Algorithms for Unmanned Search and Rescue Operations

Winter School on TCS, IISc Bangalore

Prahlad Narasimhan Kasthurirangan

December 10, 2024

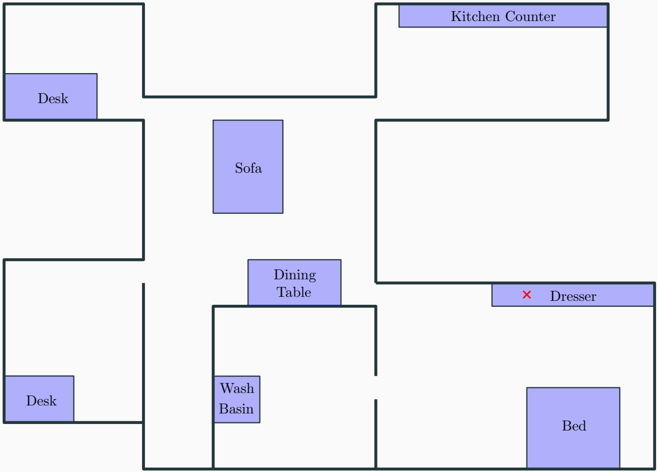
Stony Brook University

Motivation

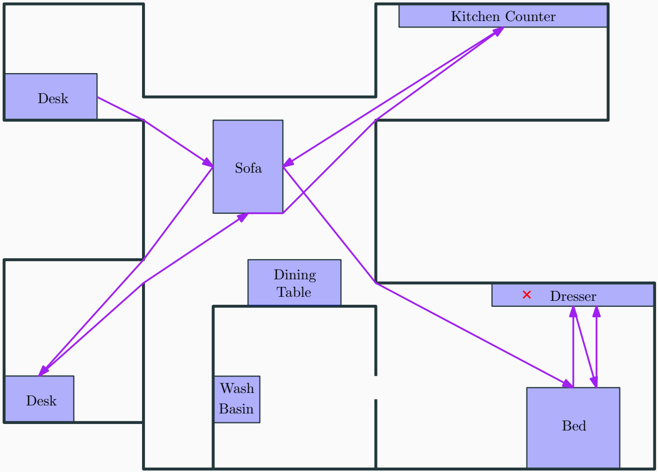
Motivation

How do we search for something quickly when we have imperfect sensing capabilities?

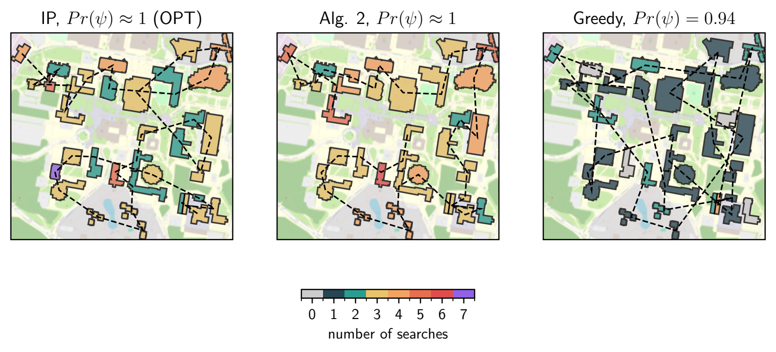
Finding Lost Glasses



Finding Lost Glasses



Search and Rescue



In [Chakraborty, Kasthurirangan, Mitchell, Nguyen, Perk, 2024].

Formal Problem Statement

Three Objectives

How do we search for something quickly when we have imperfect sensing capabilities?

Three Objectives

How do we search for something quickly when we have imperfect sensing capabilities?

- Minimize the *maximum* time required to find the target.

Three Objectives

How do we search for something quickly when we have imperfect sensing capabilities?

- Minimize the *maximum* time required to find the target.
- Minimize the *average* time required to find the target.

Three Objectives

How do we search for something quickly when we have imperfect sensing capabilities?

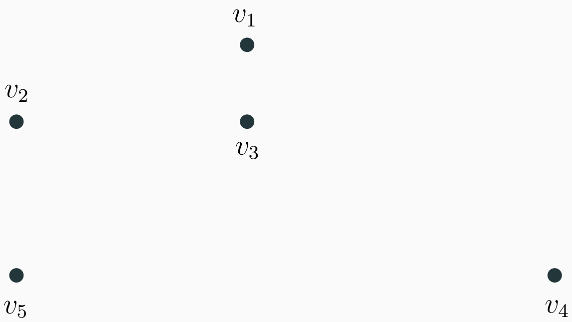
- Minimize the *maximum* time required to find the target.
- Minimize the *average* time required to find the target.
- Maximize the chance that you find the target *within* a time budget.

Three Objectives

How do we search for something quickly when we have ~~im~~perfect sensing capabilities?

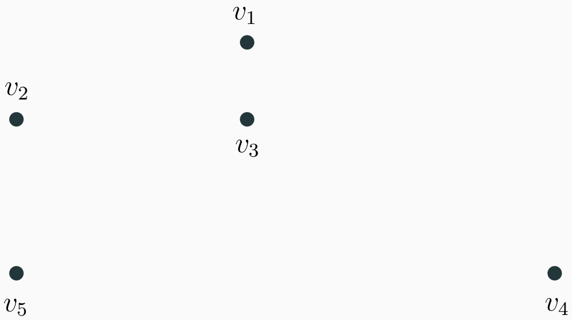
- Minimize the *maximum* time required to find the target.
- Minimize the *average* time required to find the target.
- Maximize the chance that you find the target *within* a time budget.

Travelling Salesman



A target is hidden in one of a given set of points on the plane. Minimize the *maximum* time required to find it.

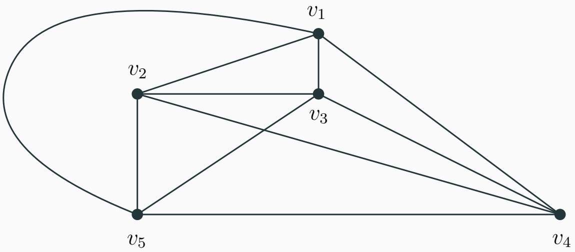
Travelling Salesman



A target is hidden in one of a given set of points on the plane.
Find the *shortest path* to visit all of them.

