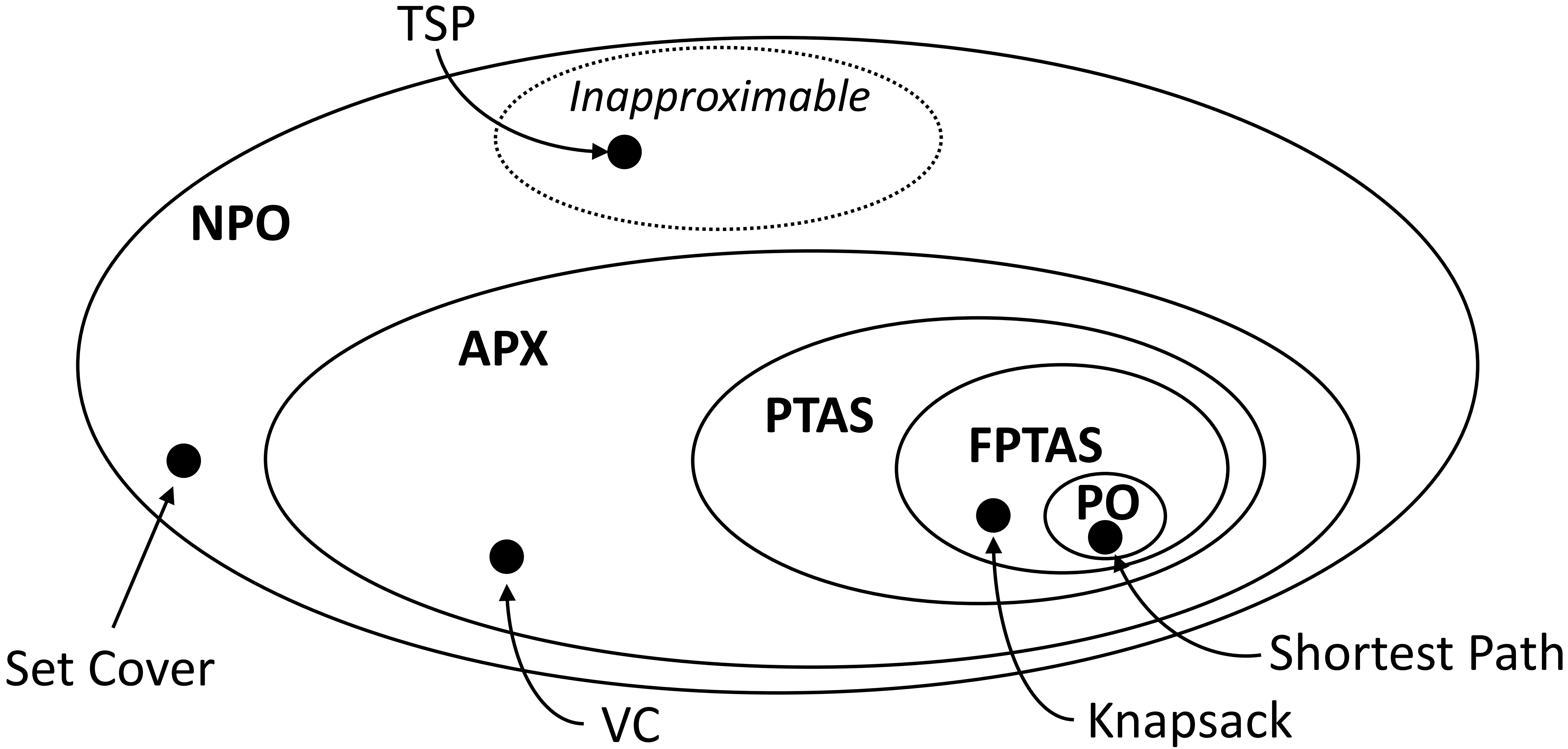
TSP Approximation Algorithm

```
HamiltonianCycleFinder(G)
   Let A be a TSP α-approximation algorithm
   Let β = α|V| and run A on G′,β
   if ALG_A ≤ α|V|
      return cycle found
   else
      return false
```

Why is this the case?

Any approx alg for TSP is required to be valid, and a valid output is a Hamiltonian Cycle.

Approximability Hierarchy



Integer Linear Programming

Suppose one could write an ILP for TSP:

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise} \end{cases}$$

For $i = 1, \ldots, n$, let u_i be a dummy variable, and finally take c_{ij} to be the distance from city i to city j.

$$\begin{aligned}
\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} & c_{ij} x_{ij} \colon && \\
& x_{ij} \in \{0,1\} && i,j = 1,\ldots,n; \\
& u_i \in \mathbf{Z} && i = 2,\ldots,n; \\
& \sum_{i=1, i \neq j}^{n} x_{ij} = 1 && j = 1,\ldots,n; \\
& \sum_{j=1, j \neq i}^{n} x_{ij} = 1 && i = 1,\ldots,n; \\
& u_i - u_j + n x_{ij} \leq n - 1 && 2 \leq i \neq j \leq n; \\
& 0 \leq u_i \leq n - 1 && 2 \leq i \leq n.
\end{aligned}$$

From: Wikipedia

Integer Linear Programming

Suppose one could write an ILP for TSP:

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise} \end{cases}$$

For *i* = 1, …, *n*, let u_i be a dummy variable, and finally take c_{ij} to be the distance from city *i* to city *j*.

$$\begin{aligned}
\min \sum_{i=1}^{n} \sum_{j\neq i, j=1}^{n} c_{ij} x_{ij} &\colon \\
x_{ij} \in \{0,1\} & \qquad i,j = 1,\ldots,n; \\
u_i \in \mathbf{Z} & \qquad i = 2,\ldots,n; \\
\sum_{i=1, i\neq j}^{n} x_{ij} = 1 & \qquad j = 1,\ldots,n; \\
\sum_{j=1, j\neq i}^{n} x_{ij} = 1 & \qquad i = 1,\ldots,n; \\
u_i - u_j + n x_{ij} \leq n-1 & \qquad 2 \leq i \neq j \leq n; \\
0 \leq u_i \leq n-1 & \qquad 2 \leq i \leq n.
\end{aligned}$$

From: Wikipedia

What could you conclude?

Integer Linear Programming

Suppose one could write an ILP for TSP:

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise} \end{cases}$$

For $i = 1, \ldots, n$, let u_i be a dummy variable, and finally take c_{ij} to be the distance from city *i* to city *j*.

$$\begin{aligned}
\min \sum_{i=1}^{n} \sum_{j\neq i, j=1}^{n} c_{ij} x_{ij}&\colon && \\
x_{ij} \in \{0,1\} & && i,j = 1,\ldots,n; \\
u_i \in \mathbf{Z} & && i = 2,\ldots,n; \\
\sum_{i=1, i\neq j}^{n} x_{ij} = 1 & && j = 1,\ldots,n; \\
\sum_{j=1, j\neq i}^{n} x_{ij} = 1 & && i = 1,\ldots,n; \\
u_i - u_j + n x_{ij} \leq n-1 & && 2 \leq i \neq j \leq n; \\
0 \leq u_i \leq n-1 & && 2 \leq i \leq n.
\end{aligned}$$

