A Realistic Model to support Rescue Operations after an Earthquake

Based on a paper (with same title) presented at Q2SWinet@MSWiM 2020

Poster at MSWiM 2020

Tiziana Calamoneri and **Federico Corò**

SAPIENZA
UNIVERSITÀ DI ROMA

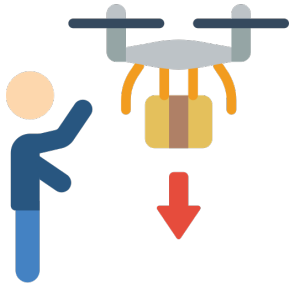
December 9, 2020

Sapienza University of Rome, Italy

Outline

Unmanned Aerial Vehicles (UAVs)

- UAVs are flying vehicles able to autonomously decide their route
- Historically, used in the military, mainly deployed in hostile territory to reduce pilot losses
- Now, new applications in civilian and commercial domains:
  - weather monitoring
  - forest fire detection
  - traffic control
  - **emergency search and rescue**



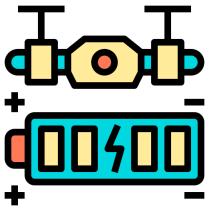
Outline

Real Life Situation

- Known Area of Interest (AoI)
- Fleet of UAVs
- Set of sites that must be examined
  - e.g., crumbled buildings after a earthquake
- Each site may need a different time to be inspected
- Each UAV must periodically go back to the operation centre in order to recharge its battery/exchange informations with the rescue team
- We want to overfly the AoI "as soon as possible" in order to collect data and possibly save people

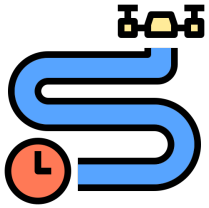
The Problem: from Real Life Situation to Models

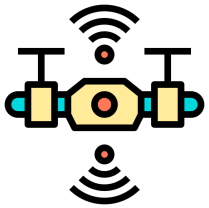
- All sites need to be visited in the "shortest time"



- Battery constraint

- UAV could spend a time even very different from the expected one to explore or traverse an area





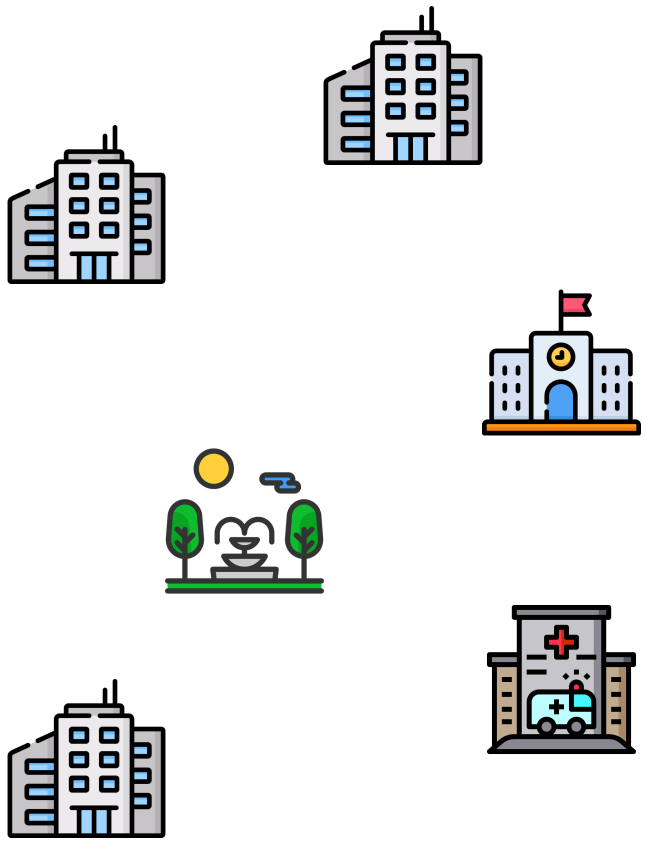
- Radio coverage for communications may not be guaranteed

The Graph Model

- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$

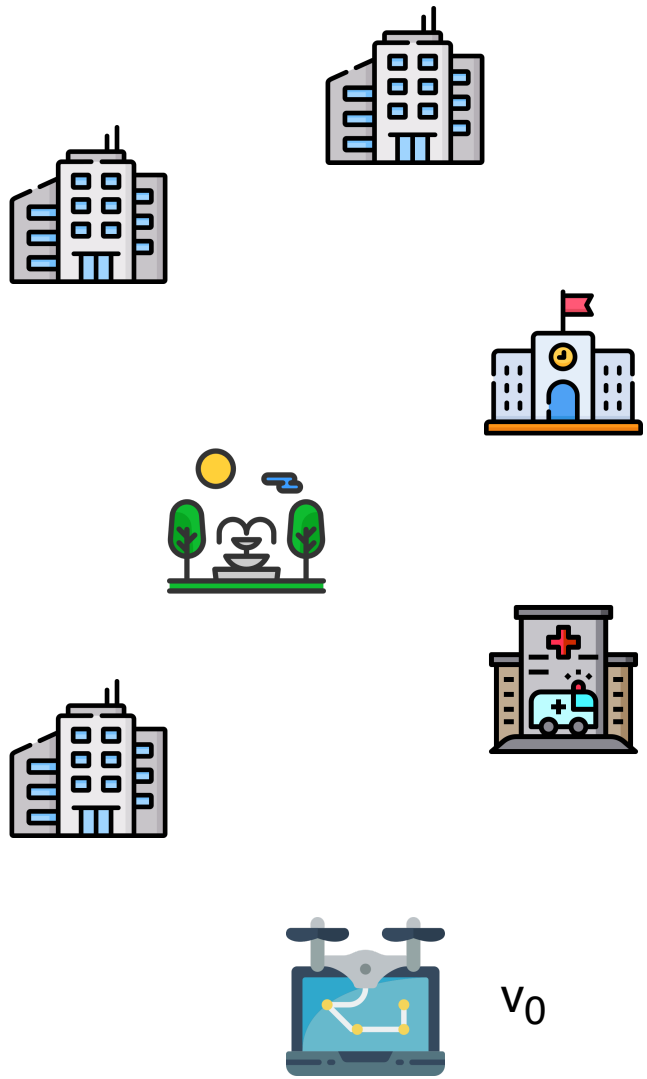
The Graph Model

- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$
- n nodes represents buildings (or areas) that need to be explored



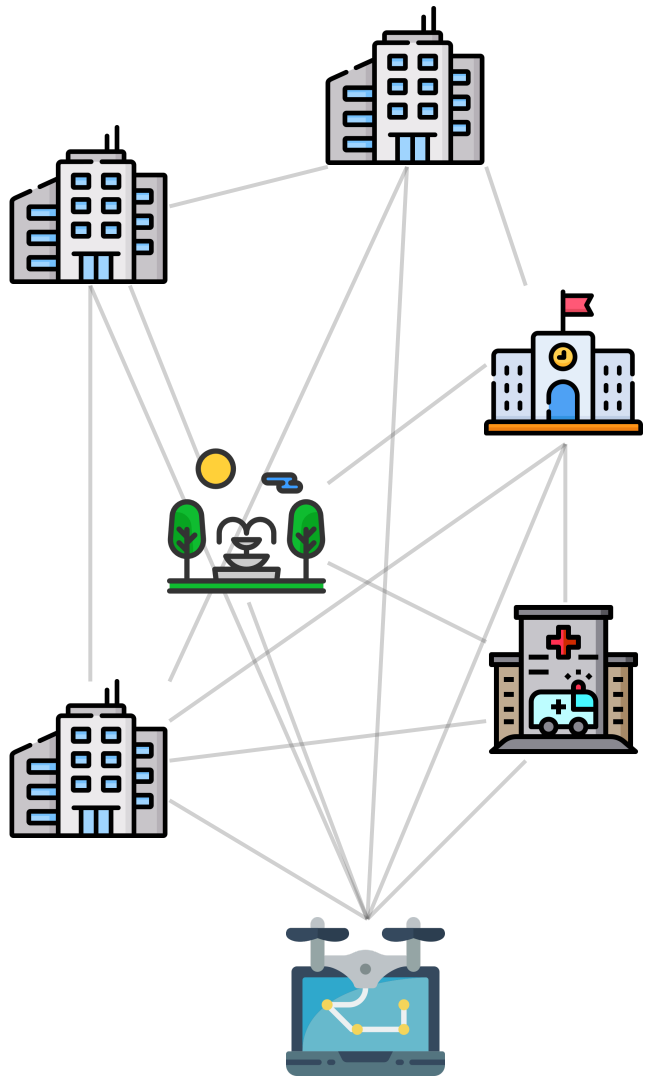
The Graph Model

- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$
- n nodes represents buildings (or areas) that need to be explored
- v_0 represents the operations centre



The Graph Model

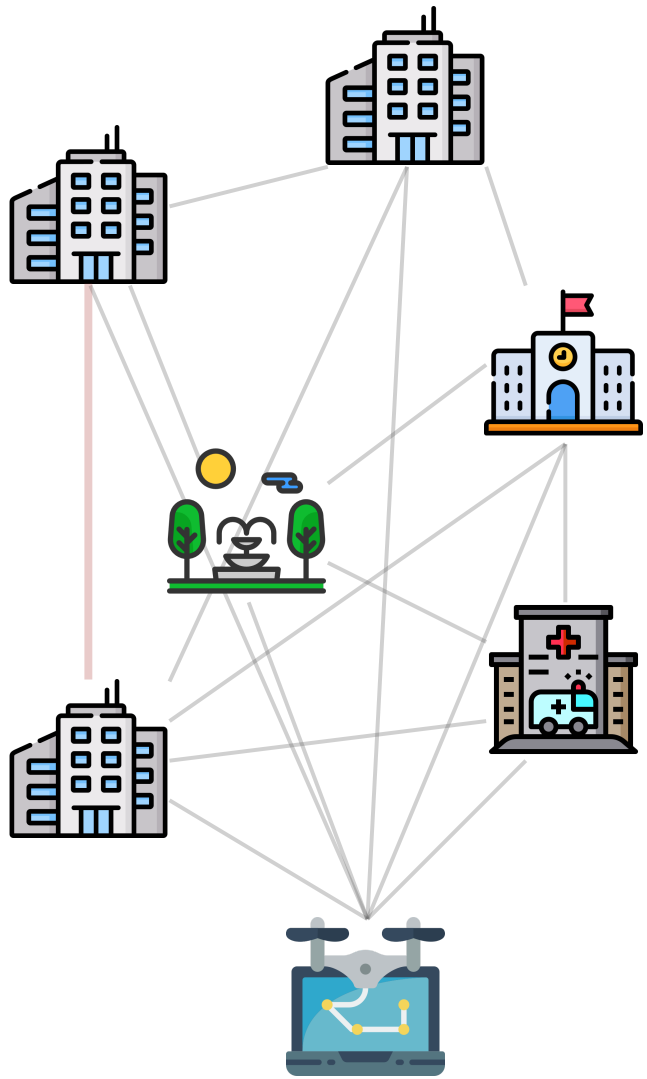
- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$
- n nodes represents buildings (or areas) that need to be explored
- v_0 represents the operations centre



The Graph Model

- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$
- n nodes represents buildings (or areas) that need to be explored
- v_0 represents the operations centre
- Weight of the edge represents distance

$$dist(u, v) = \sqrt{(u_x^2 - v_x^2) + (u_y^2 - v_y^2)}$$



The Graph Model

- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$
- $\sigma : V \to \mathbb{R}^+$ represents needed time to explore a site
  - building size
  - damage