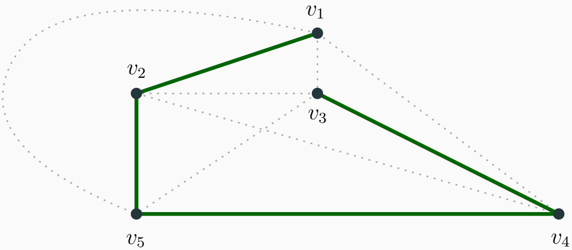
Travelling Salesman

Travelling Salesman

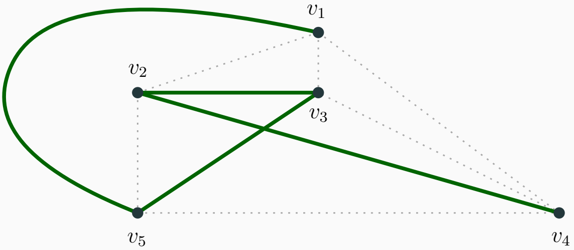


A target is hidden in one of a given set of points on the plane. Find the *shortest path* to visit all of them.

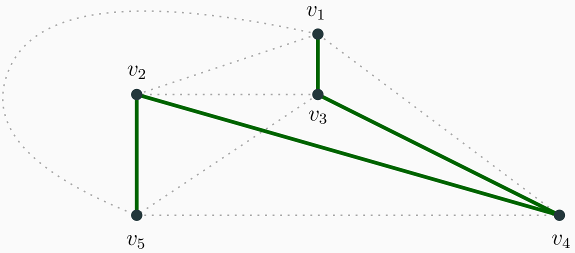
Travelling Salesman

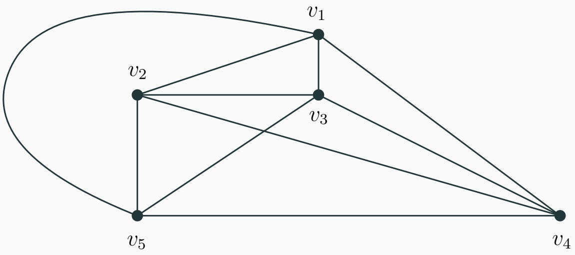


Travelling Salesman



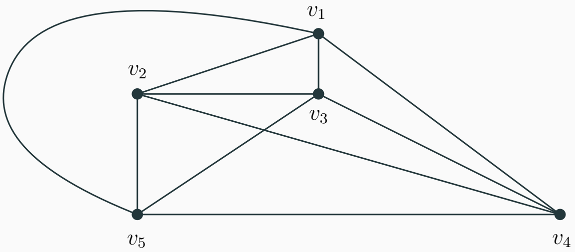
Travelling Salesman





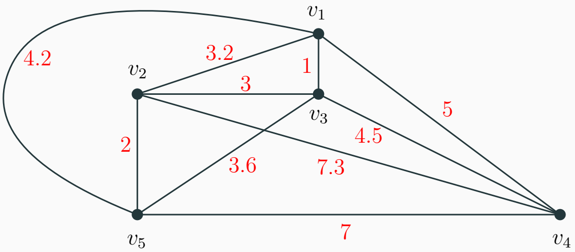
The *shortest path* connects all vertices.

Travelling Salesman

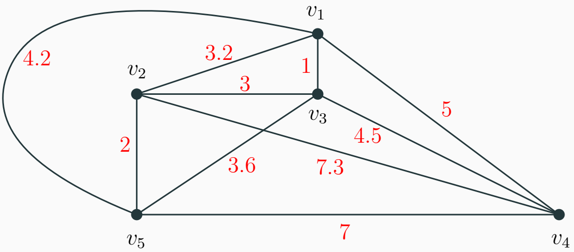


The *shortest path* that connects all vertices is larger than the *smallest weight subgraph* that connects all vertices.

Finding the Smallest Weight Subgraph

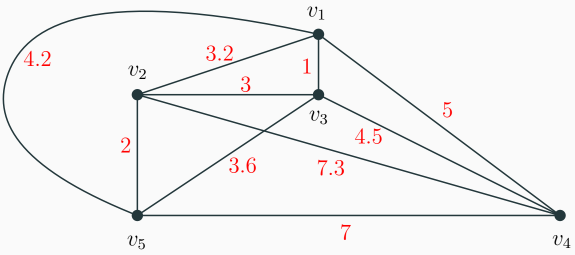


Finding the Smallest Weight Subgraph



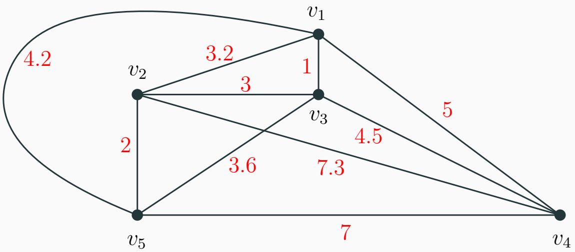
How do we construct the *smallest weight subgraph* that connects all vertices?

Finding the Smallest Weight Subgraph



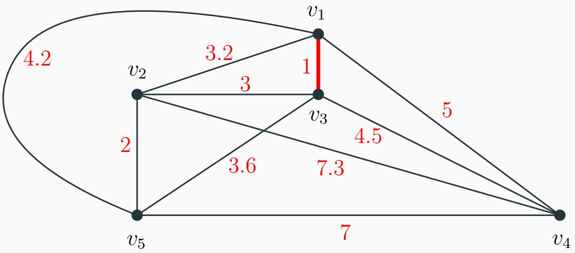
How do we construct the *Minimum Spanning Tree* (MST)?

Finding the Smallest Weight Subgraph

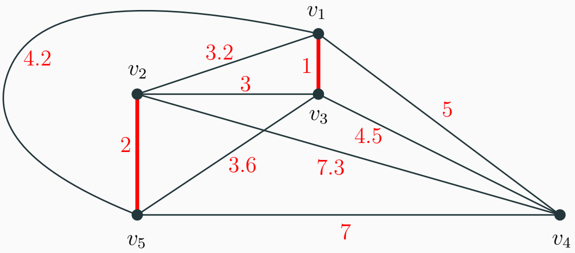


Iteratively pick small edges!