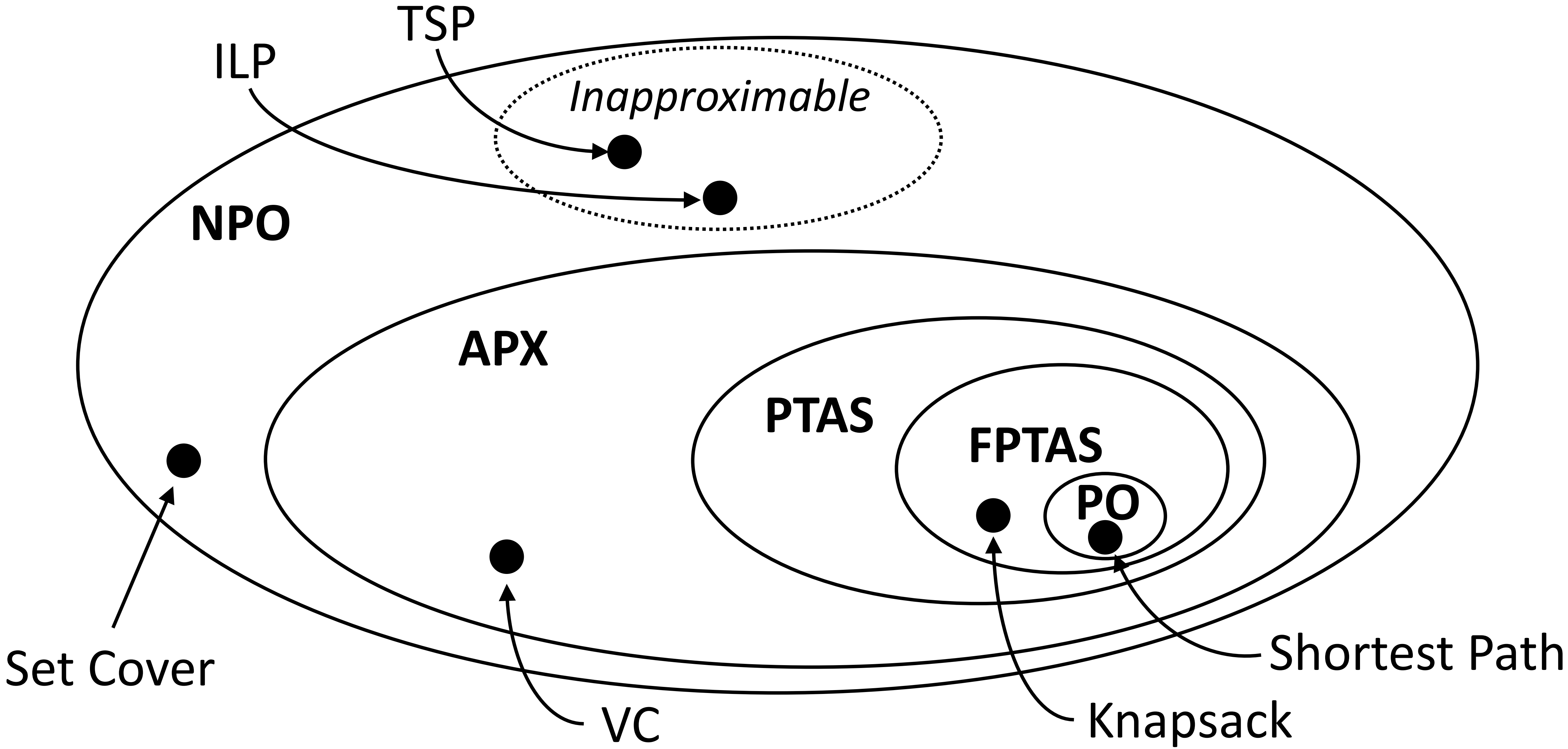
From: Wikipedia

What could you conclude?

If TSP in inapproximable, solving ILPs must be too!

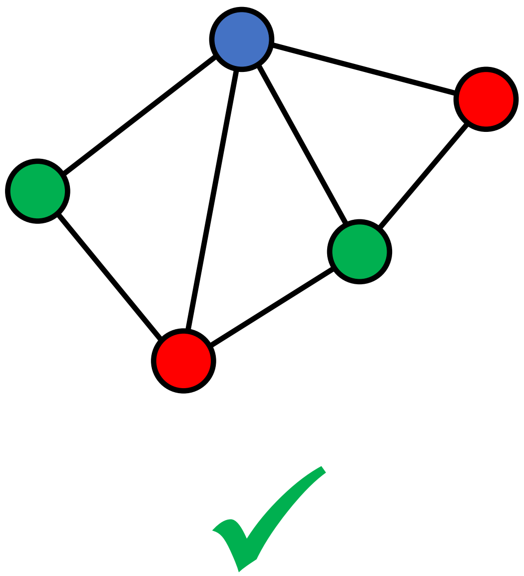
(Otherwise an approximation algorithm for ILPs would provide an approximation algorithm for TSP.)

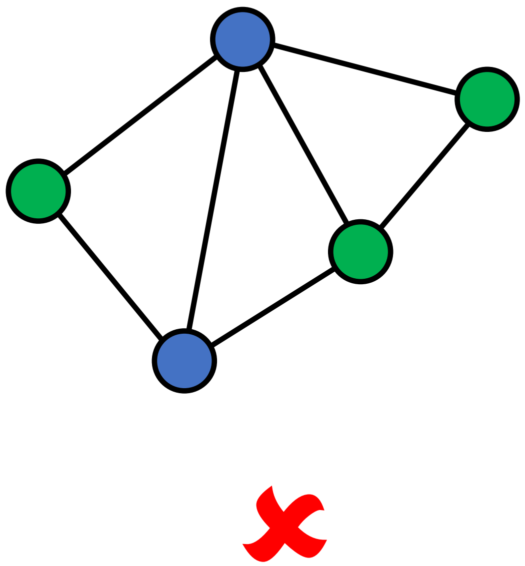
Approximability Hierarchy



Graph Coloring

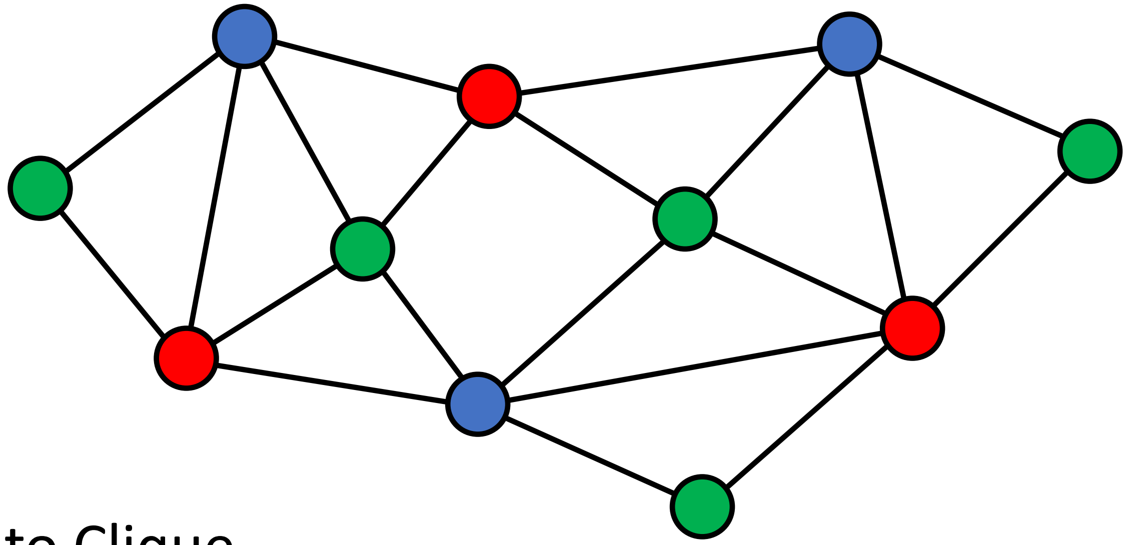
Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.





Graph Coloring

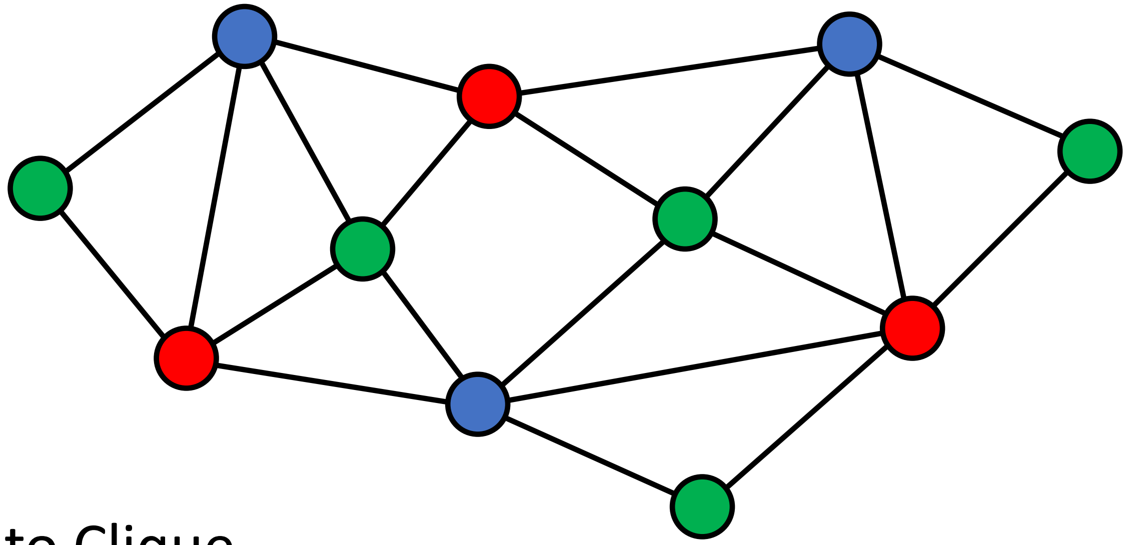
Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.



- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

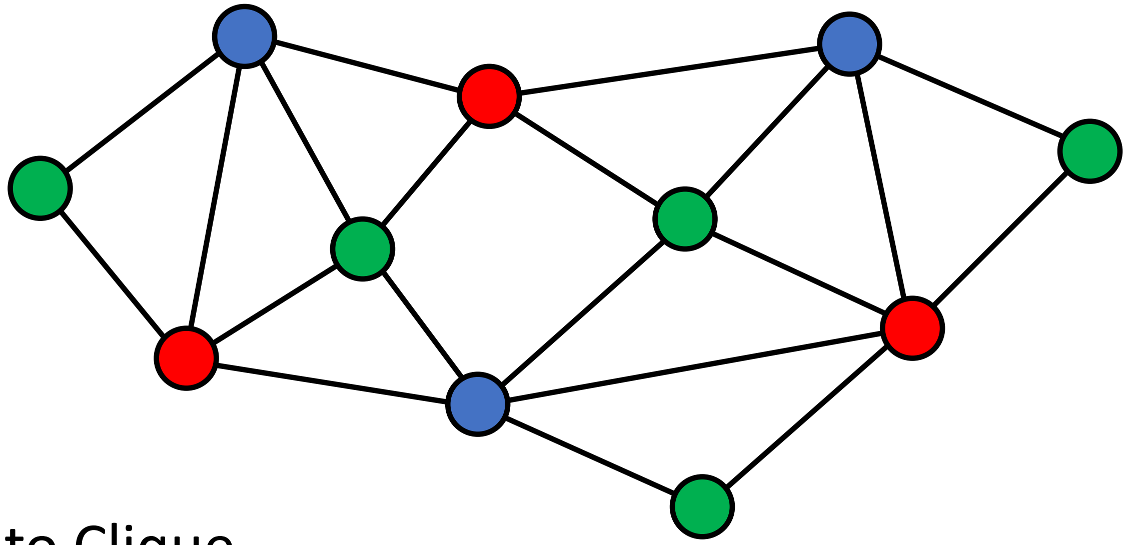


n	$\chi(n)$
0	?

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

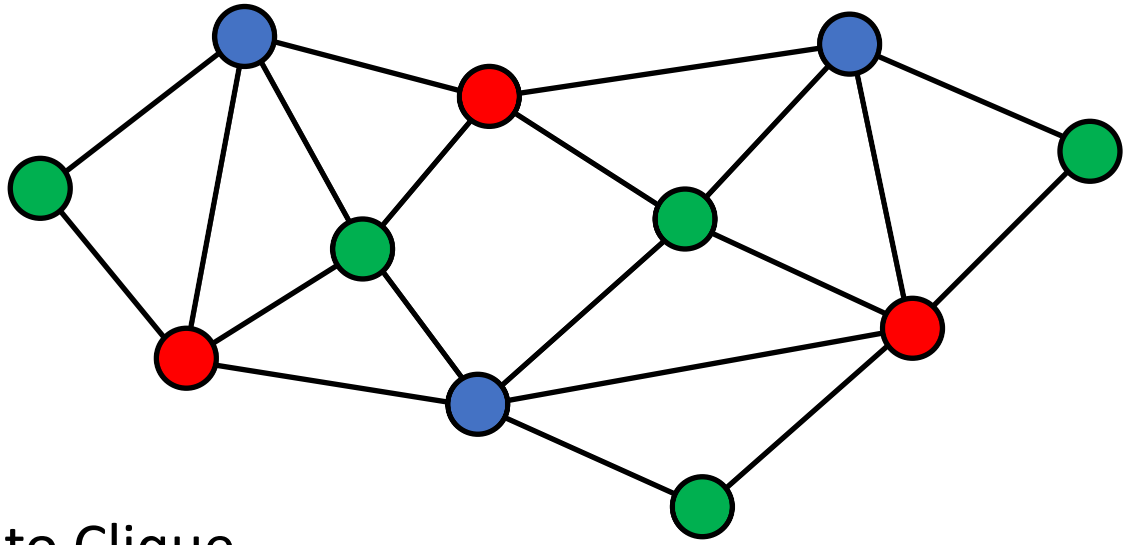


n	$\chi(n)$
0	✓

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

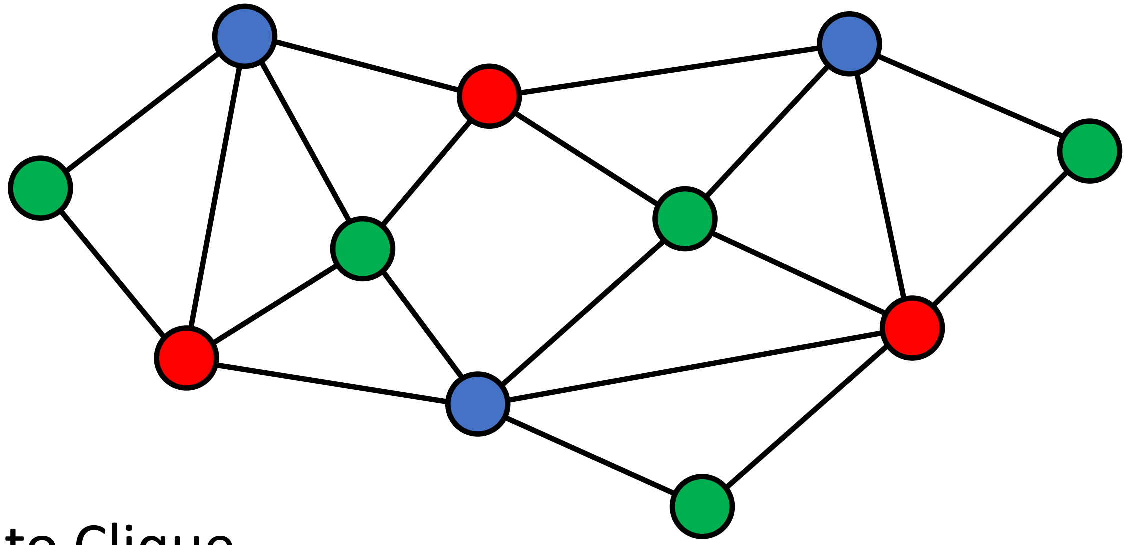


n	$\chi(n)$
0	✓
1	?