$\sigma = 5$

$\sigma = 7$

$\sigma = 8$

The Graph Model

- Complete node- and edge-weighted graph $G = (V, E, dist, \sigma, p)$
- $\sigma : V \to \mathbb{R}^+$ represents needed time to explore a site
- $p : V \to \{p_{min}, p_{med}, p_{max}\}$ represents the priority

$\sigma = 5$

p_{min}

$\sigma = 7$

p_{med}

$\sigma = 8$

p_{max}

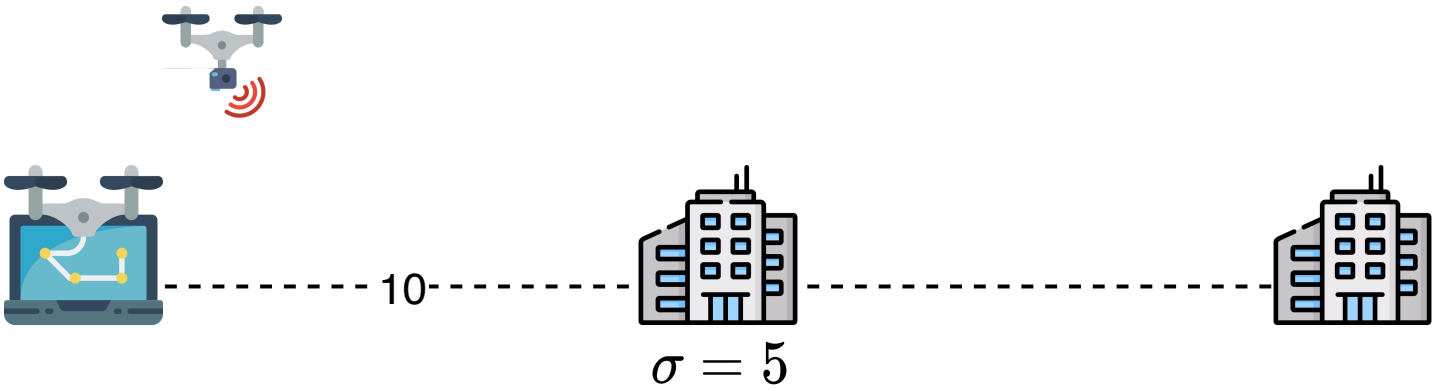
Scenario I

- Each UAV is equipped with an RGB commercial camera
- No computational power
- No communication devices
- UAVs can only follow assigned route
- Collected information must be brought back to be checked
- The positive side of this case is that the fleet can be easily constituted by a large number of UAVs because they are very cheap

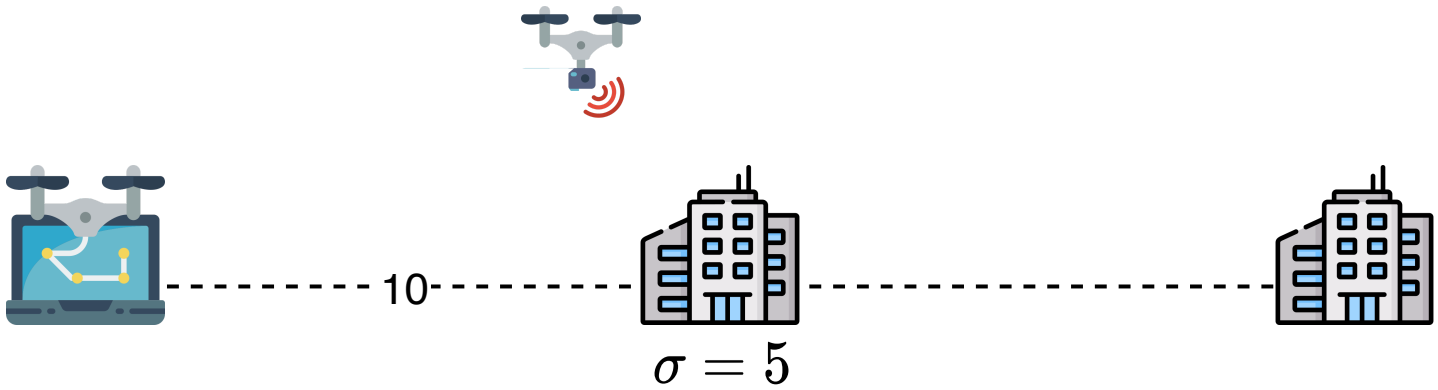
Scenario II

- Each UAV is equipped with an RGB commercial camera
- Good computational power
- Communication device
- Able to scan in real-time
- Able to detect and recognize an emergency
  - $\sigma' : V \rightarrow \mathbb{R}^+$ additional time
  - occur with given probability
  - e.g., decrease altitude, take precise information, send message to base