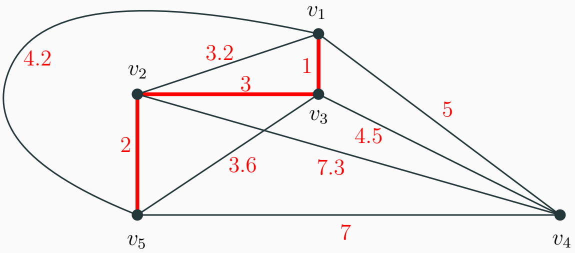
Finding the Smallest Weight Subgraph



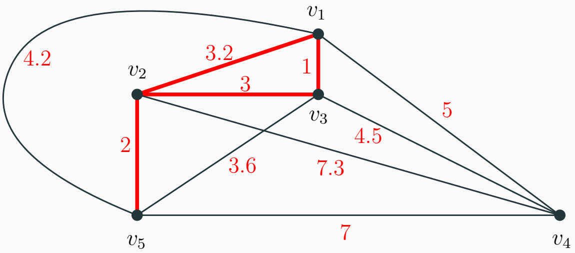
Finding the Smallest Weight Subgraph



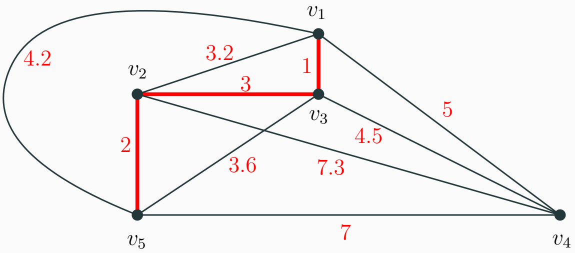
Finding the Smallest Weight Subgraph



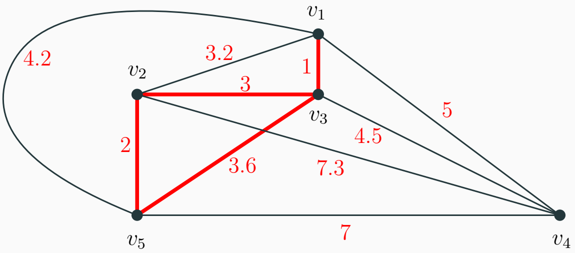
Finding the Smallest Weight Subgraph



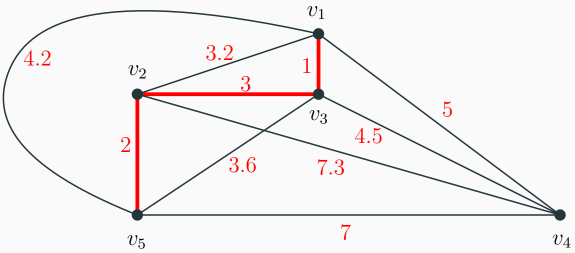
Finding the Smallest Weight Subgraph



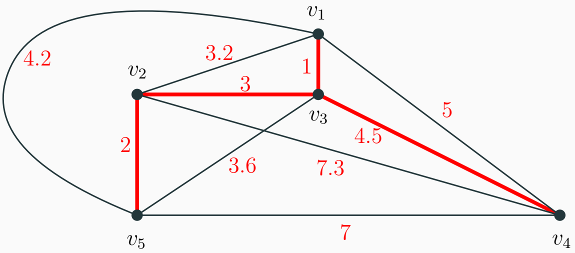
Finding the Smallest Weight Subgraph



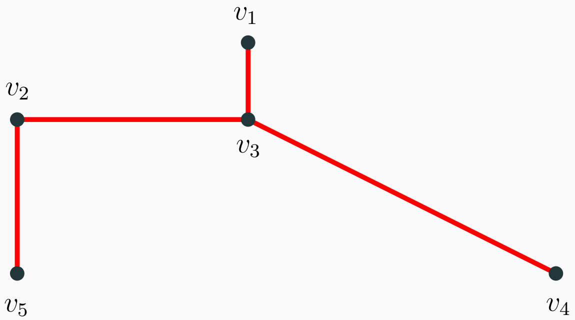
Finding the Smallest Weight Subgraph



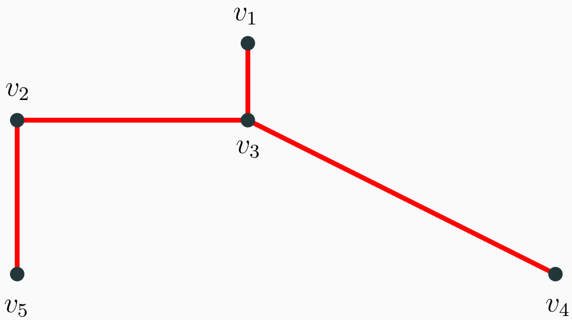
Finding the Smallest Weight Subgraph



Finding the Smallest Weight Subgraph

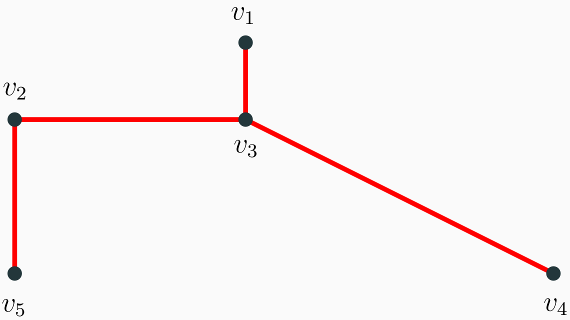


Finding the Smallest Weight Subgraph



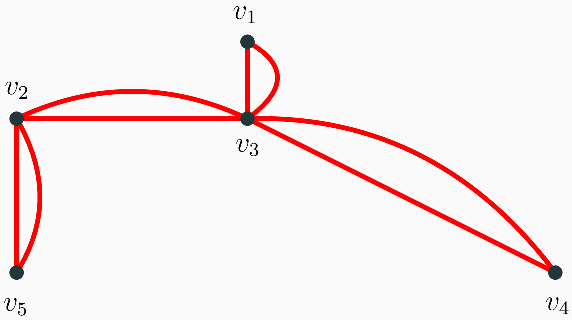
Recall: the *shortest path* is larger than the *smallest weight subgraph* that connects all vertices.

Finding the Smallest Weight Subgraph



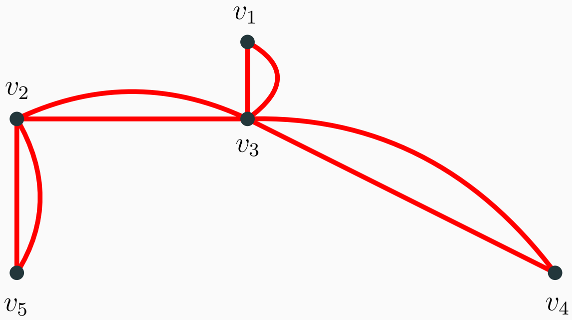
Recall: the *shortest path* is larger than the *smallest weight subgraph* that connects all vertices. $|\text{MST*}| \leq |\text{TSP*}|$.