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

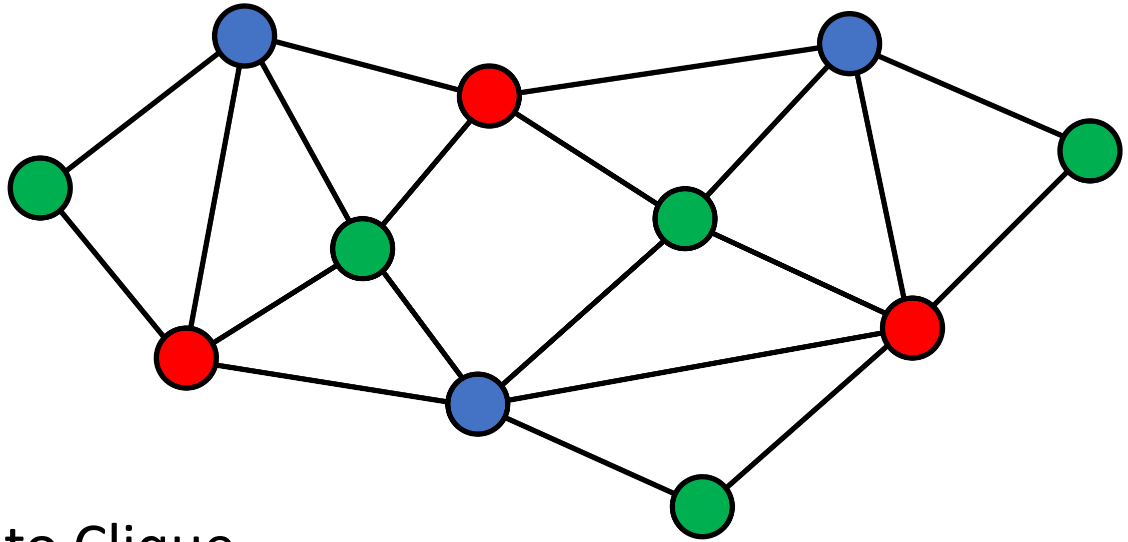


n	$\chi(n)$
0	✓
1	✓

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

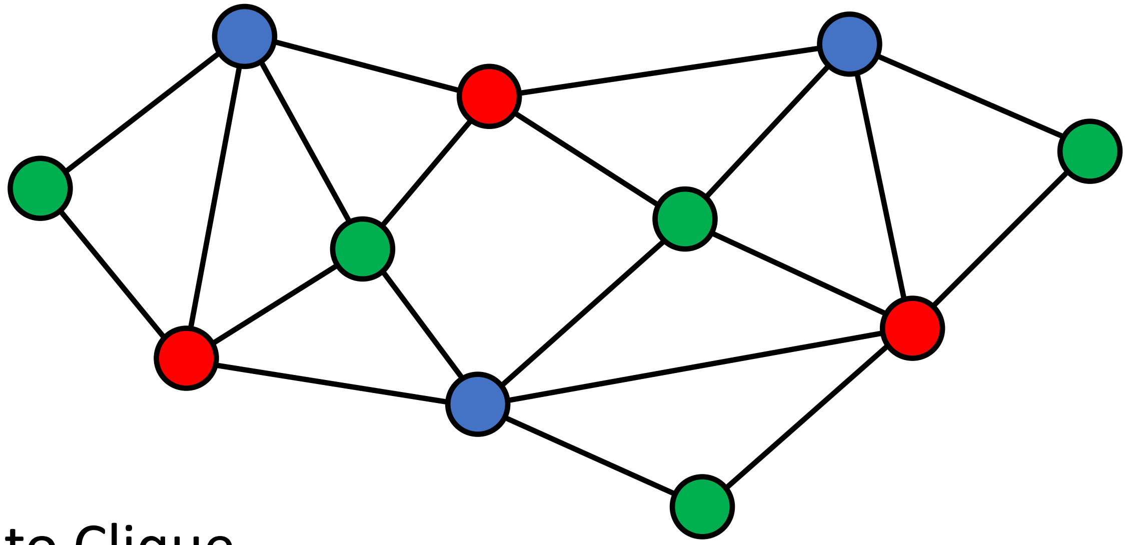


n	$\chi(n)$
0	✓
1	✓
2	?

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

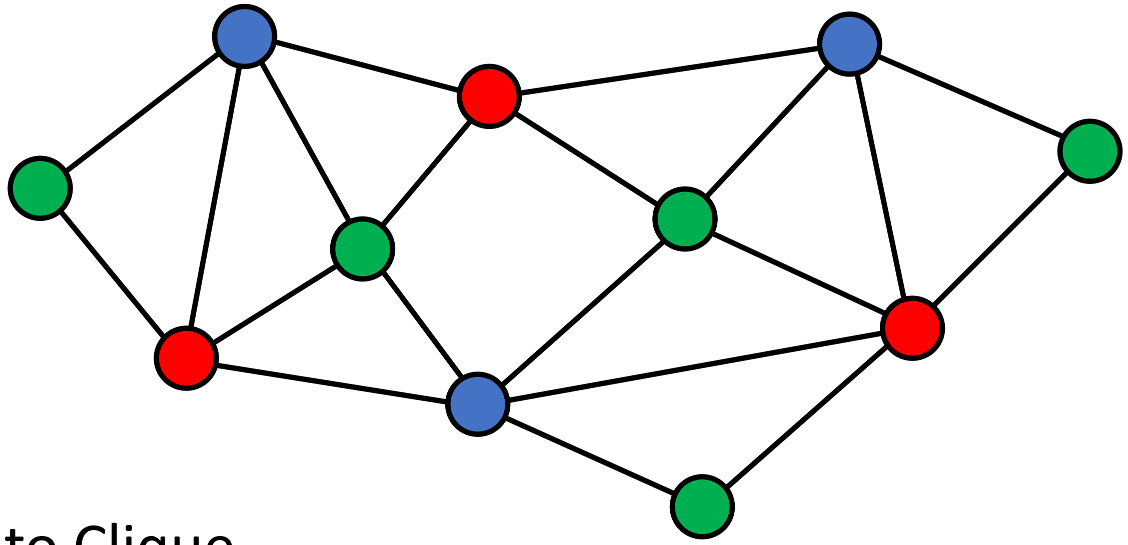


n	$\chi(n)$
0	✓
1	✓
2	✓

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

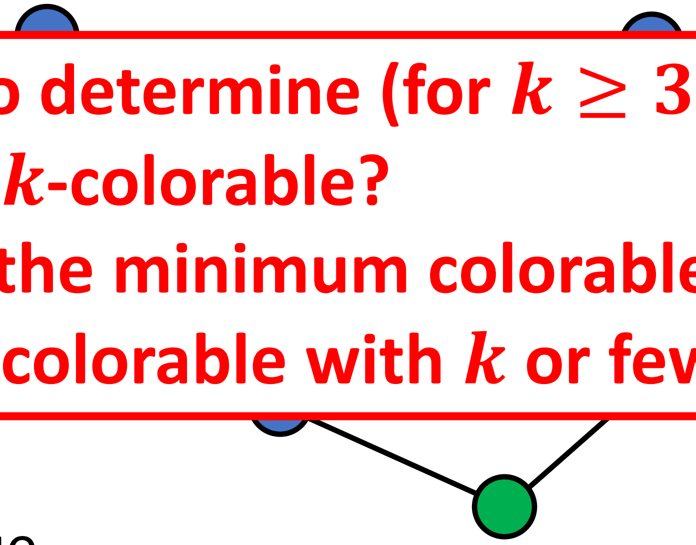


n	$\chi(n)$
0	✓
1	✓
2	✓
3	✗
4	✗
⋮	⋮

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.



NP-Hard to determine (for $k \geq 3$):
- **Is graph k-colorable?**
- **What is the minimum colorable value?**
- **Is graph colorable with k or fewer colors?**

n	$\chi(n)$
0	✓
1	✓
2	✓
3	✗
4	✗
⋮	⋮

- Related to Clique.
- Influenced by vertex degree.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for the problem.

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for the problem.

What happens when we run it on a graph with $\chi(G) = 3$?

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for the problem.

What happens when we run it on a graph with $\chi(G) = 3$?

$$ALG \leq \left(\frac{4}{3} - \varepsilon\right) OPT$$

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3}-\varepsilon\right)$-approximation algorithm for the problem.

What happens when we run it on a graph with $\chi(G) = 3$?

$$ALG \leq \left(\frac{4}{3}-\varepsilon\right) OPT = \left(\frac{4}{3}-\varepsilon\right) 3 = 4 - 3\varepsilon$$

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for the problem.

What happens when we run it on a graph with $\chi(G) = 3$?

$$ALG \leq \left(\frac{4}{3} - \varepsilon\right) OPT = \left(\frac{4}{3} - \varepsilon\right) 3 = 4 - 3\varepsilon < 4$$

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for the problem.

What happens when we run it on a graph with $\chi(G) = 3$?

$$ALG \leq \left(\frac{4}{3} - \varepsilon\right) OPT = \left(\frac{4}{3} - \varepsilon\right) 3 = 4 - 3\varepsilon < 4 \Rightarrow ALG \leq 3$$

Graph Coloring

Minimum Graph Coloring: Given graph, find the smallest number of colors such that each vertex can be assigned a color and neighboring vertices do not share colors.

Suppose we had a $\left(\frac{4}{3} - \varepsilon\right)$-approximation algorithm for the problem.