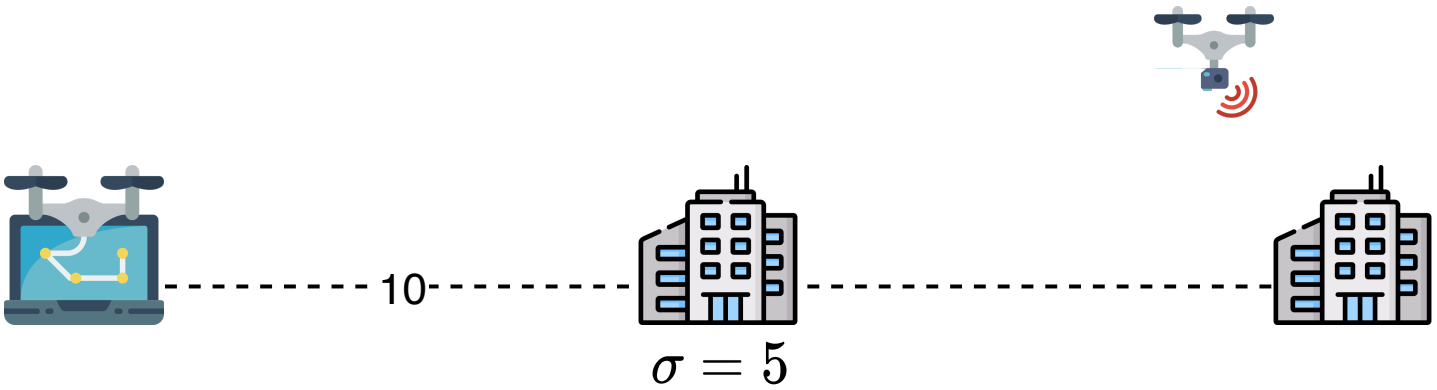
Scenario I: Example



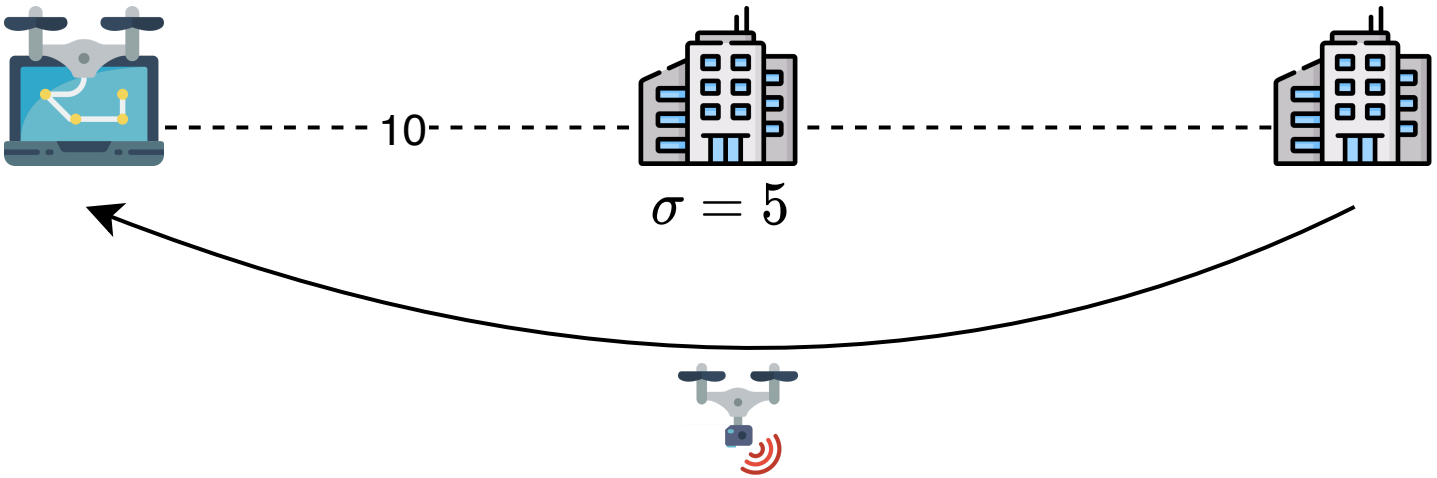
Scenario I: Example



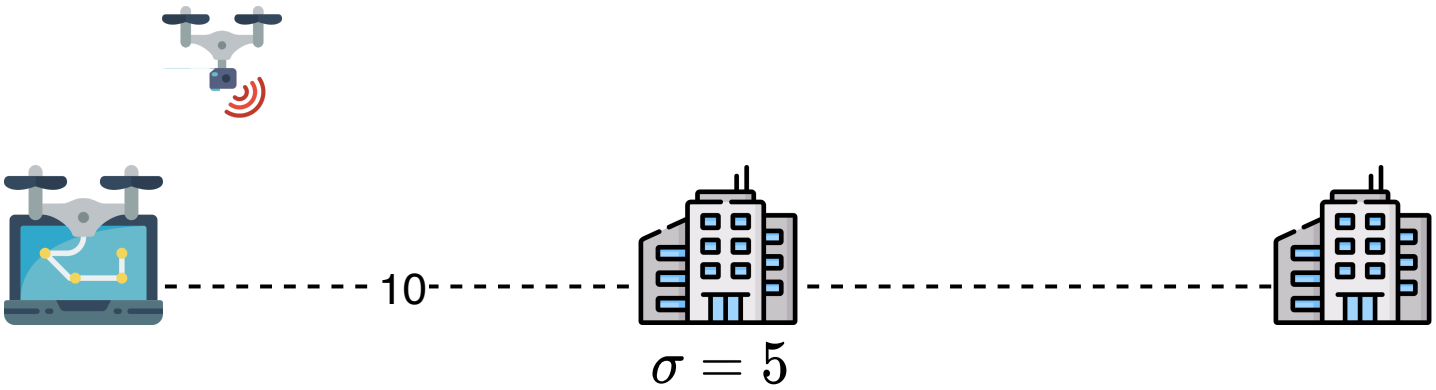
Scenario I: Example



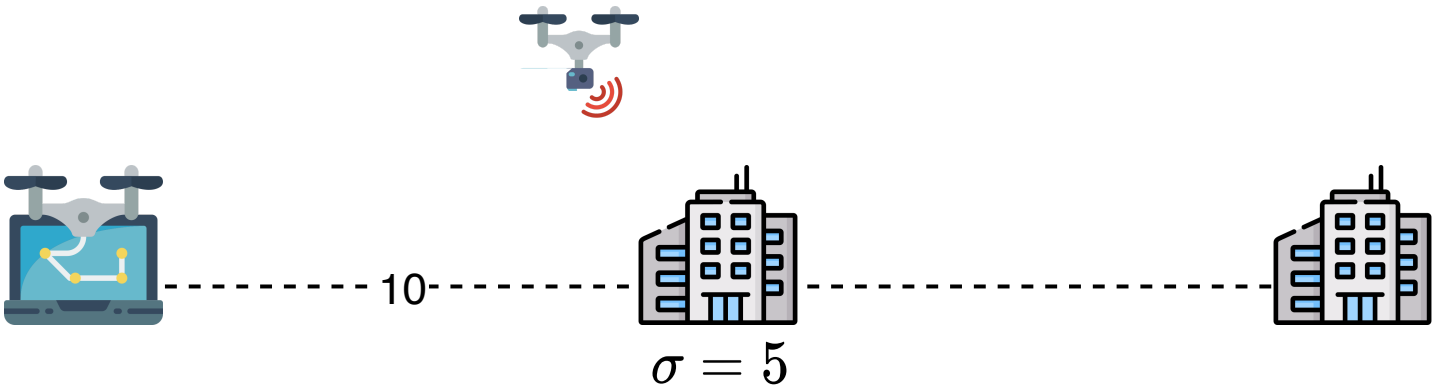
Scenario I: Example



Scenario II: Example



Scenario II: Example

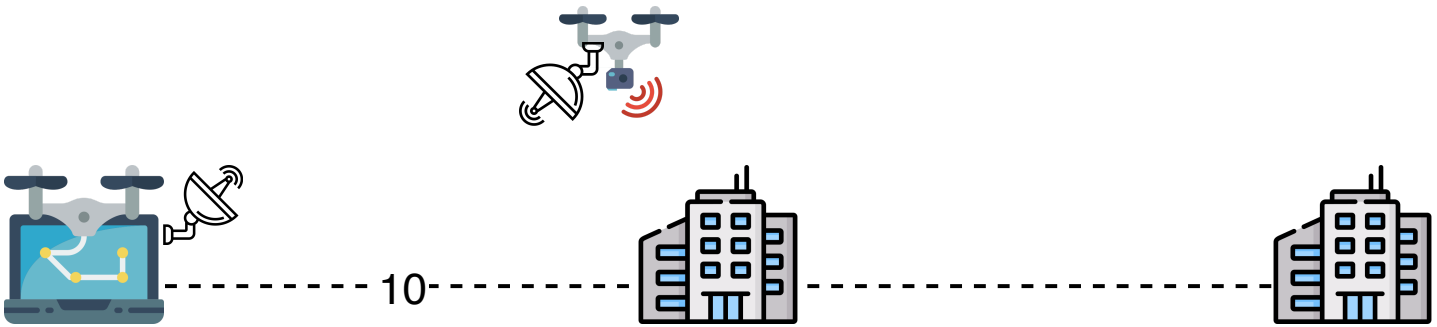


Scenario II: Example

$\sigma = 5$

$\sigma' = 5$

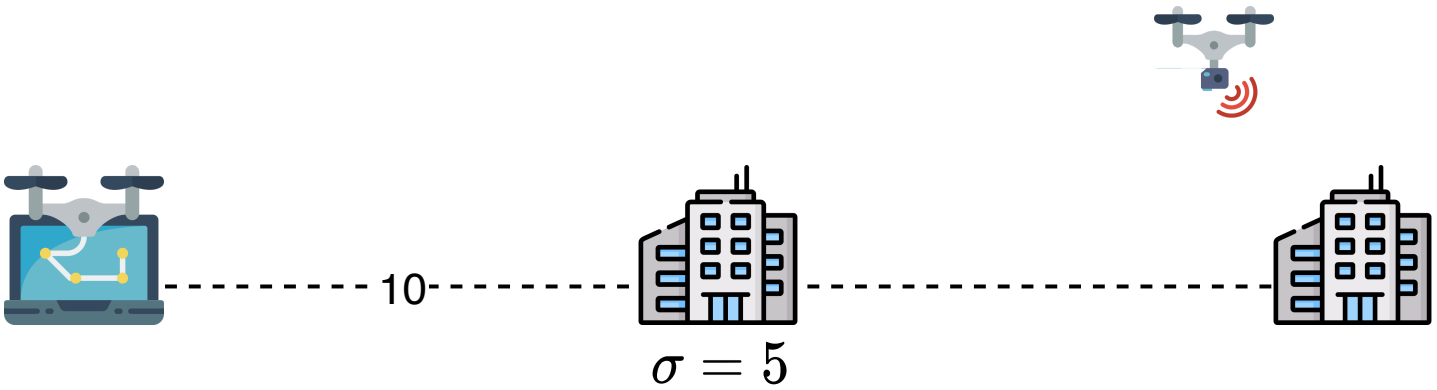
Scenario II: Example



$\sigma = 5$

$\sigma' = 5$

Scenario II: Example



Performance Metrics

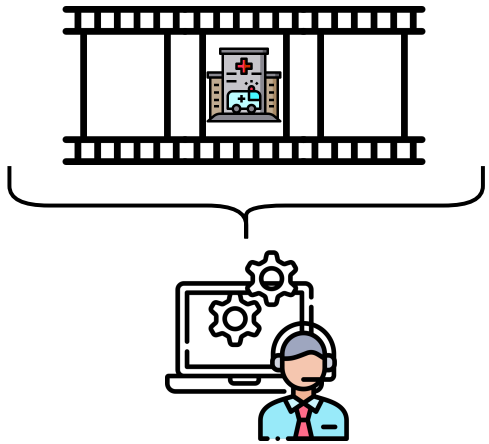
- Each UAV flies along a cycle and visits as many sites as it can, it goes back to the home-base to recharge its battery and it leaves again...
- All sites need to be visited in the "shortest time"
  - completion time == necessary time to know if people need help
  - sites with highest priority should be served first

Performance Metrics (cont.)

- In the two scenarios, we consider a site as "served" by a solution *SOL* if different conditions are verified

Scenario I a node has been "served" only after that the video of the cycle including it has been delivered to the base and the portion corresponding to it has been analyzed

$$cost^{(I)}(v_k, SOL) = t_f(C^e) + t^{(I)}(v_k)$$

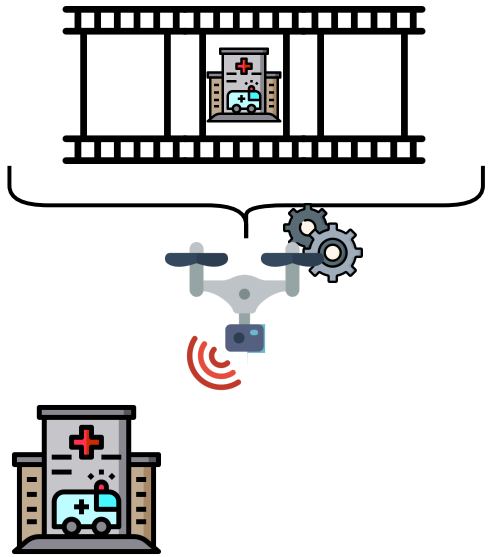


Performance Metrics (cont.)

- In the two scenarios, we consider a site as "served" by a solution *SOL* if different conditions are verified

Scenario II node is "served" as soon as it has been completely overflight since UAVs are equipped with a real-time tool able to immediately detect people needing help

$$cost^{(II)}(v_k, SOL) = t_s(C^e) + t^{(II)}(v_k)$$



Performance Metrics (cont.)

Definition

The *weighted latency of a solution SOL*, $wL(SOL)$, is the mean of the completion times of all sites (either in the first or in the second scenario), taking into account their priorities:

$$wL(SOL)^{(i)} = \frac{1}{n} \sum_{v \in V} p(v) cost^{(i)}(v, SOL^e) \text{ where } i = I, II$$

Definition

We denote as the *completion time* of a solution *SOL* as:

$$ct^{(i)}(SOL) = \max_{v \in V} cost^{(i)}(v, SOL) \text{ where } i = I, II$$

Outline

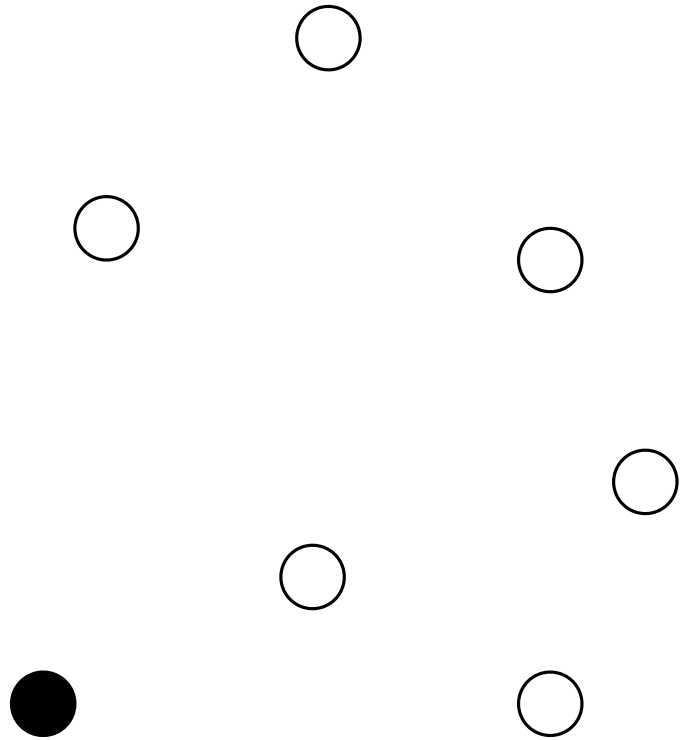
Meta-Algorithm

Meta-Algorithm:

- Runs at the operation center
- At the beginning, has complete knowledge of G (except σ')
- Can be used as pre-processing algorithm to do preliminary estimates on:
  - completion time
  - additional batteries
  - number of UAVs

Meta-Algorithm

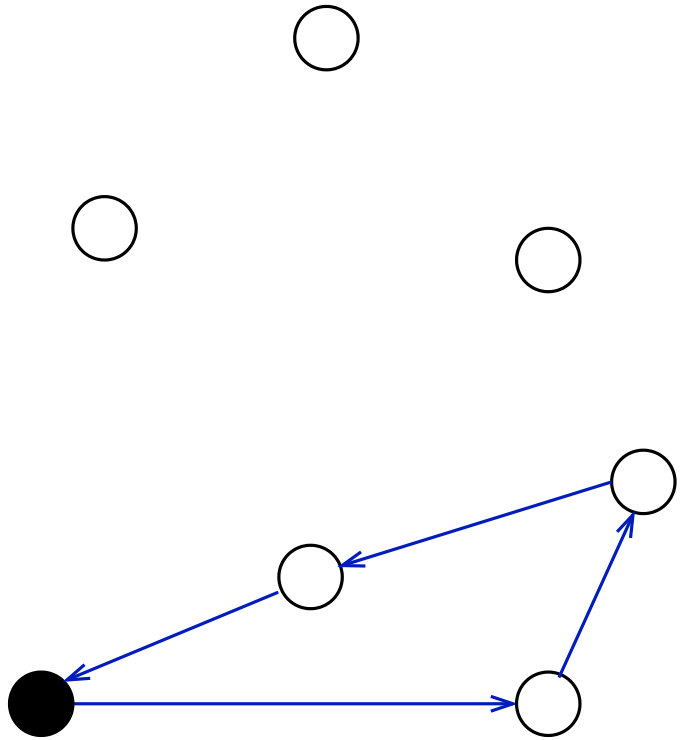
- The meta-algorithm consists of several iterations to be executed until all sites have been overflight
- At each iteration a certain algorithm computes a set of presumed cycles (one for each UAV)



Meta-Algorithm

- The meta-algorithm consists of several iterations to be executed until all sites have been overflight
- At each iteration a certain algorithm computes a set of presumed cycles (one for each UAV)