Doubling



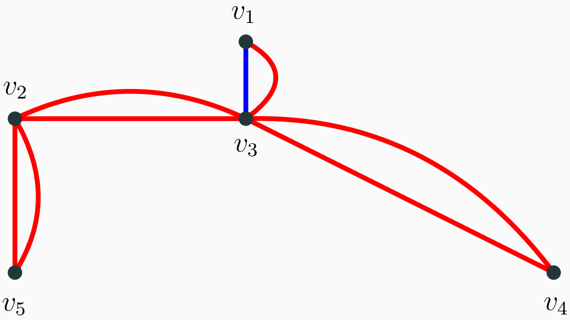
Doubling edges gives us a way to “backtrack”.

Doubling



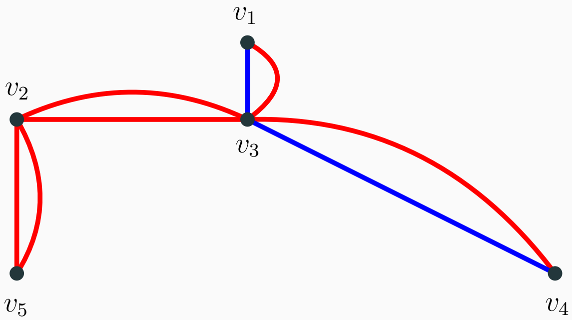
Doubling edges gives us a way to "backtrack". Let us make a *tour* from this!

A Cycle from a Doubled Tree



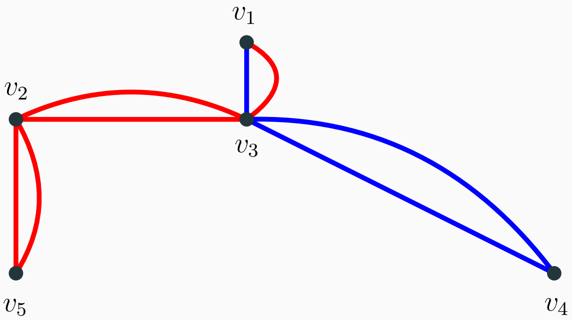
(v_1, v_3).

A Cycle from a Doubled Tree



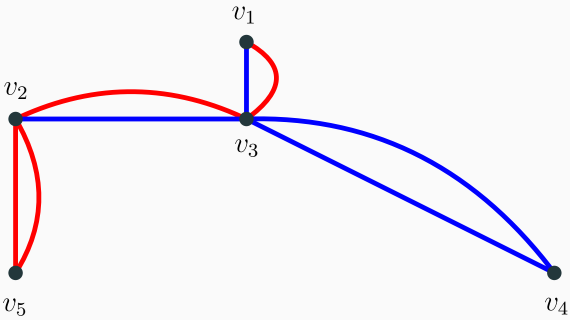
(v_1, v_3, v_4).

A Cycle from a Doubled Tree



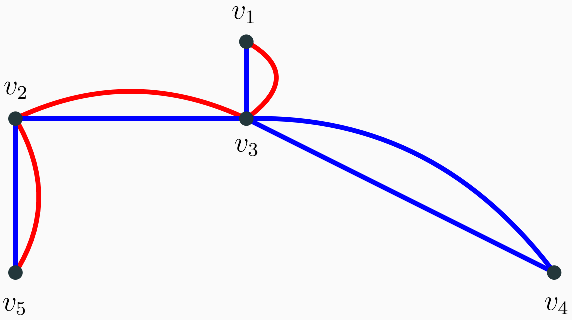
(v_1, v_3, v_4, v_3).

A Cycle from a Doubled Tree



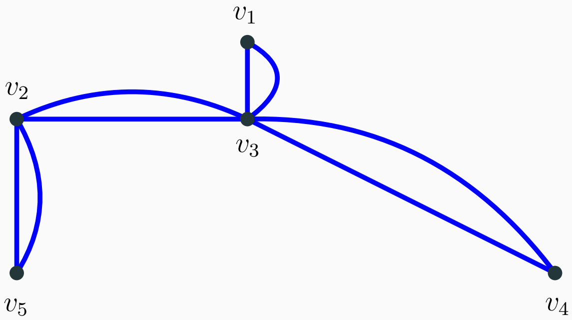
$(v_1, v_3, v_4, v_3, v_2)$.

A Cycle from a Doubled Tree



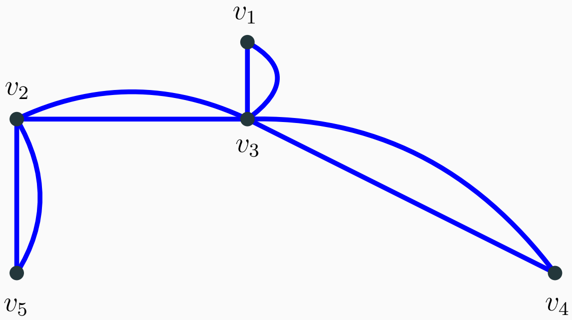
$(v_1, v_3, v_4, v_3, v_2, v_5)$.

A Cycle from a Doubled Tree



$(v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

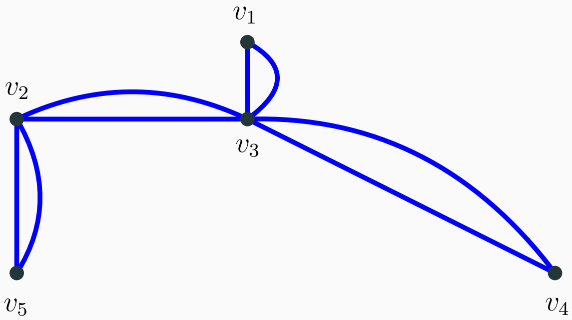
A Cycle from a Doubled Tree



$(v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

$|\text{DMST*}| =$

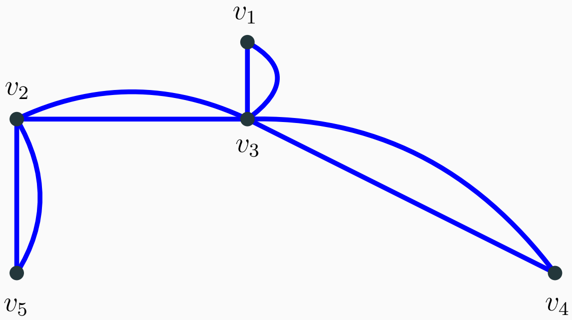
A Cycle from a Doubled Tree



$(v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

$|\text{DMST*}| = 2|\text{MST*}|$.