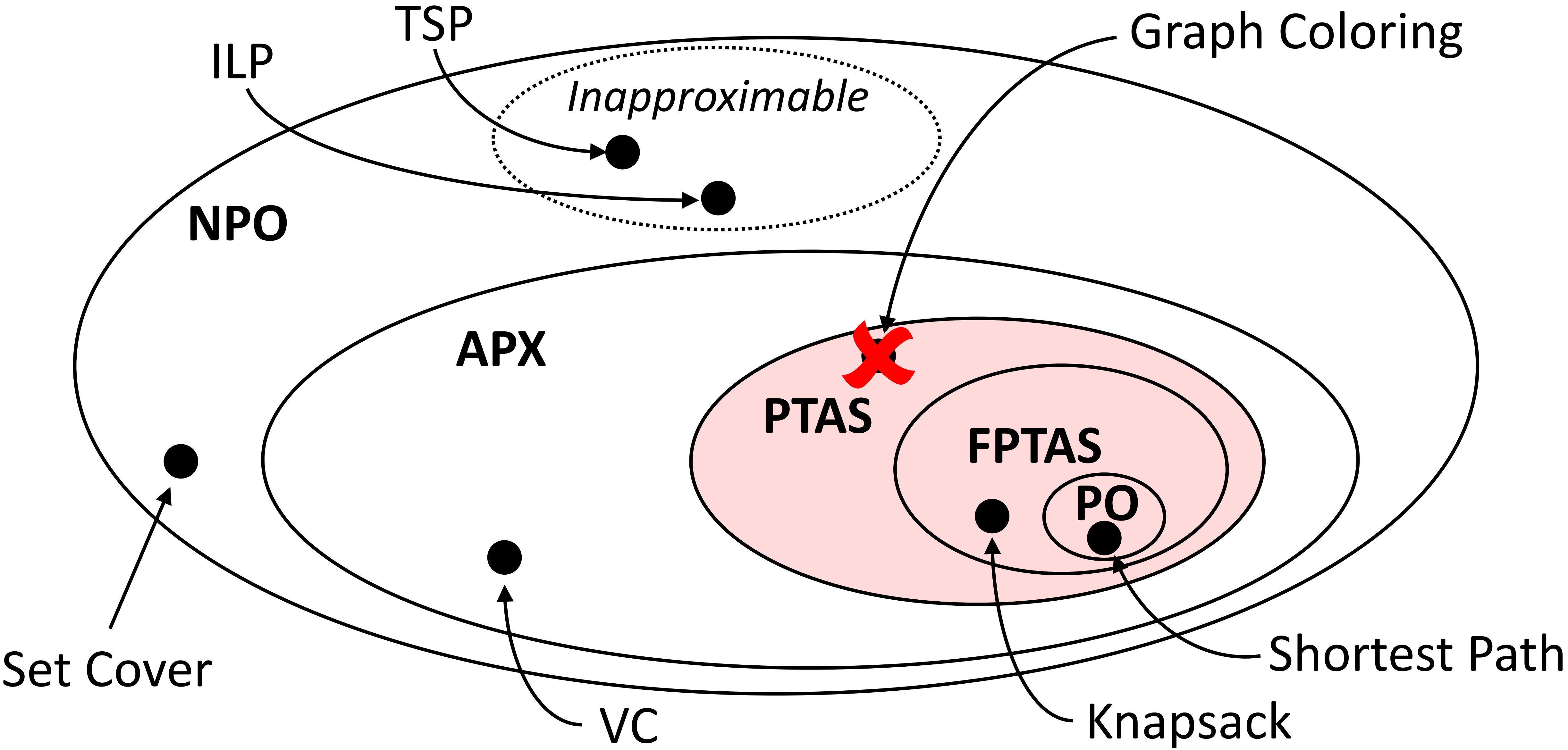
What happens when we run it on a graph with $\chi(G) = 3$?

$$ALG \leq \left(\frac{4}{3} - \varepsilon\right) OPT = \left(\frac{4}{3} - \varepsilon\right)\ 3 = 4 - 3\varepsilon < 4 \Rightarrow ALG \leq 3$$

Thus, we could determine if a graph is 3-colorable

$\Rightarrow$ algorithm can't exist.

Approximability Hierarchy



Special Case - Metric TSP

TSP: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?

Special Case - Metric TSP

TSP: Given a list of cities and the distances between each pair of cities **(satisfying the triangle inequality)**, what is the shortest possible route that visits each city once and returns to the origin city?

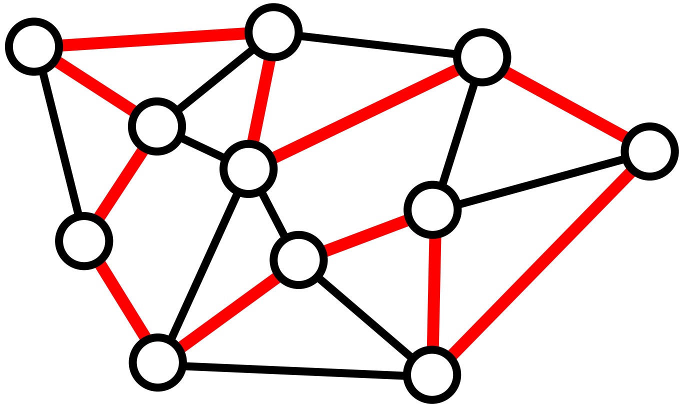
$$\text{dist}(u, v) \leq \text{dist}(u, w) + \text{dist}(w, v)$$

Special Case - Metric TSP

Find some structure that is:

1. Easy to compute.
2. Related to TSP.
3. Lower bound on OPT.

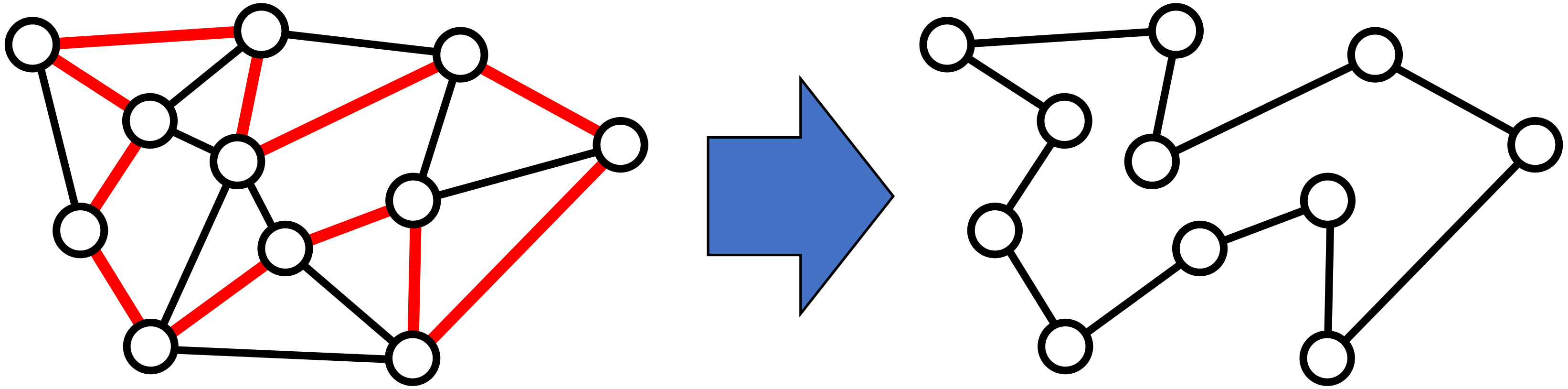
Special Case - Metric TSP



Find some structure that is:

1. Easy to compute.
2. Related to TSP.
3. Lower bound on OPT.

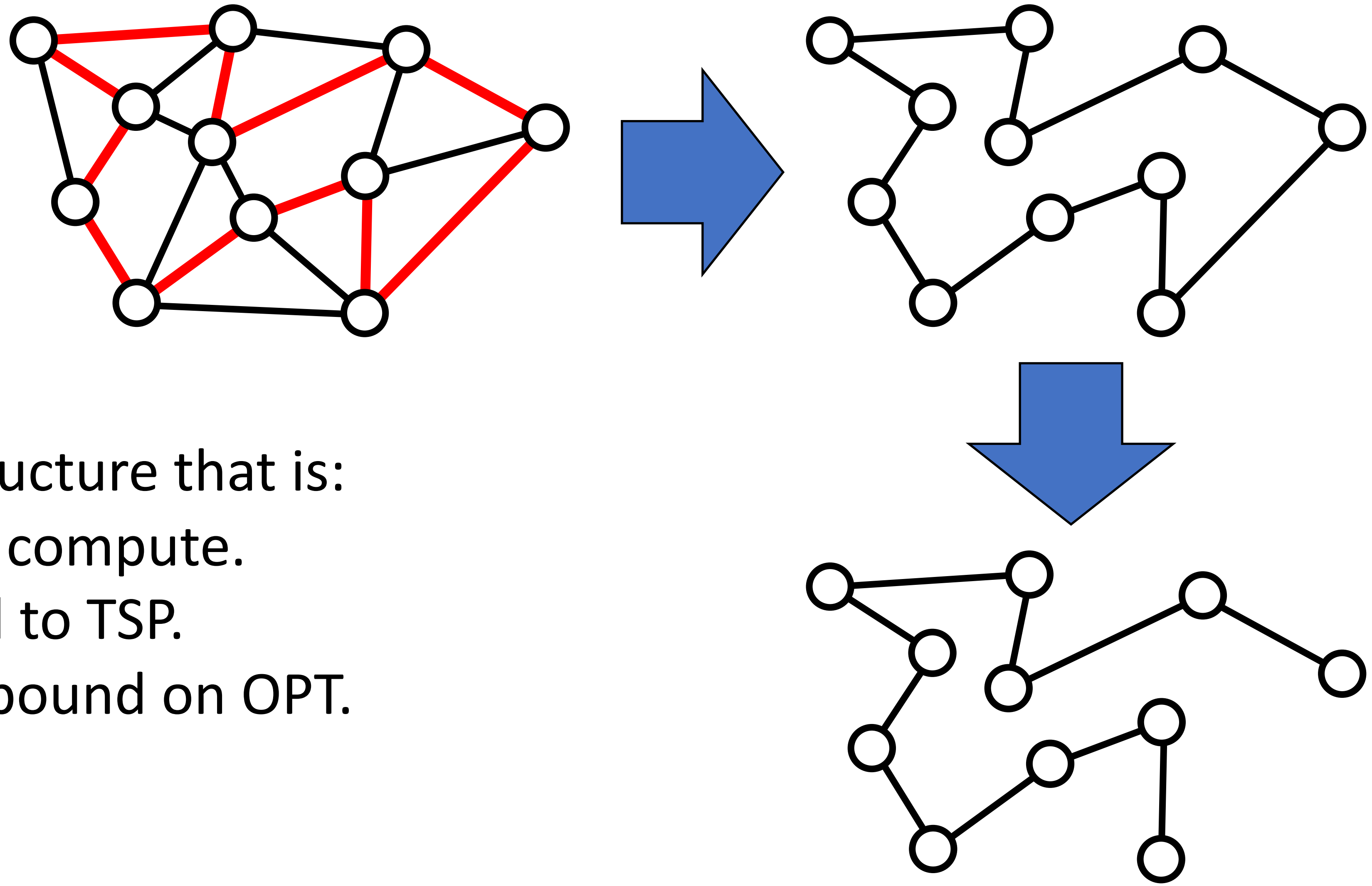
Special Case - Metric TSP



Find some structure that is:

1. Easy to compute.
2. Related to TSP.
3. Lower bound on OPT.

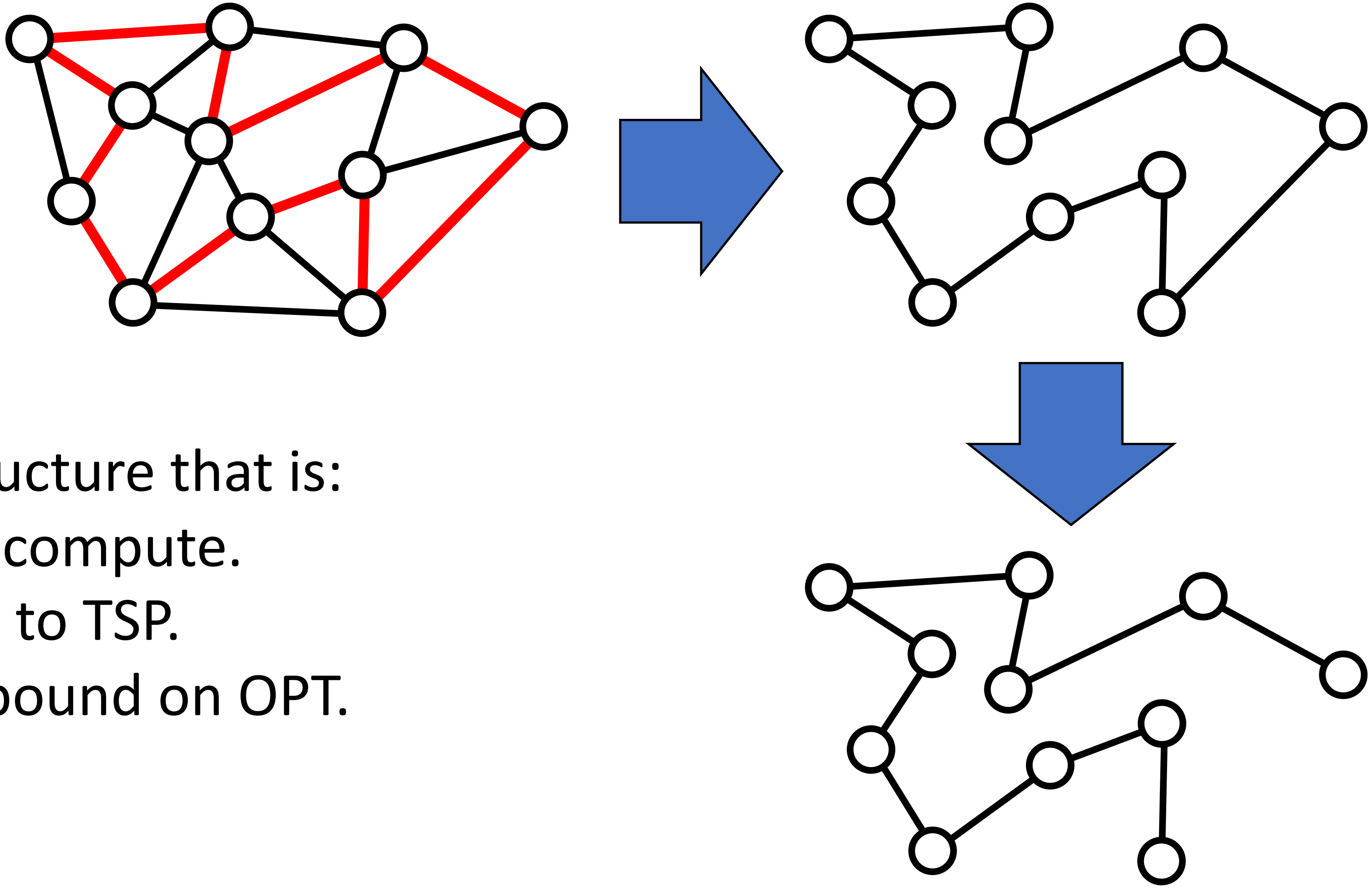
Special Case - Metric TSP



Find some structure that is:

1. Easy to compute.
2. Related to TSP.
3. Lower bound on OPT.

Special Case - Metric TSP

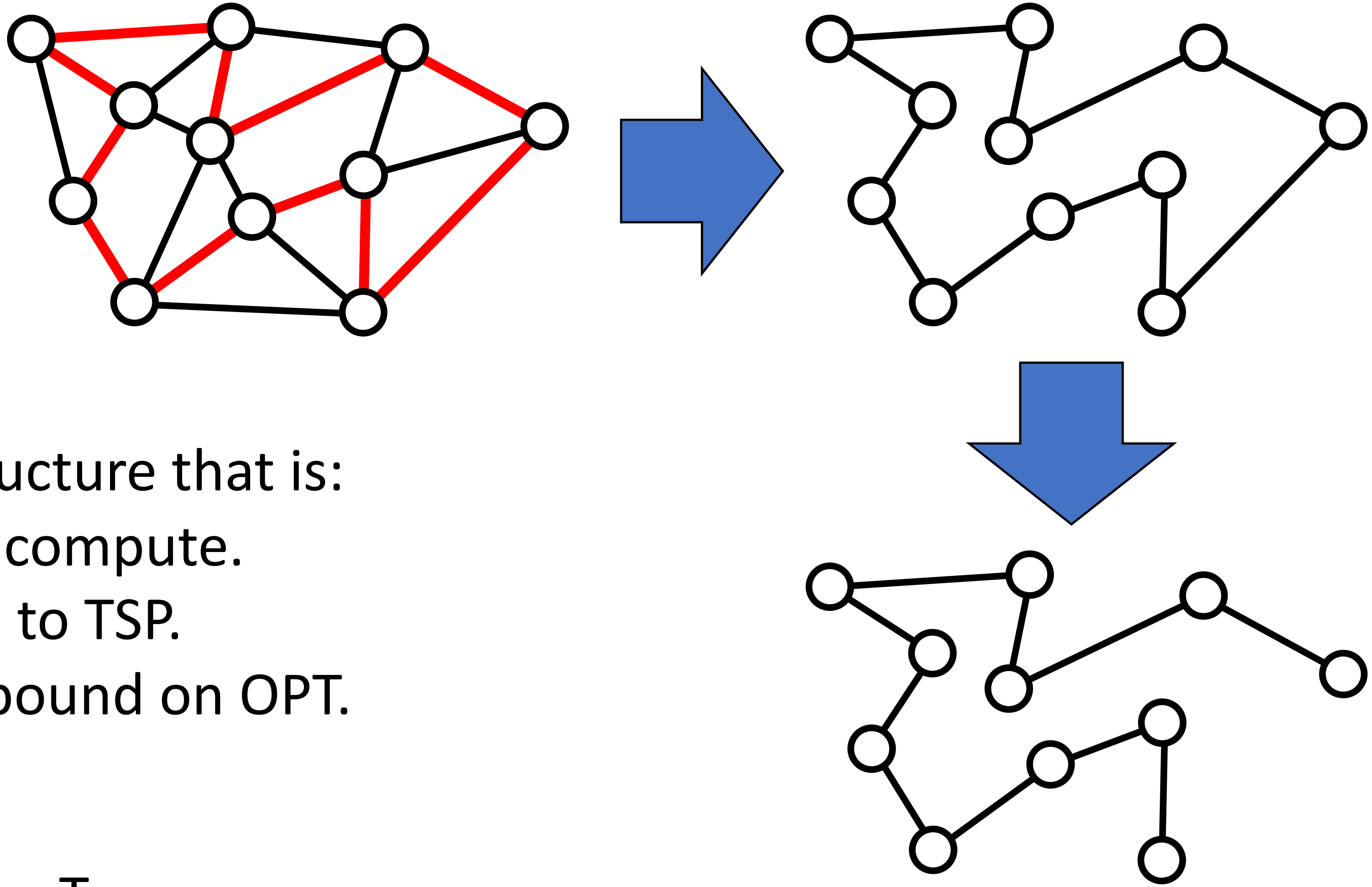


Find some structure that is:

1. Easy to compute.
2. Related to TSP.
3. Lower bound on OPT.

What is this?