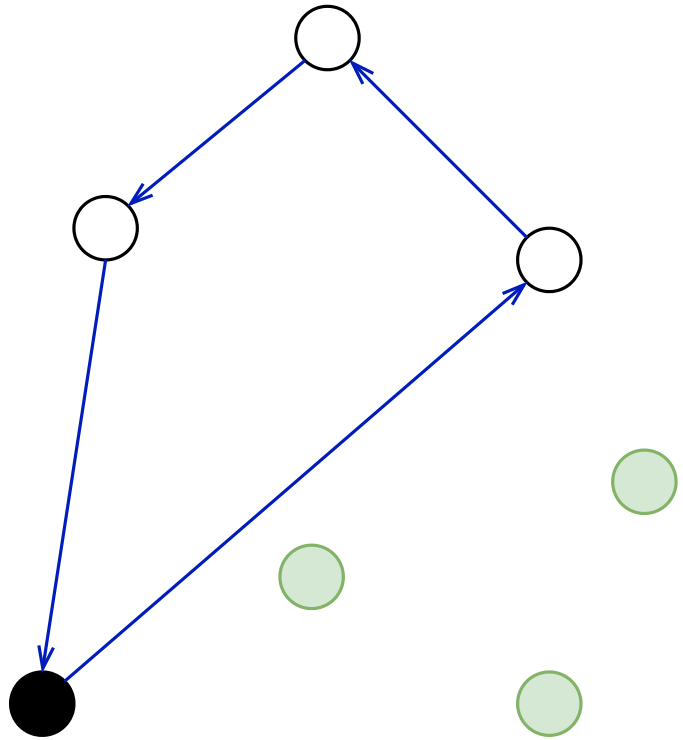
Scenario I presumed and effective cycles coincide $\rightarrow$ meta-algorithm outputs final solution



Meta-Algorithm

- The meta-algorithm consists of several iterations to be executed until all sites have been overflight
- At each iteration a certain algorithm computes a set of presumed cycles (one for each UAV)

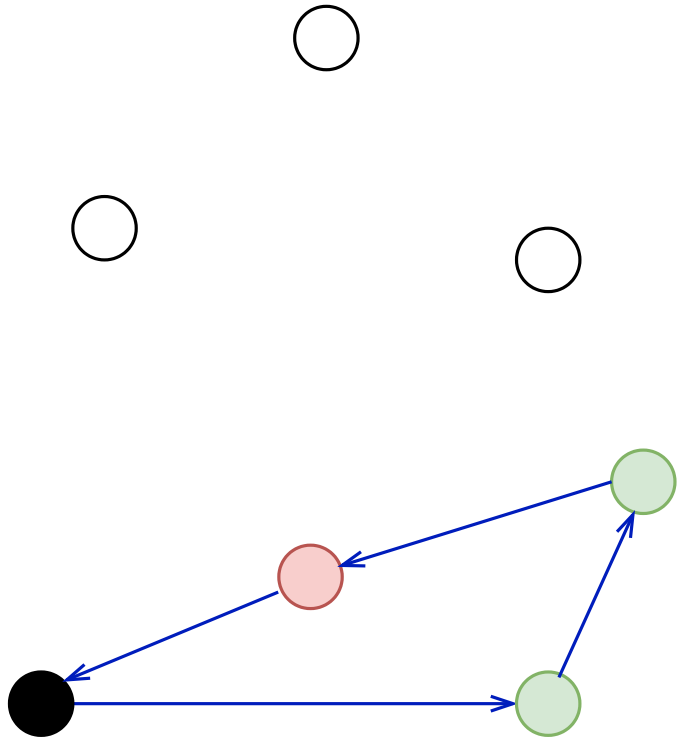
Scenario I presumed and effective cycles coincide $\rightarrow$ meta-algorithm outputs final solution



Meta-Algorithm

- The meta-algorithm consists of several iterations to be executed until all sites have been overflight
- At each iteration a certain algorithm computes a set of presumed cycles (one for each UAV)

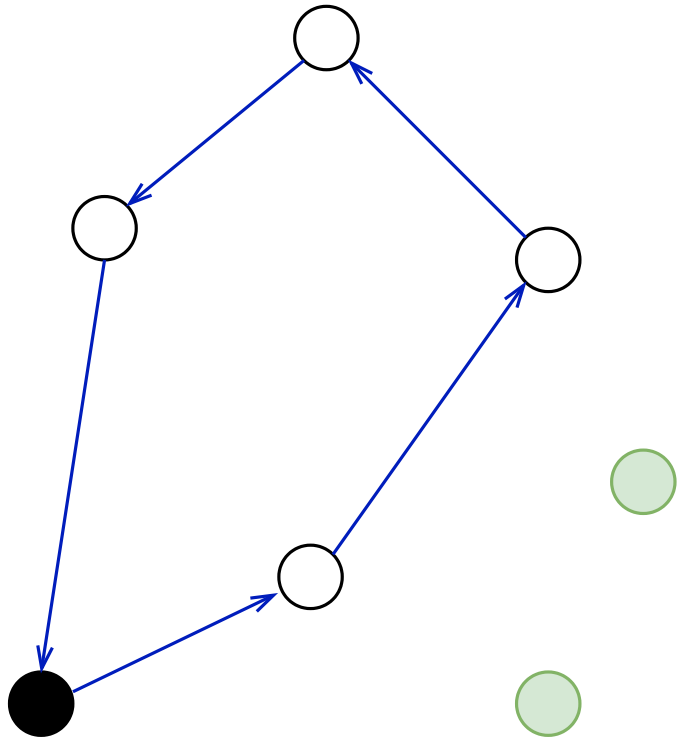
Scenario II presumed and effective cycles may not coincide $\rightarrow$ meta-algorithm compute a cycle for each UAV until all the sites have been overflight



Meta-Algorithm

- The meta-algorithm consists of several iterations to be executed until all sites have been overflight
- At each iteration a certain algorithm computes a set of presumed cycles (one for each UAV)

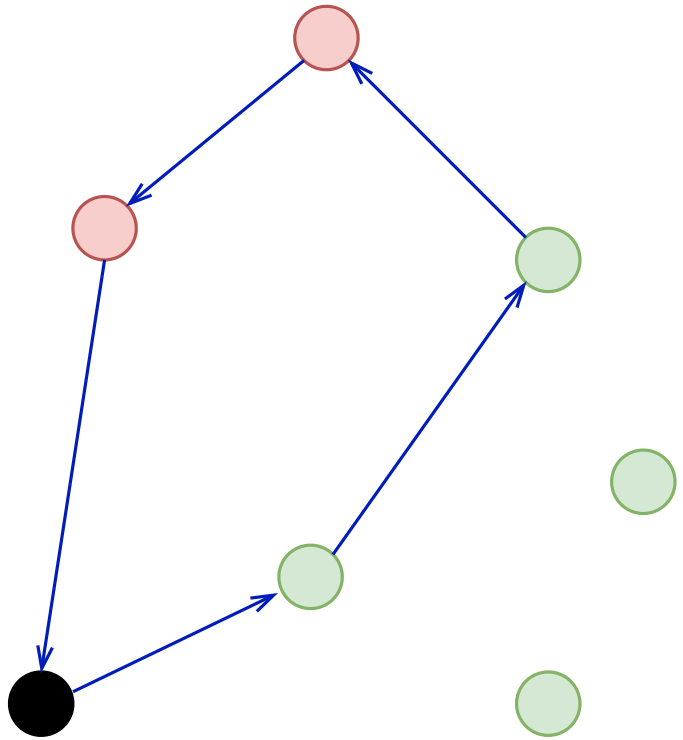
Scenario II presumed and effective cycles may not coincide $\rightarrow$ meta-algorithm compute a cycle for each UAV until all the sites have been overflight



Meta-Algorithm

- The meta-algorithm consists of several iterations to be executed until all sites have been overflight
- At each iteration a certain algorithm computes a set of presumed cycles (one for each UAV)

Scenario II presumed and effective cycles may not coincide $\rightarrow$ meta-algorithm compute a cycle for each UAV until all the sites have been overflight



Proposed Heuristics

- We propose four algorithms build upon our Meta-Algorithm:
  - Algorithm $\mathcal{H}_{TSPN}$
  - Algorithm $\mathcal{H}_{TOP}$
  - Algorithm $\mathcal{H}_{Greedy}$
  - Algorithm $\mathcal{H}_{Mixed}$
- Every time each of them is executed, it produces as output q node-disjoint cycles

Algorithm $\mathcal{H}_{TSPN}$

- Based on the work of Kim et al.[1]
  - q-Travelling Salesman Problem with Neighborhood (q-TSPN)
  - no battery constraints and no priorities for sites

q-Travelling Salesman Problem with Neighborhood

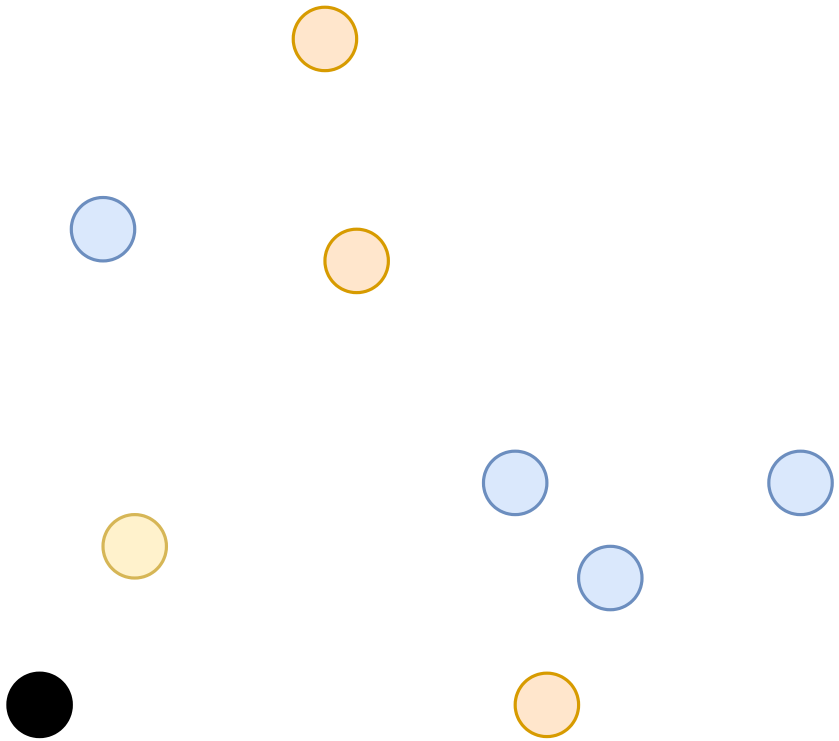
- Goal: find q rooted tours such that
  - each tour starts from a distinct root
  - each node is visited
  - length of the longest tour is minimized

[1] Kim, Donghyun, et al. "On theoretical trajectory planning of multiple UAVs to minimize latency in search-and-reconnaissance operations." IEEE transactions on mobile computing 16.11 (2017): 3156-3166.