A Cycle from a Doubled Tree

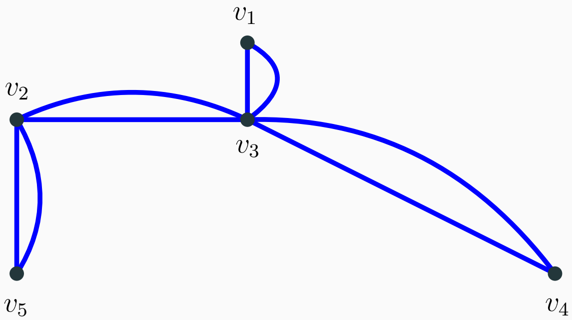


$(v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

$|\text{DMST*}| = 2|\text{MST*}|$.

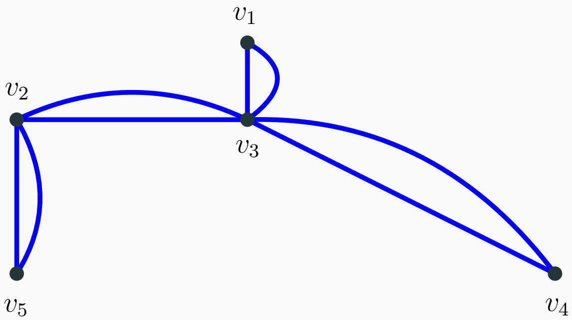
Recall: $|\text{MST*}| \leq |\text{TSP*}|$.

A Cycle from a Doubled Tree



$|\text{DMST*}| = 2|\text{MST*}|$. $|\text{MST*}| \leq |\text{TSP*}|$.

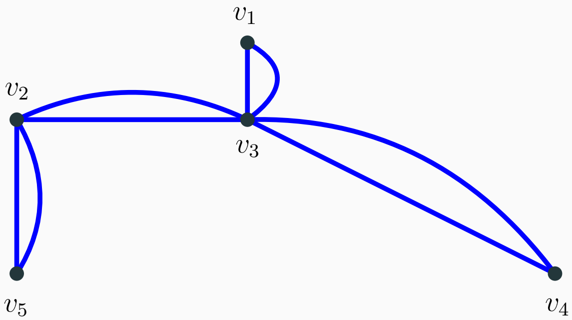
A Cycle from a Doubled Tree



$|\text{DMST}^*| = 2|\text{MST}^*|$. $|\text{MST}^*| \leq |\text{TSP}^*|$.

If $|\text{TSP}| \leq |\text{DMST}^*|$

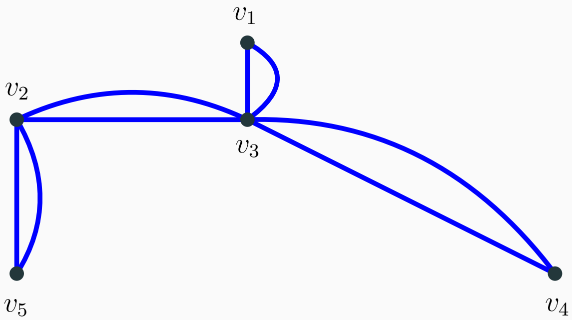
A Cycle from a Doubled Tree



$|$DMST*$| = 2|$MST*$|$. $|$MST*$| \leq |$TSP*$|$.

If $|$TSP$| \leq |$DMST*$|$; then, $|$TSP$| \leq 2|$MST*$| \implies$

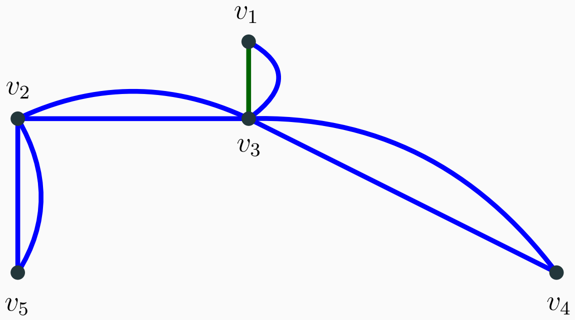
A Cycle from a Doubled Tree



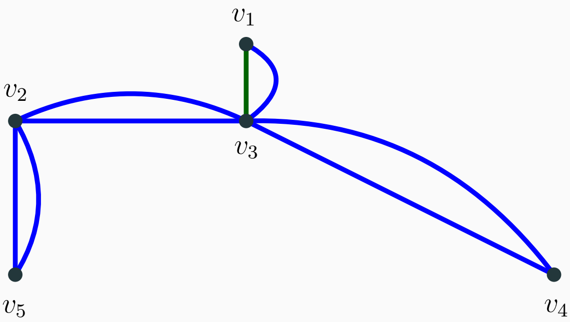
$|\text{DMST*}| = 2|\text{MST*}|$. $|\text{MST*}| \leq |\text{TSP*}|$.

If $|\text{TSP}| \leq |\text{DMST*}|$; then, $|\text{TSP}| \leq 2|\text{MST*}| \implies |\text{TSP}| \leq 2|\text{TSP*}|$.

Travelling Salesman

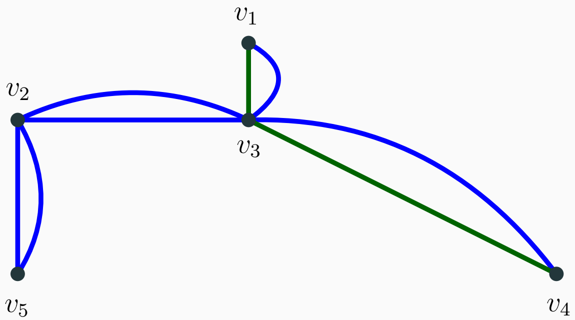


TSP $= (v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.



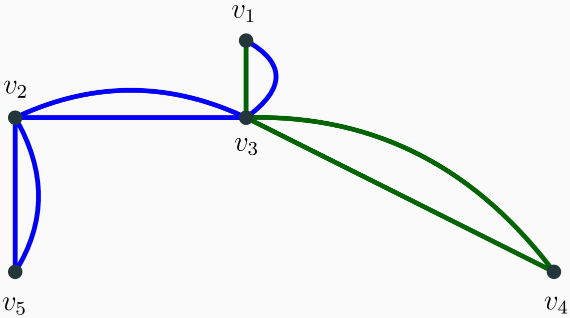
$\text{TSP} = (v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

$|\text{TSP}| = |v_1 v_3| \leq |\text{DMST*}|$.