Special Case - Metric TSP



Find some structure that is:

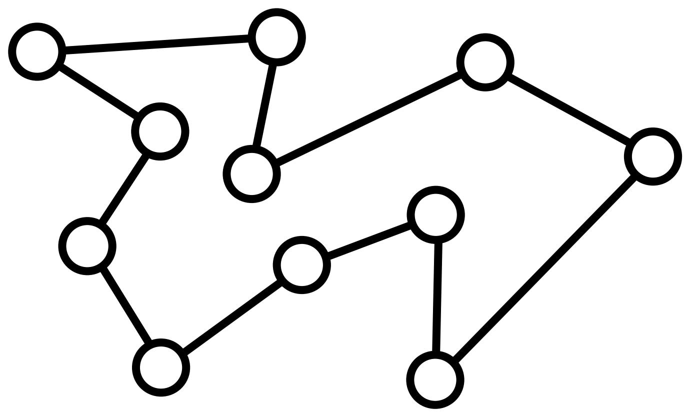
1. Easy to compute.
2. Related to TSP.
3. Lower bound on OPT.

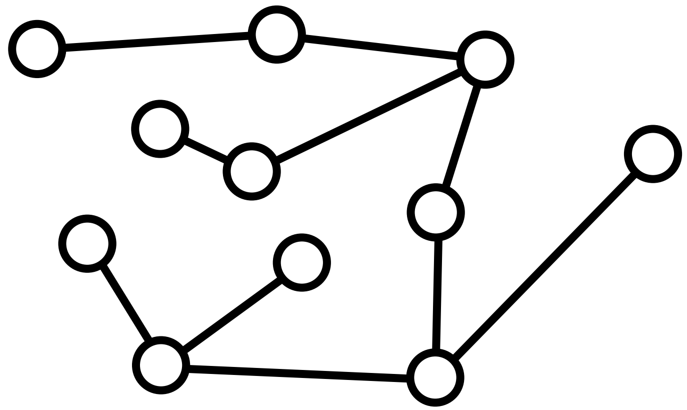
What is this?

Spanning Tree

Special Case - Metric TSP

Relationship between OPT and cost of MST?



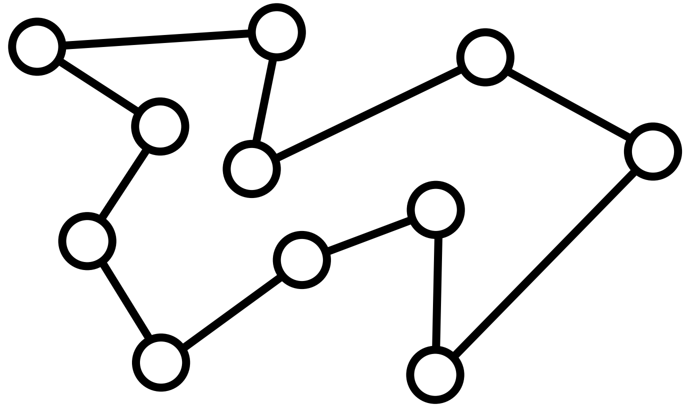


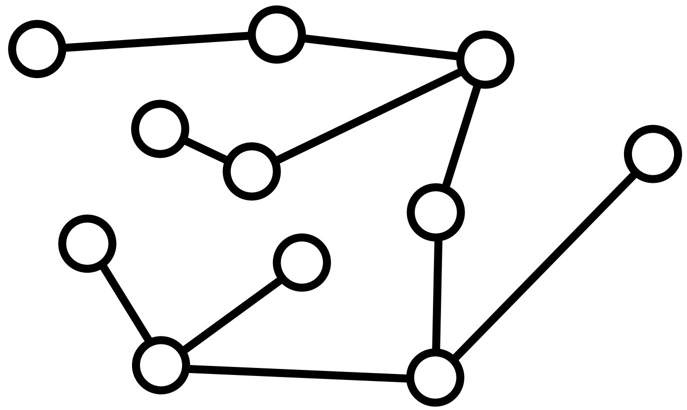
Special Case - Metric TSP

Relationship between OPT and cost of MST?

$\text{OPT} \geq \text{cost(MST)}$

How to turn MST into tour of cities?





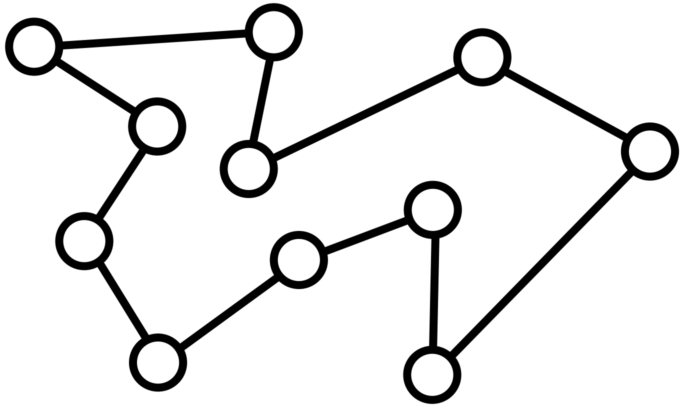
Special Case - Metric TSP

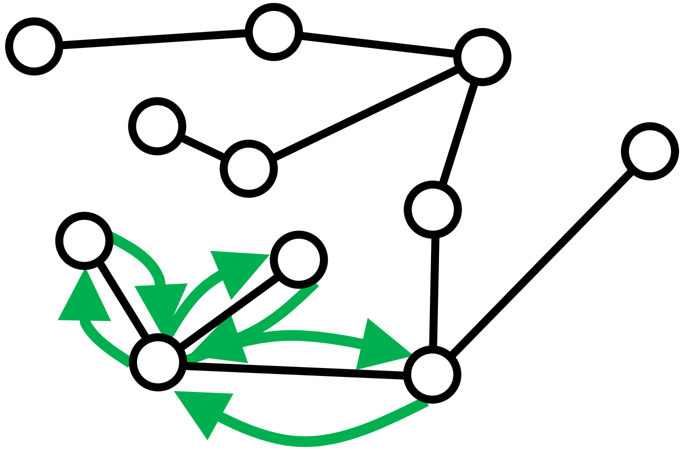
Relationship between OPT and cost of MST?

$$\mathrm{OPT} \geq \mathrm{cost(MST)}$$

How to turn MST into tour of cities?

What is the cost of this tour?





Special Case - Metric TSP

Relationship between OPT and cost of MST?

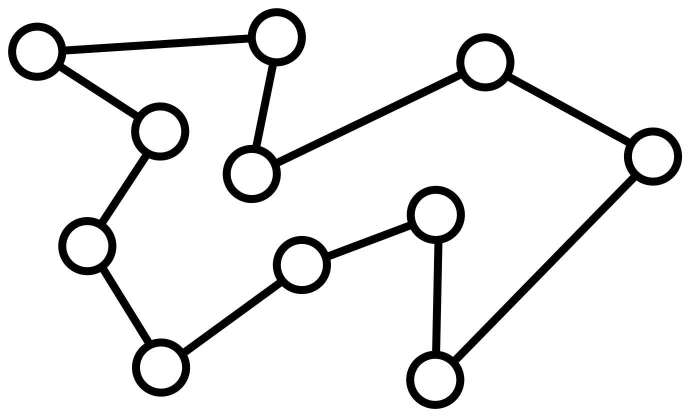
$$\text{OPT} \geq \text{cost}(\text{MST})$$

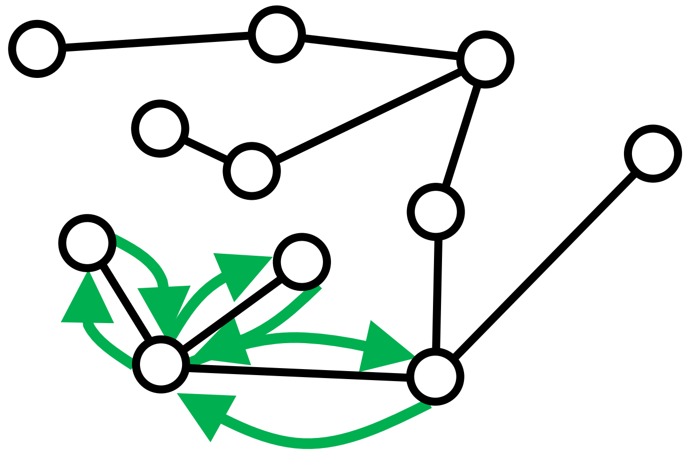
How to turn MST into tour of cities?