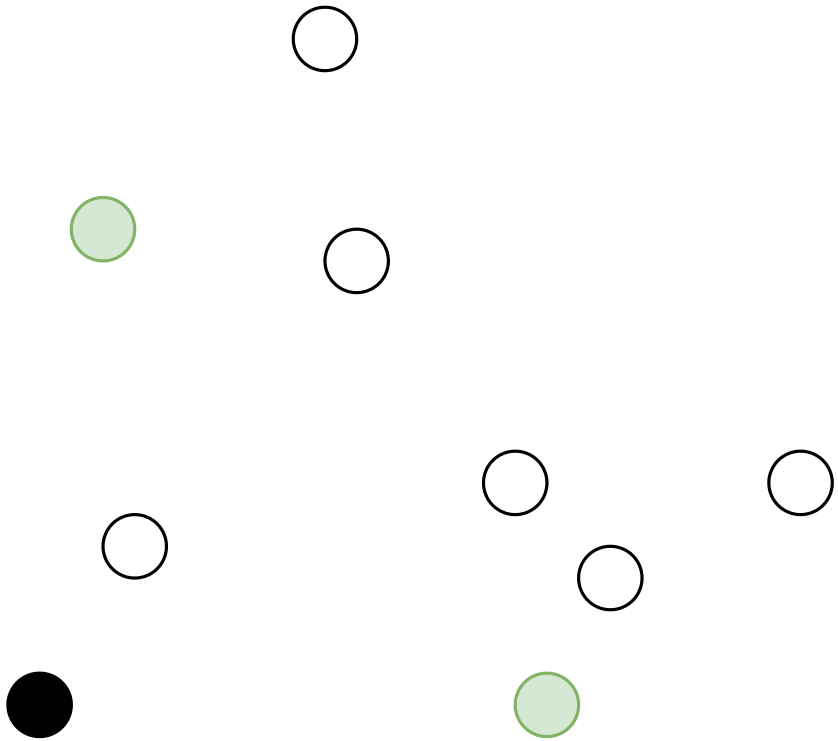
Algorithm $\mathcal{H}_{TSPN}$

- Algorithm (adapted to our problem)
  - Select an initial set of q nodes via q-center problem approximation
  - Divide graph in three sets respect to priorities
  - Find minimum spanning tree starting from the one with maximum priority
  - Find TSP (Christofides's approximation algorithm)

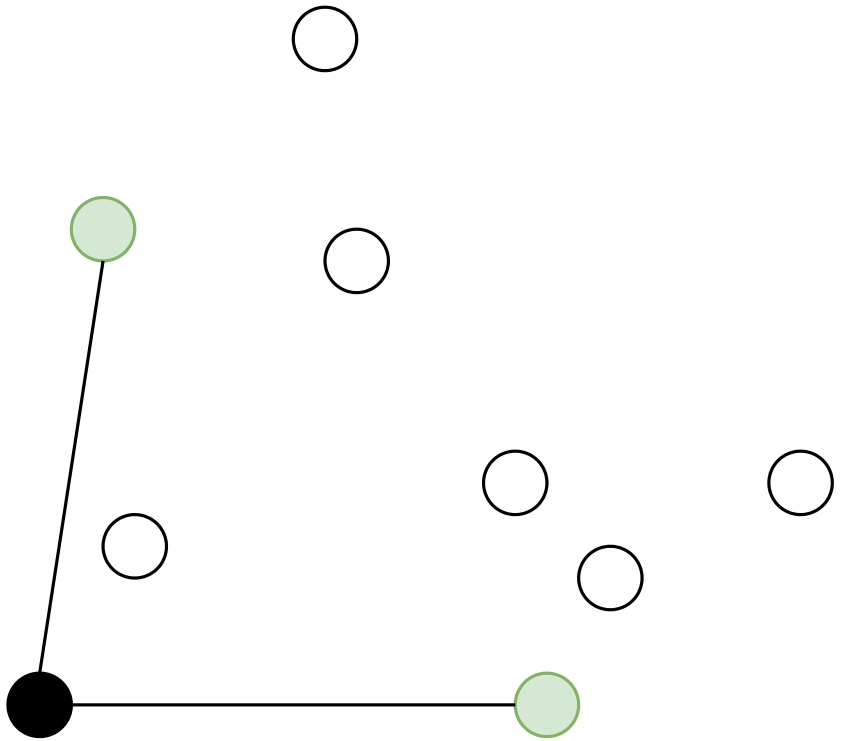
Algorithm $\mathcal{H}_{TSPN}$



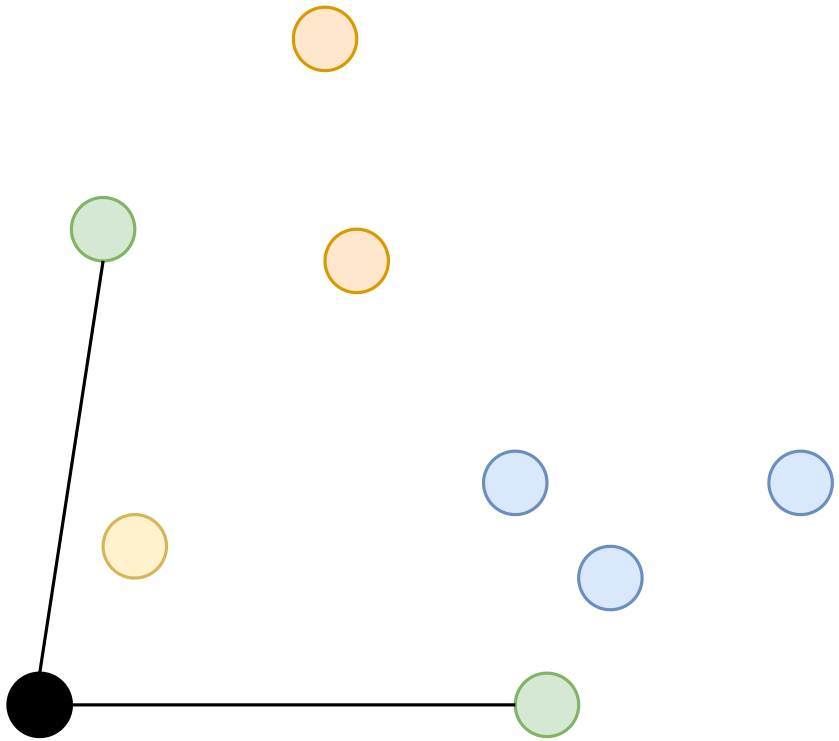
Algorithm $\mathcal{H}_{TSPN}$



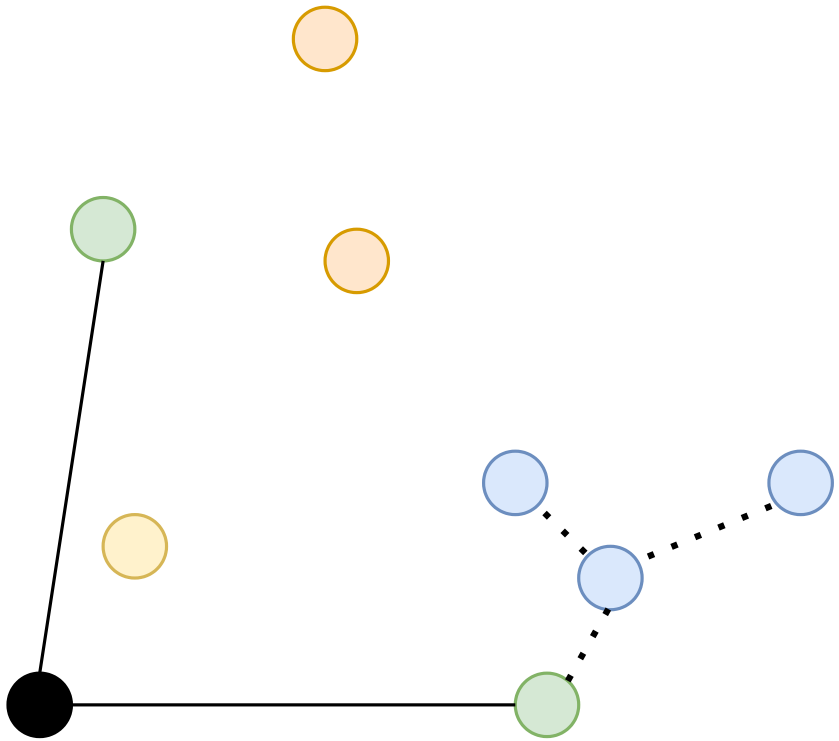
Algorithm $\mathcal{H}_{TSPN}$



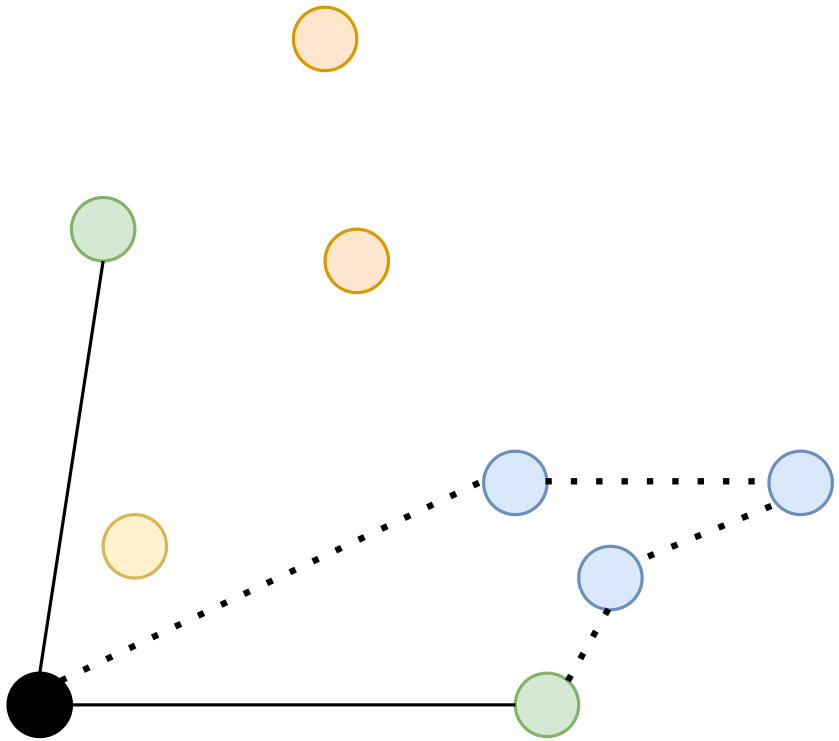
Algorithm $\mathcal{H}_{TSPN}$



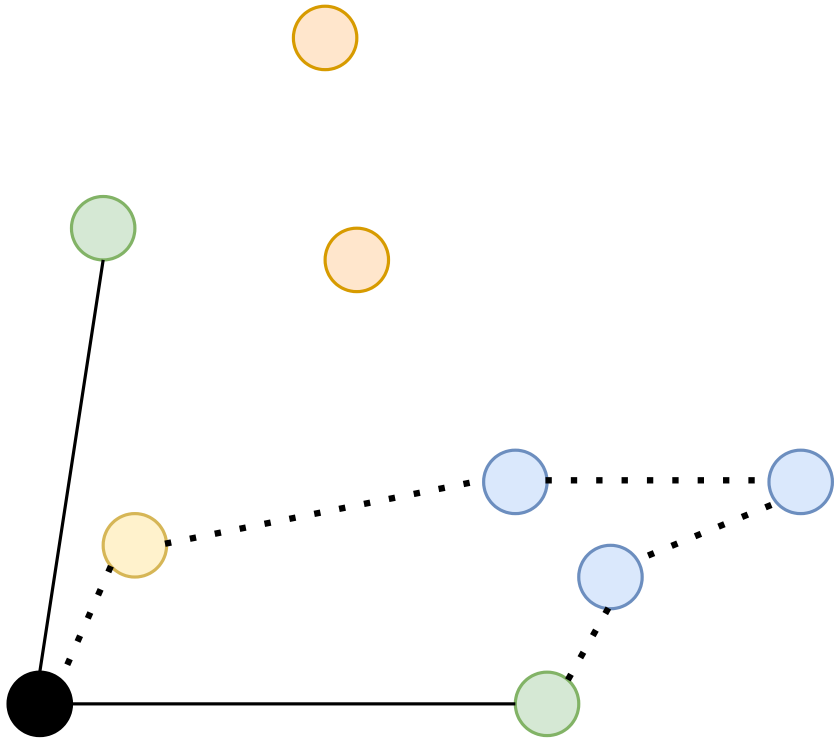
Algorithm $\mathcal{H}_{TSPN}$



Algorithm $\mathcal{H}_{TSPN}$



Algorithm $\mathcal{H}_{TSPN}$



TOP based algorithm $\mathcal{H}_{TOP}$

Team Orienteering Problem (TOP)