$\text{TSP} = (v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

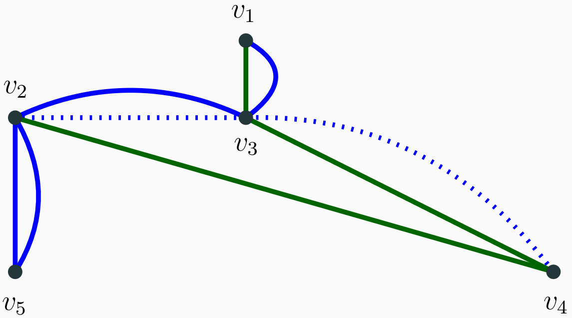
$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| \leq |\text{DMST*}|$.



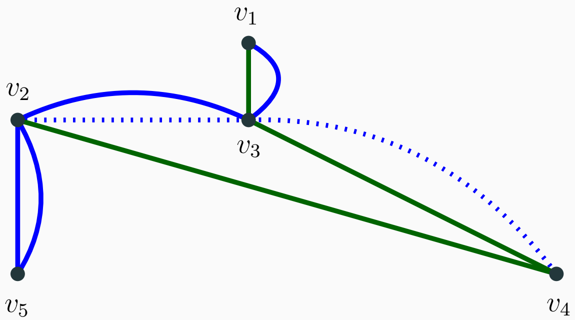
$\text{TSP} = (v_1, v_3, v_4, v_3, v_2, v_5, v_2, v_3, v_1)$.

$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| + |v_4 v_3| \leq |\text{DMST}^*|$.

Travelling Salesman



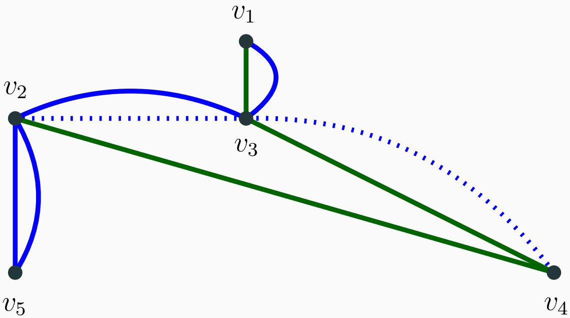
$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, v_2, v_3, v_1).$



$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, v_2, v_3, v_1)$.

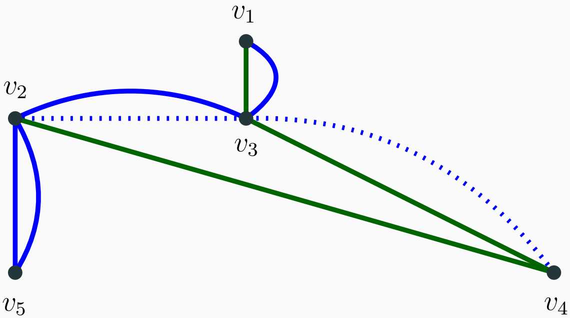
$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| + |v_4 v_2|$.

Travelling Salesman



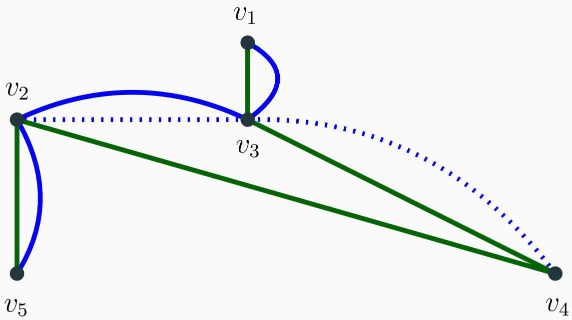
$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, v_2, v_3, v_1).$

$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| + \underbrace{|v_4 v_2|}_{\leq |v_4 v_3| + |v_3 v_2|} .$



$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, v_2, v_3, v_1).$

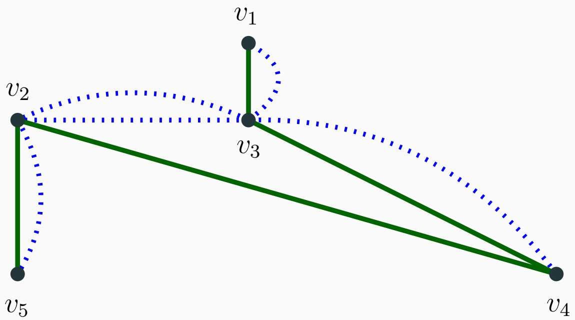
$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| + \underbrace{|v_4 v_2|}_{\leq |v_4 v_3| + |v_3 v_2|} \leq |\text{DMST}^*|.$



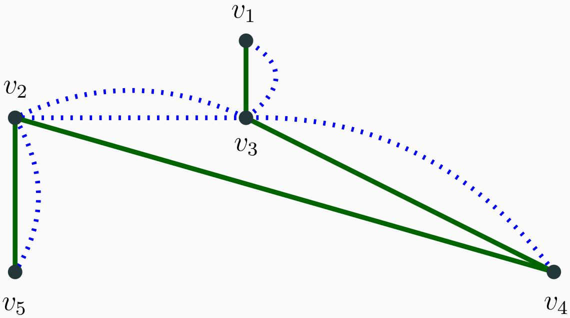
$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, v_2, v_3, v_1)$.

$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| + |v_4 v_2| + |v_2 v_5| \leq |\text{DMST*}|$.

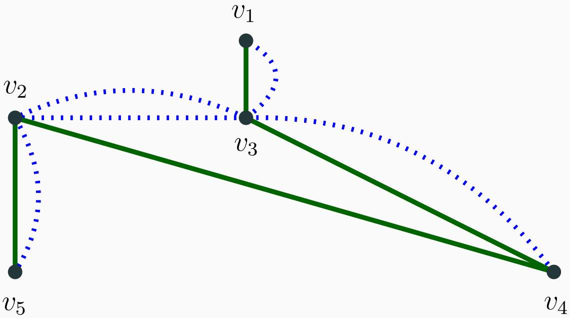
Travelling Salesman



$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, \cancel{v_2}, \cancel{v_3}, \cancel{v_1}).$



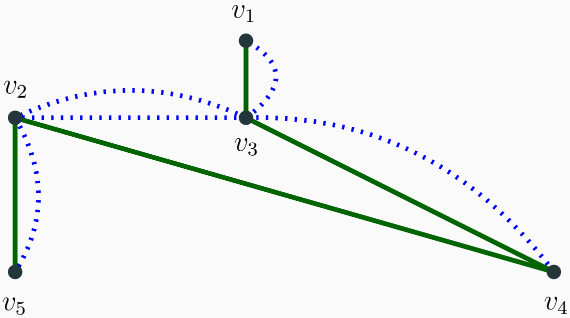
$\mathsf{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, \cancel{v_2}, \cancel{v_3}, \cancel{v_1})$. $\mathsf{TSP} = (v_1, v_3, v_4, v_2, v_5)$.