What is the cost of this tour?

$$\text{ALG} = 2\ \text{cost}(\text{MST})$$

Relationship between ALG and OPT?





Special Case - Metric TSP

Relationship between OPT and cost of MST?

$$\text{OPT} \geq \text{cost(MST)}$$

How to turn MST into tour of cities?

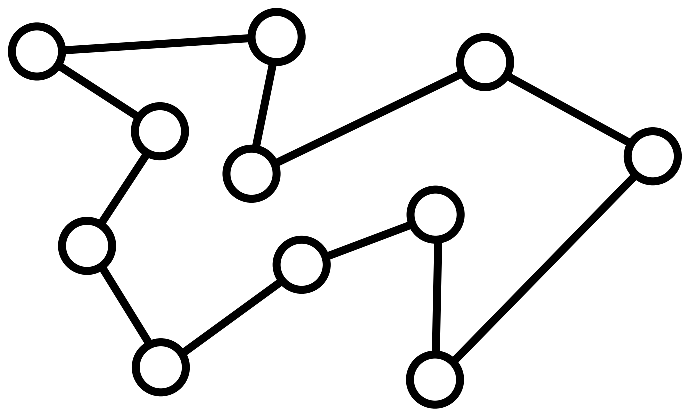
What is the cost of this tour?

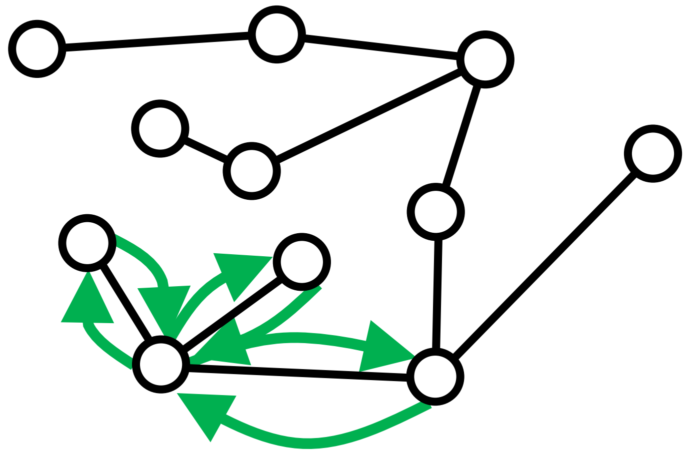
$$\text{ALG} = 2\ \text{cost(MST)}$$

Relationship between ALG and OPT?

$$\text{ALG} = 2\ \text{cost(MST)} \leq 2\ \text{OPT}$$

Any problems?





Special Case - Metric TSP

Relationship between OPT and cost of MST?

$$\text{OPT} \geq \text{cost(MST)}$$

How to turn MST into tour of cities?

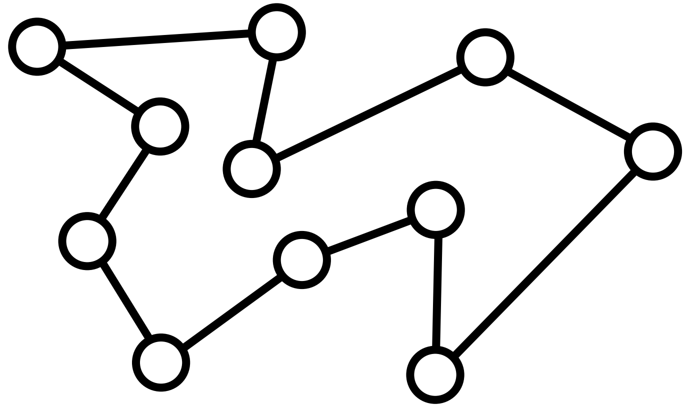
What is the cost of this tour?

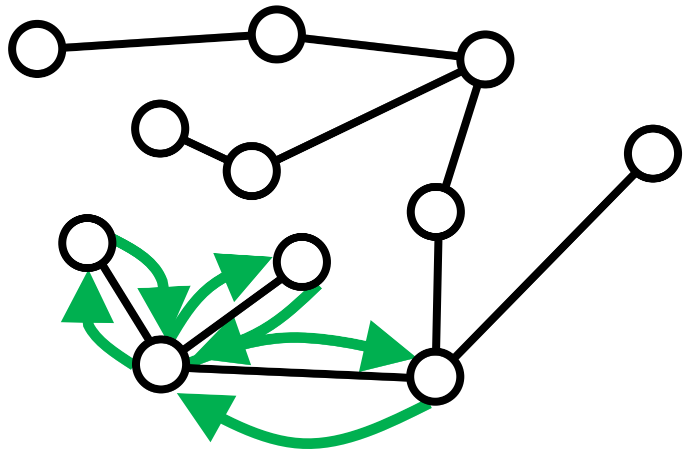
$$\text{ALG} = 2\ \text{cost(MST)}$$

Relationship between ALG and OPT?

$$\text{ALG} = 2\ \text{cost(MST)} \leq 2\ \text{OPT}$$

How can we eliminate double visits (without messing up the cost)?





Special Case - Metric TSP

Relationship between OPT and cost of MST?

$\mathrm{OPT} \geq \mathrm{cost}(\mathrm{MST})$

How to turn MST into tour of cities?

What is the cost of this tour?

$\mathrm{ALG} = 2\ \mathrm{cost}(\mathrm{MST})$

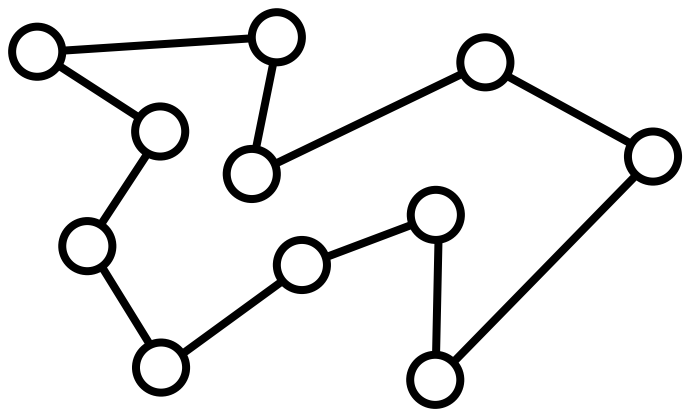
Relationship between ALG and OPT?

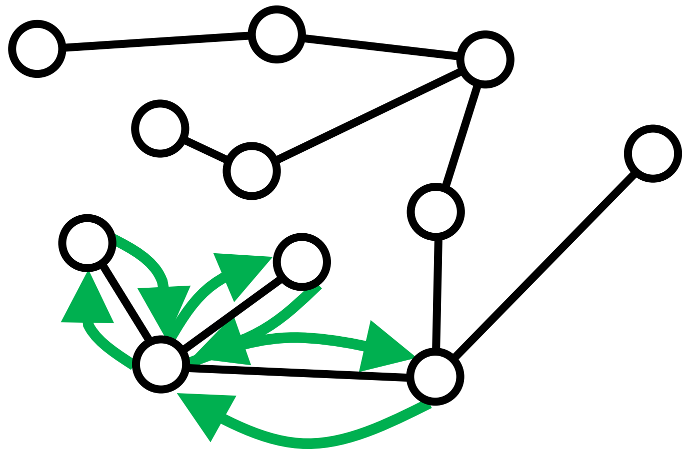
$\mathrm{ALG} = 2\ \mathrm{cost}(\mathrm{MST}) \leq 2\ \mathrm{OPT}$

How can we eliminate double visits (without messing up the cost)?

Skip to next unvisited vertex. Can only decrease cost (triangle inequality).

$\mathrm{dist}(u, v) \leq \mathrm{dist}(u, w) + \mathrm{dist}(w, v)$





Approximability Hierarchy