- Complete node- and edge-weighted graph $G = (V, E, w, dist)$
- Vertex v_0 is the starting and ending point of each tour
- Each node as a weight $w : V \rightarrow \mathbb{R}^+$
- Budget B
- **Goal:** find q tours rooted in v_0 such that
  - each tour has length maximum B
  - maximize the total gain (weights of covered nodes)

TOP based algorithm $\mathcal{H}_{TOP}$

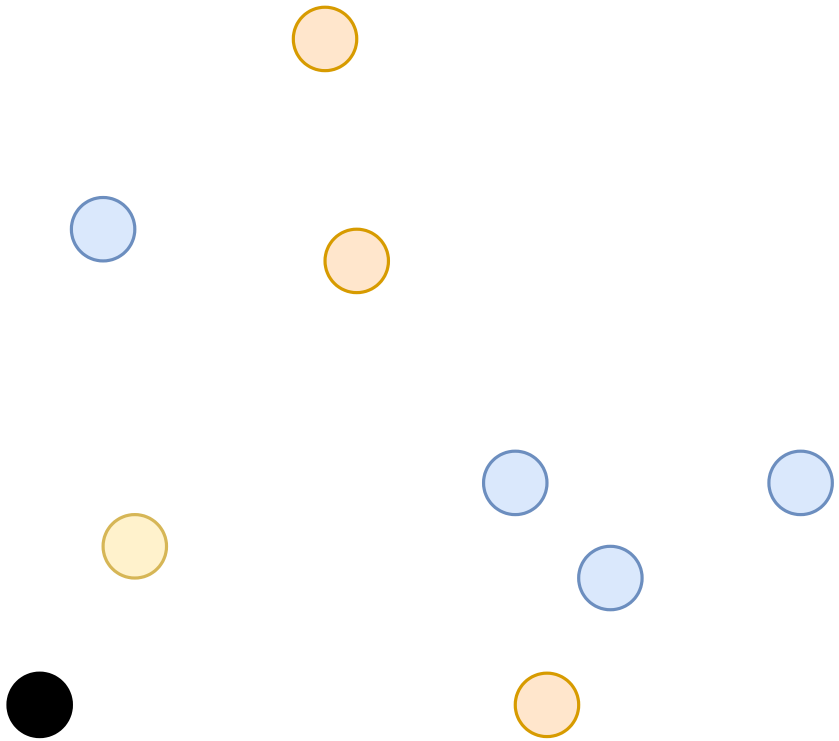
- TOP aims at maximizing the profit
- Does not require to cover all nodes
- Equivalent to the first round of our problem (with some adjustment!)
- If $w = 1$ then $\rightarrow$
  Objective: maximize the number of covered sites (in one round!)

TOP based algorithm $\mathcal{H}_{TOP}$

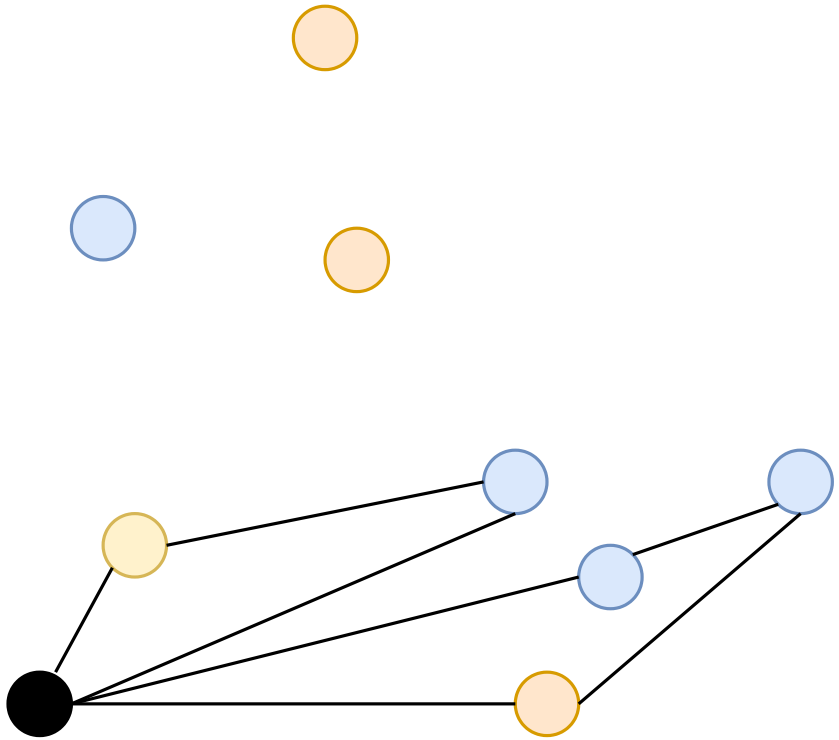
- (Adaptation) Run until all nodes are visited
- (Adaptation) We set for each node profit == priorities
  - $\rightarrow$ minimize weighted latency
  - choosing high priority sites
  - choosing larger number of low priority sites
- We use algorithm by Vansteenwegen et al.[2]
  - fastest known heuristic to solve TOP

[2]Vansteenwegen, et al. "Metaheuristics for tourist trip planning." Metaheuristics in the service industry. Springer, Berlin, Heidelberg, 2009. 15-31.

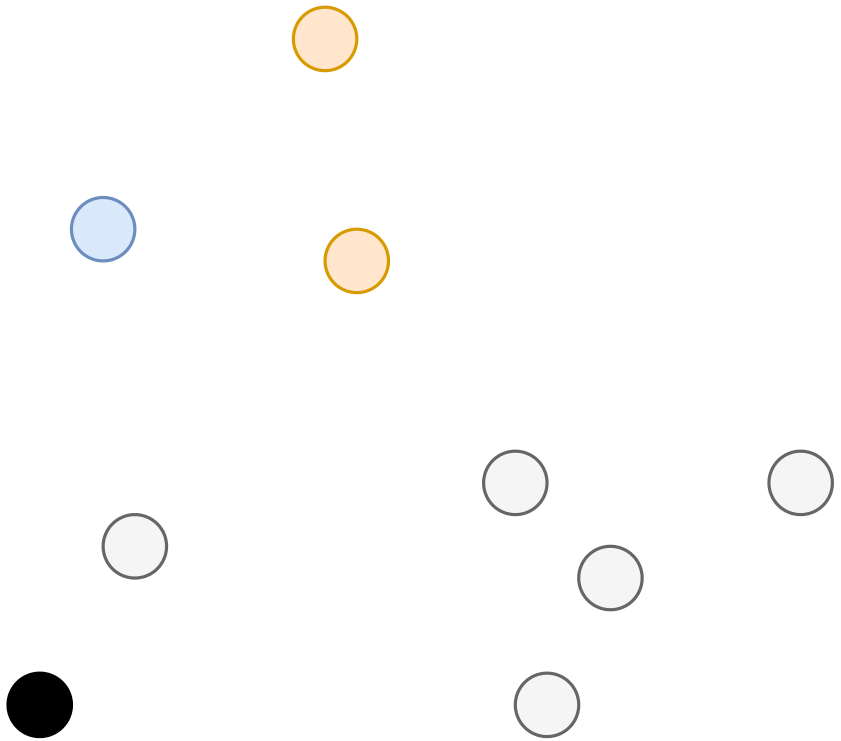
Algorithm $\mathcal{H}_{TOP}$



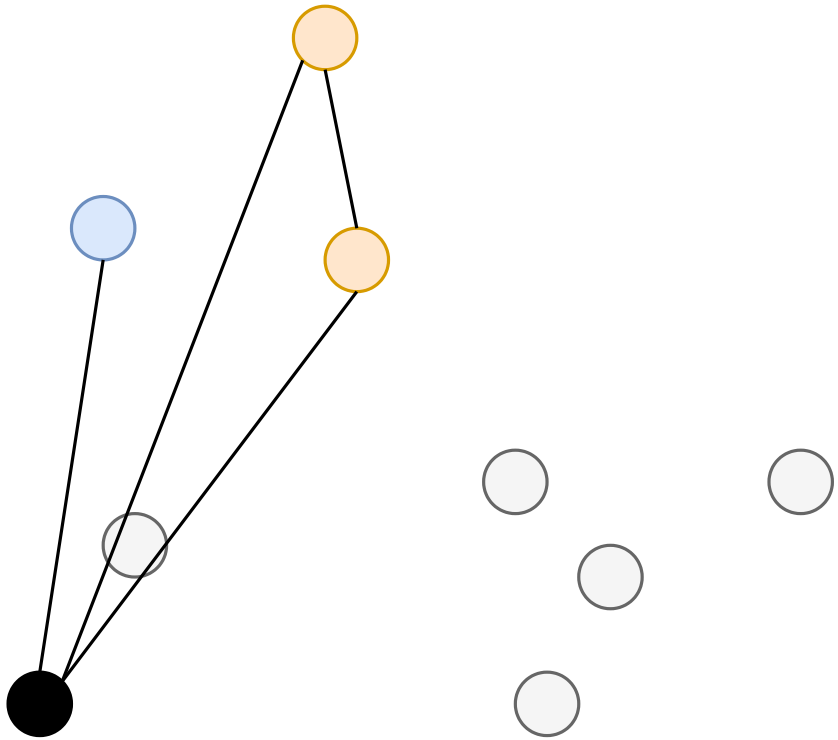
Algorithm $\mathcal{H}_{TOP}$



Algorithm $\mathcal{H}_{TOP}$



Algorithm $\mathcal{H}_{TOP}$



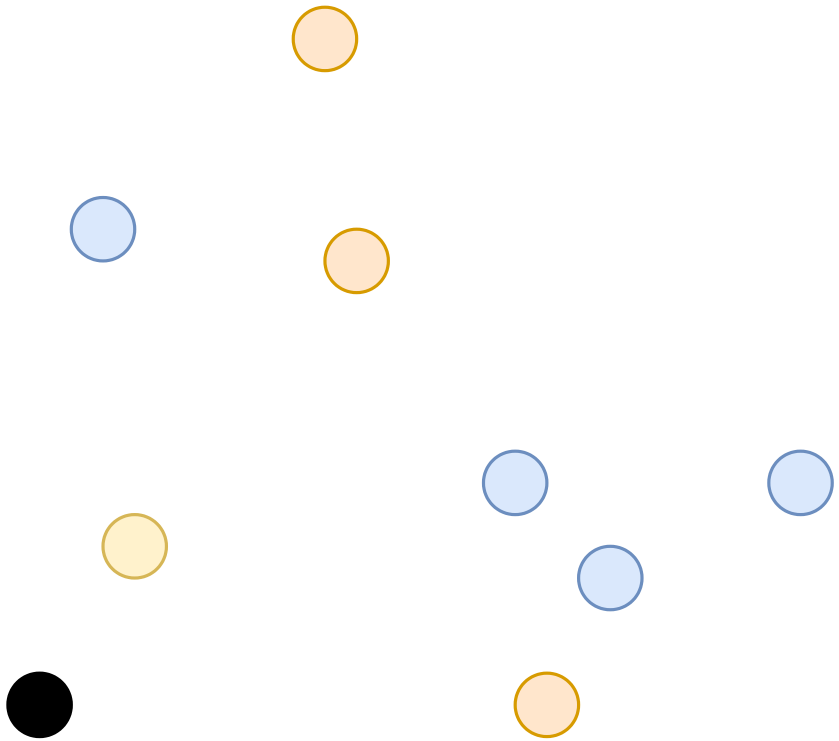
Greedy based algorithm $\mathcal{H}_{Greedy}$