$\text{TSP} = (v_1, v_3, v_4, \cancel{v_3}, v_4, v_5, \cancel{v_2}, \cancel{v_3}, \cancel{v_1})$. $\text{TSP} = (v_1, v_3, v_4, v_2, v_5)$.

$|\text{TSP}| = |v_1 v_3| + |v_3 v_4| + |v_4 v_2| + |v_2 v_5| \leq |\text{DMST*}|$.

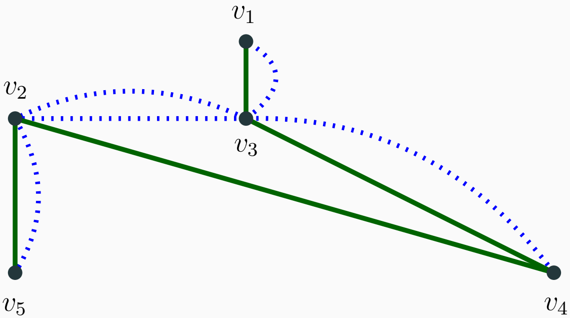
Recall



$|\text{DMST*}| = 2|\text{MST*}|$. $|\text{MST*}| \leq |\text{TSP*}|$.

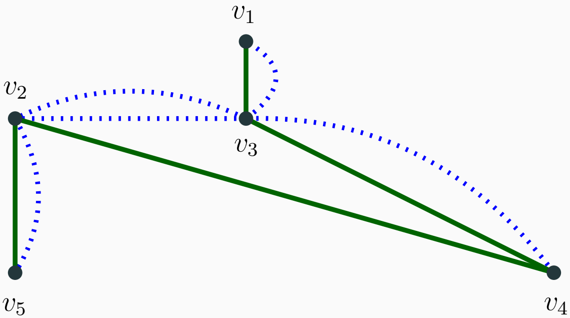
If $|\text{TSP}| \leq |\text{DMST*}|$; $|\text{TSP}| \leq 2|\text{MST*}| \implies |\text{TSP}| \leq 2|\text{TSP*}|$.

Travelling Salesman



$|\mathrm{TSP}| \leq 2|\mathrm{TSP}^*|$.

Travelling Salesman



$|\text{TSP}| \leq 2|\text{TSP}^*|$. We have given a *2-approximation algorithm* for Travelling Salesman!

Travelling Salesman– Better Algorithms?

- A small modification gives us a $\frac{3}{2}$-approximation algorithm [Christofedes, 1976]!

Travelling Salesman– Better Algorithms?

- A small modification gives us a $\frac{3}{2}$-approximation algorithm [Christofedes, 1976]!
- Breaking news! There is a a $(\frac{3}{2} - 10^{-36})$-approximation algorithm for Travelling Salesman on any *metric space* [Karlin, Klein, Gharan, 2021].

Travelling Salesman– Better Algorithms?

- A small modification gives us a $\frac{3}{2}$-approximation algorithm [Christofedes, 1976]!
- Breaking news! There is a a $(\frac{3}{2} - 10^{-36})$-approximation algorithm for Travelling Salesman on any *metric space* [Karlin, Klein, Gharan, 2021].
- For any $\epsilon > 0$, there is a $(1 + \epsilon)$-approximation algorithm [Arora, 1998], [Mitchell, 1999] for Travelling Salesman on the plane!

Travelling Salesman– Better Algorithms?

- Travelling Salesman is NP-Hard.

Travelling Salesman– Better Algorithms?

- Travelling Salesman is NP-Hard.
- It admits an obvious $\mathcal{O}(n^n)$-time algorithm.

Travelling Salesman– Better Algorithms?

- Travelling Salesman is NP-Hard.
- It admits an obvious $\mathcal{O}(n^n)$-time algorithm.
- Using *dynamic programming* techniques, we can get an $\mathcal{O}(2^n)$-time algorithm [Held, Karp, 1961], [Bellman, 1962].

Travelling Salesman– Better Algorithms?

- Travelling Salesman is NP-Hard.
- It admits an obvious $\mathcal{O}(n^n)$-time algorithm.
- Using *dynamic programming* techniques, we can get an $\mathcal{O}(2^n)$-time algorithm [Held, Karp, 1961], [Bellman, 1962].
- Open question: Does there exists a faster exact algorithm for Travelling Salesman?

Other Objectives

Three Objectives

How do we search for something quickly when we have ~~im~~perfect sensing capabilities?

- Minimize the *maximum* time required to find the target.
- Minimize the *average* time required to find the target.
- Maximize the chance that you find the target *within* a time budget.

TRAVELLING SALESMAN.

Three Objectives

How do we search for something quickly when we have ~~im~~perfect sensing capabilities?

- Minimize the *maximum* time required to find the target.
- Minimize the *average* time required to find the target.
- Maximize the chance that you find the target *within* a time budget.

Minimum Latency.

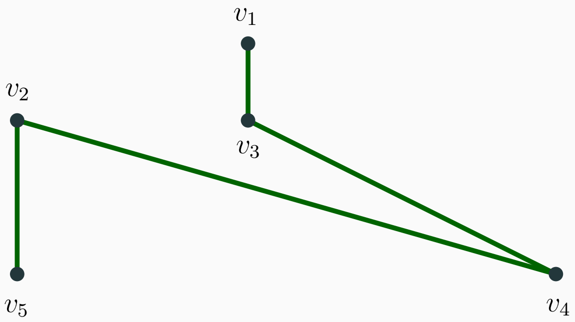
Three Objectives

How do we search for something quickly when we have ~~im~~perfect sensing capabilities?

- Minimize the *maximum* time required to find the target.
- Minimize the *average* time required to find the target.
- Maximize the chance that you find the target *within* a time budget.

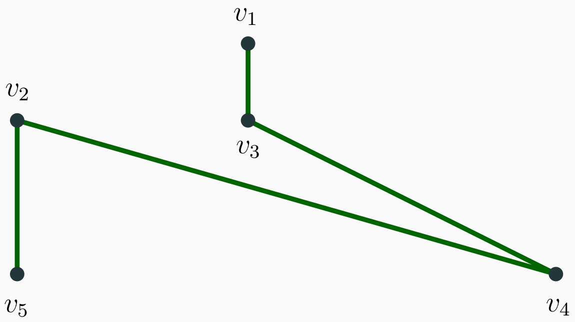
ORIEENTEERING.