- Finds q cycles according to a greedy approach
- Extends q cycles simultaneously
- At each step chooses the node that maximizes:

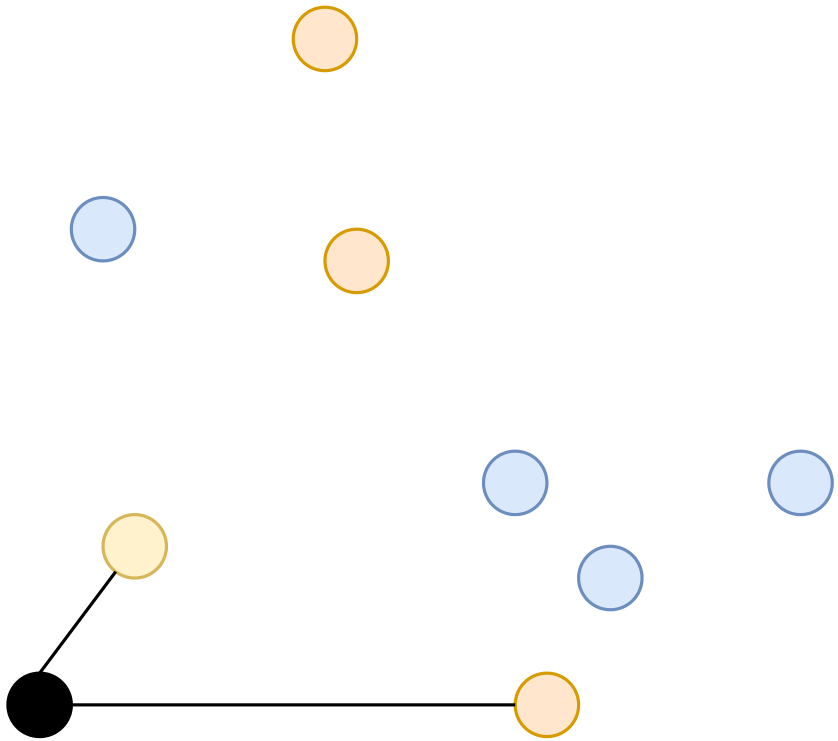
$$\frac{p(v_{next})}{dist(v_{last}, v_{next})}$$