Generalizations of Travelling Salesman

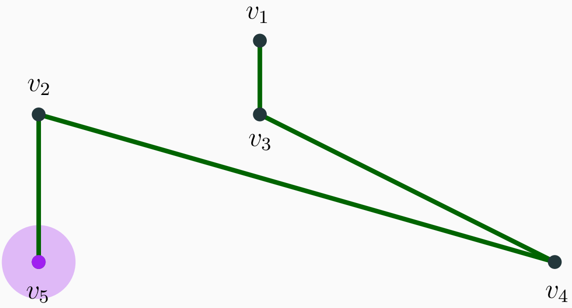


Minimize the *maximum* time required to find the target.

Generalizations of Travelling Salesman

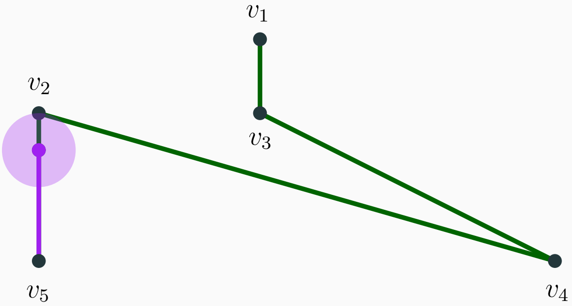


Minimize the *maximum* time required to find the target. To search a point, you must be *at* that point.



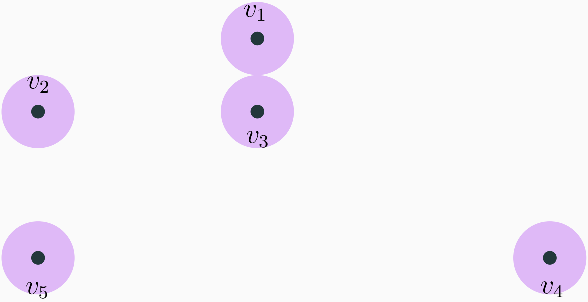
What if you have a search radius $r > 0$?

Generalizations of Travelling Salesman



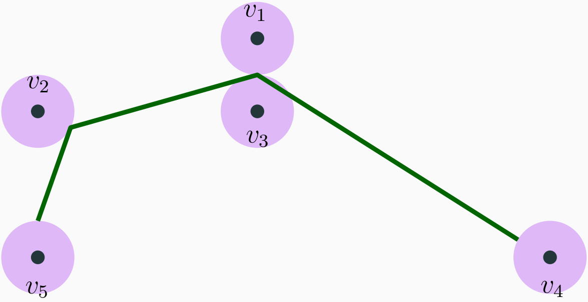
You only need to get within *r* of a point to search it!

Generalizations of Travelling Salesman



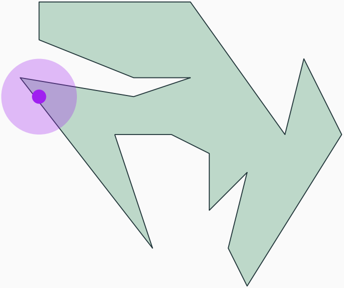
Travelling Salesman with Neighbourhoods.

Generalizations of Travelling Salesman



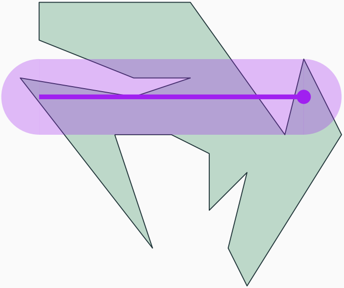
A Travelling Salesman with Neighbourhoods search path.

Generalizations of TRAVELLING SALESMAN



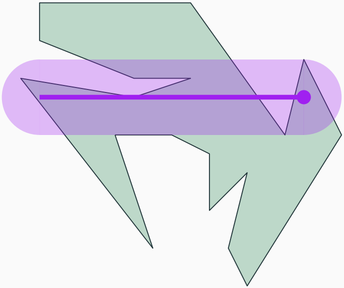
Searching a continuous domain.

Generalizations of TRAVELLING SALESMAN



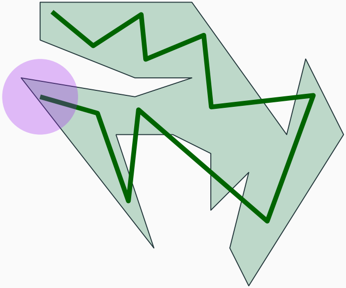
Searching a continuous domain.

Generalizations of Travelling Salesman



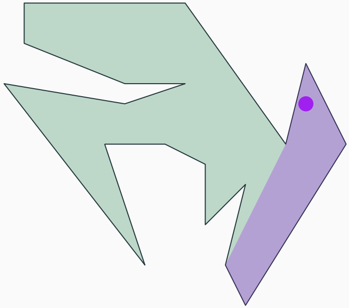
Searching a continuous domain. Lawn Mowing.

Generalizations of Travelling Salesman



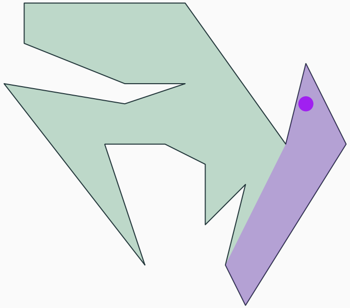
A Lawn Mowing search path.

Generalizations of Travelling Salesman



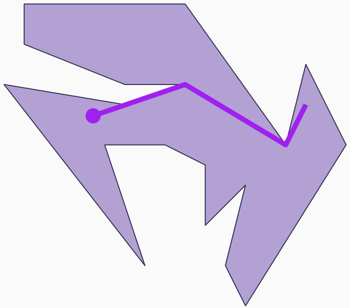
What if you can search everything that you can see?

Generalizations of Travelling Salesman



What if you can search everything that you can see? Watchman Route.

Generalizations of Travelling Salesman



A Watchman Route search path.

Questions?

How do we search for something quickly when we have imperfect sensing capabilities?

- You will *never* be completely sure.
- You might have to search the same point *multiple* times.
- $\Pr(X = i)$ *changes* as we conduct the search.

Questions?

How do we search for something quickly when we have imperfect sensing capabilities?

- You will *never* be completely sure.
- You might have to search the same point *multiple* times.
- $\Pr(X = i)$ *changes* as we conduct the search.

Thank you!