- Prefers sites with high priority without going too far

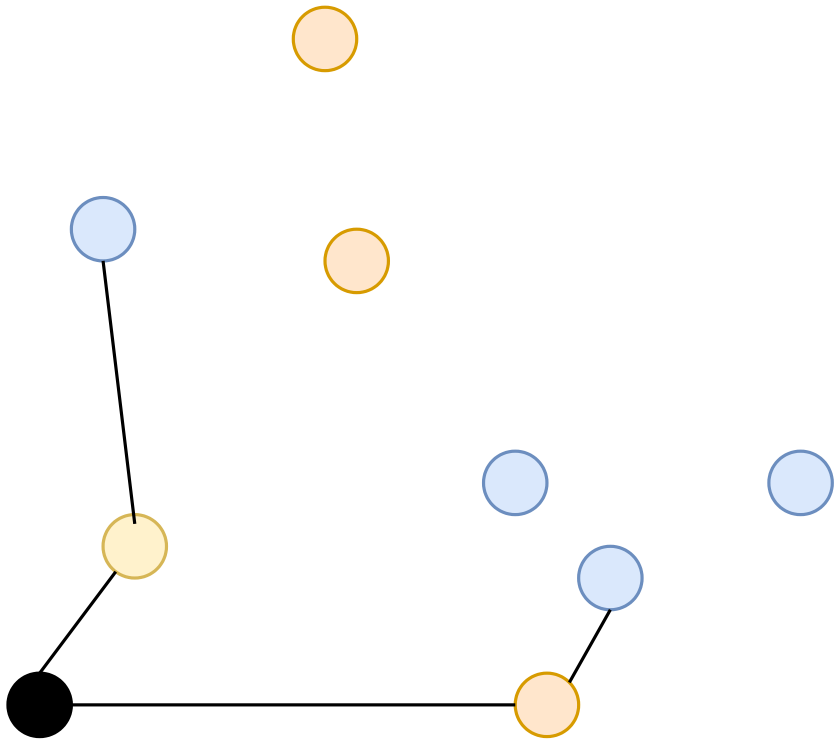
Algorithm $\mathcal{H}_{Greedy}$



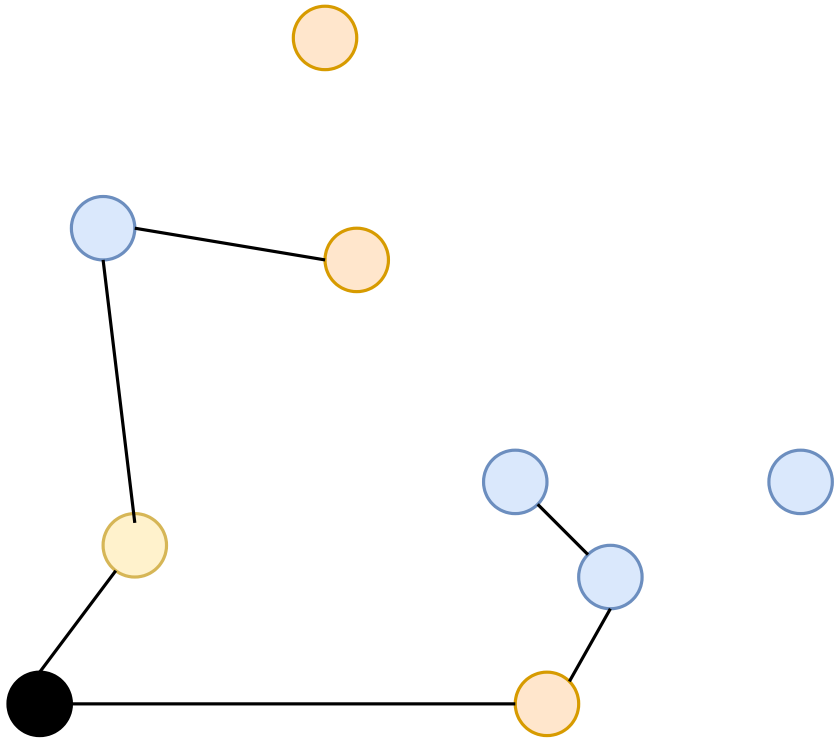
Algorithm $\mathcal{H}_{Greedy}$



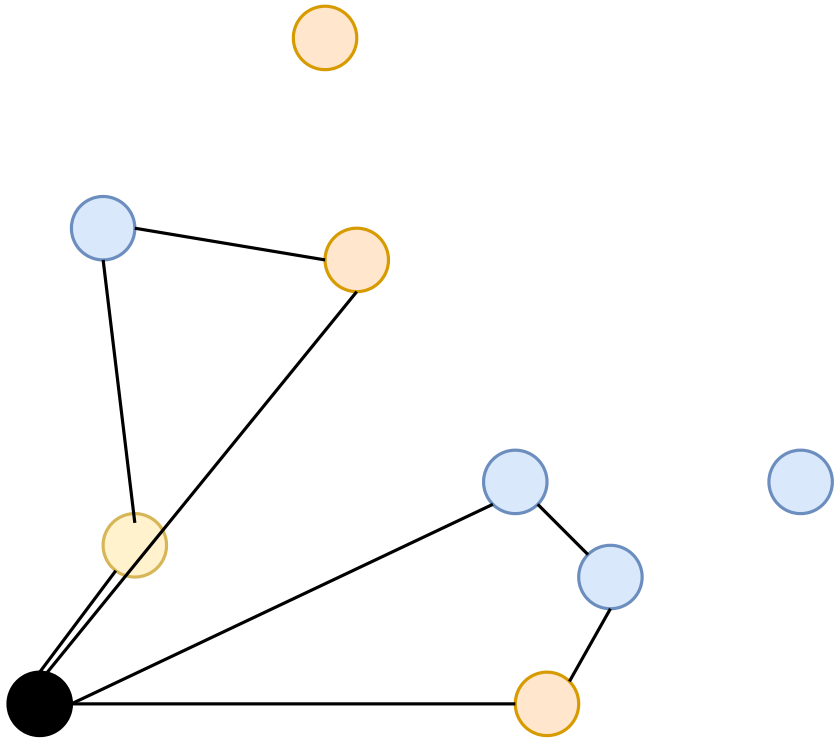
Algorithm $\mathcal{H}_{Greedy}$



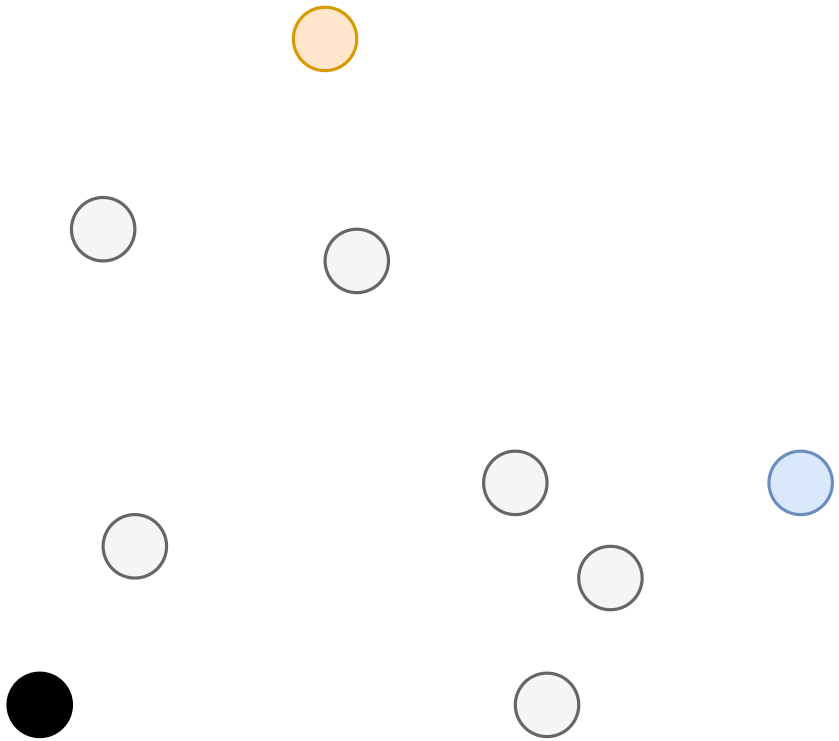
Algorithm $\mathcal{H}_{Greedy}$



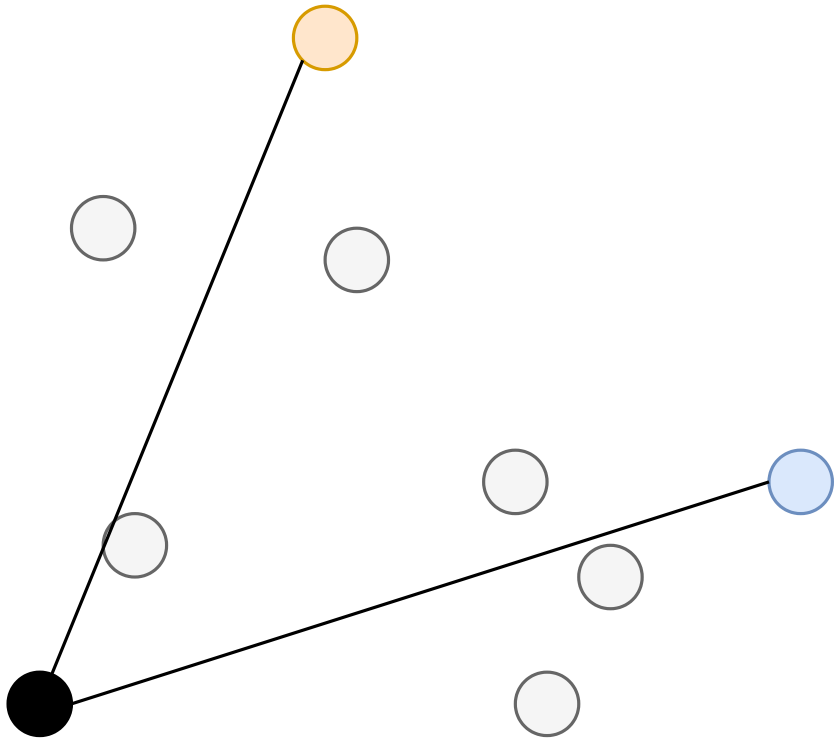
Algorithm $\mathcal{H}_{Greedy}$



Algorithm $\mathcal{H}_{Greedy}$



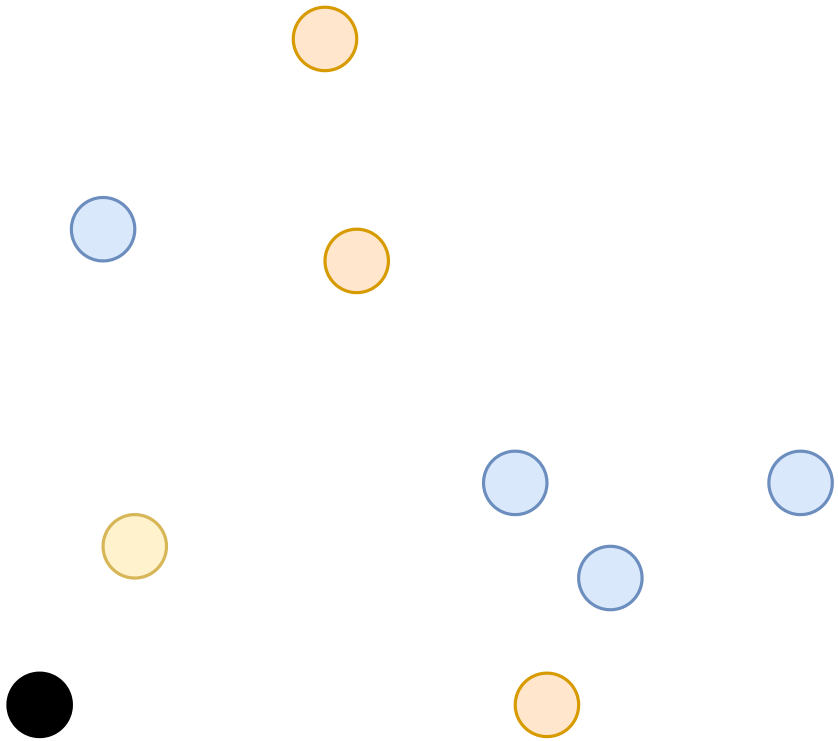
Algorithm $\mathcal{H}_{Greedy}$



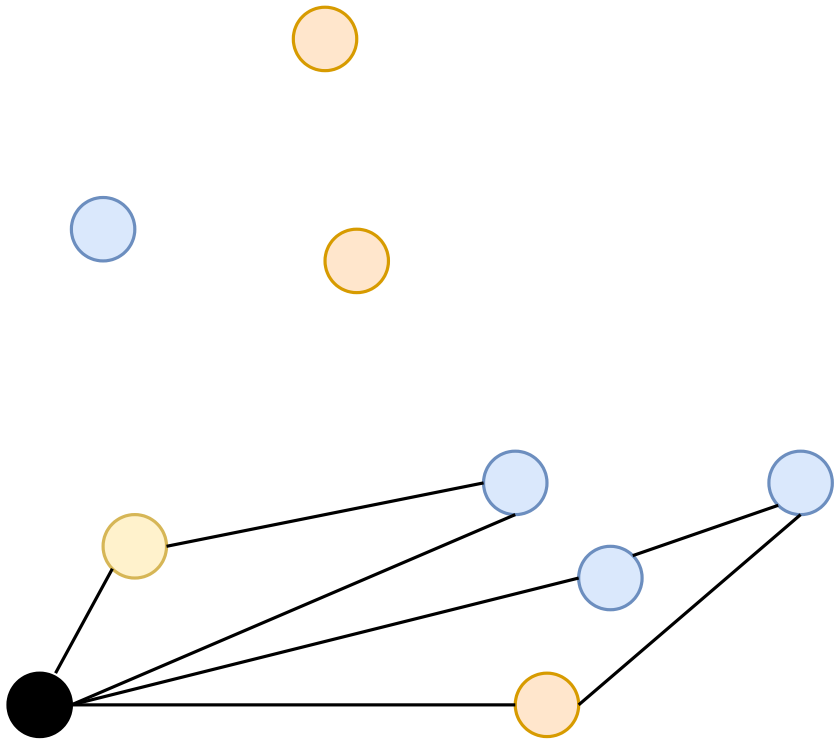
Algorithm $\mathcal{H}_{mixed}$

- Observe that TOP is not designed to be iterated
  - choose closest sites to v_0 to maximize total number
- We formulate $\mathcal{H}_{mixed}$
  - first iteration: $\mathcal{H}_{TOP}$
  - successive ones: $\mathcal{H}_{Greedy}$

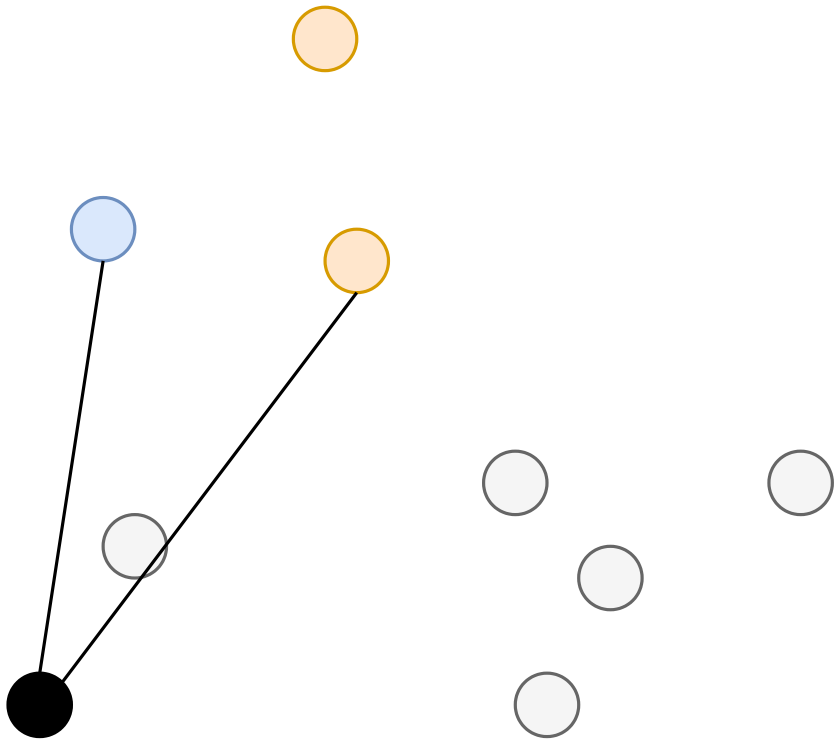
Algorithm $\mathcal{H}_{Mixed}$



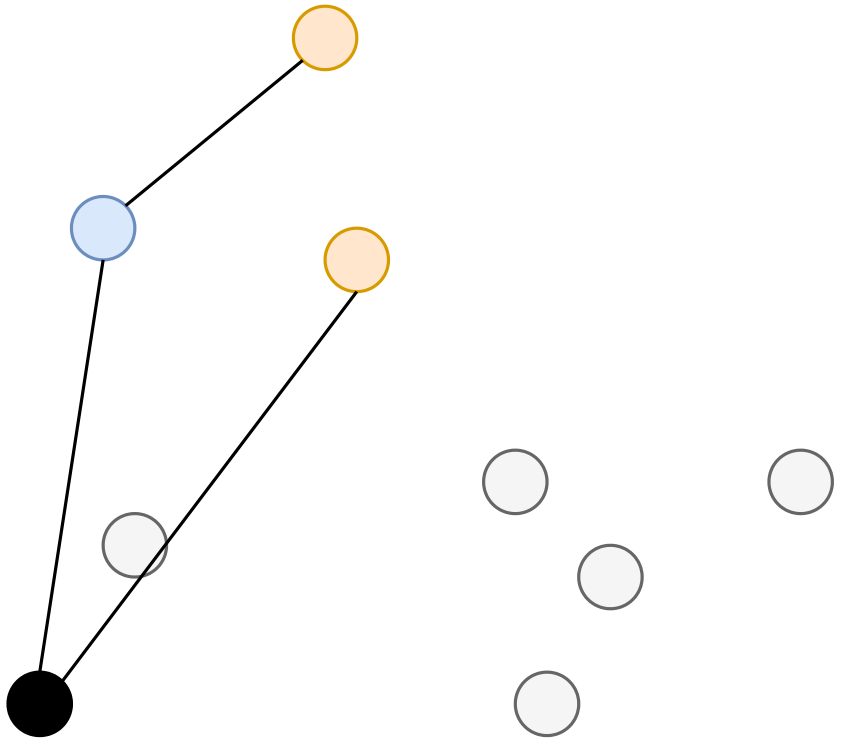
Algorithm $\mathcal{H}_{Mixed}$



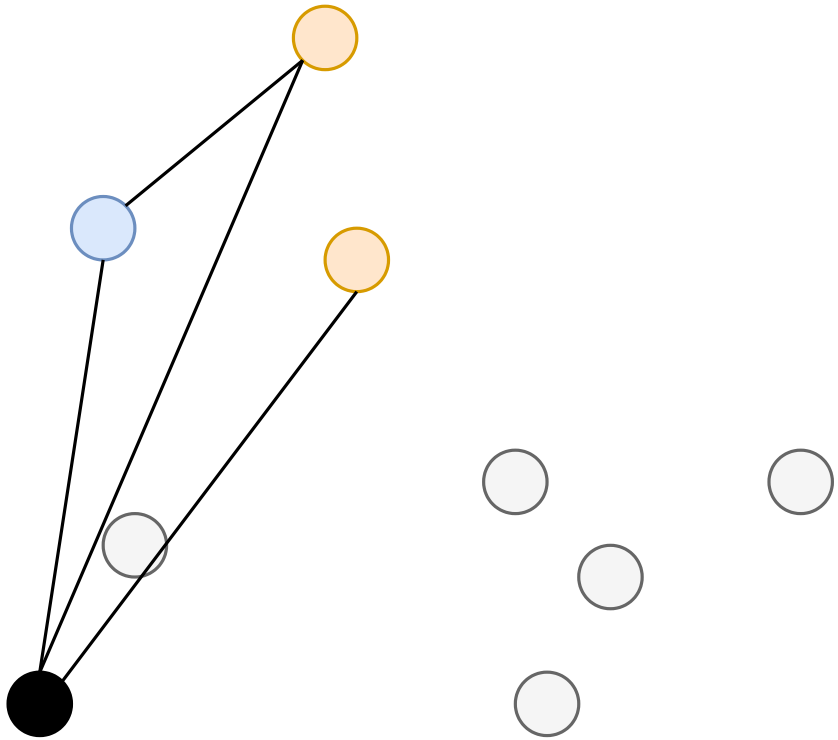
Algorithm $\mathcal{H}_{Mixed}$



Algorithm $\mathcal{H}_{Mixed}$

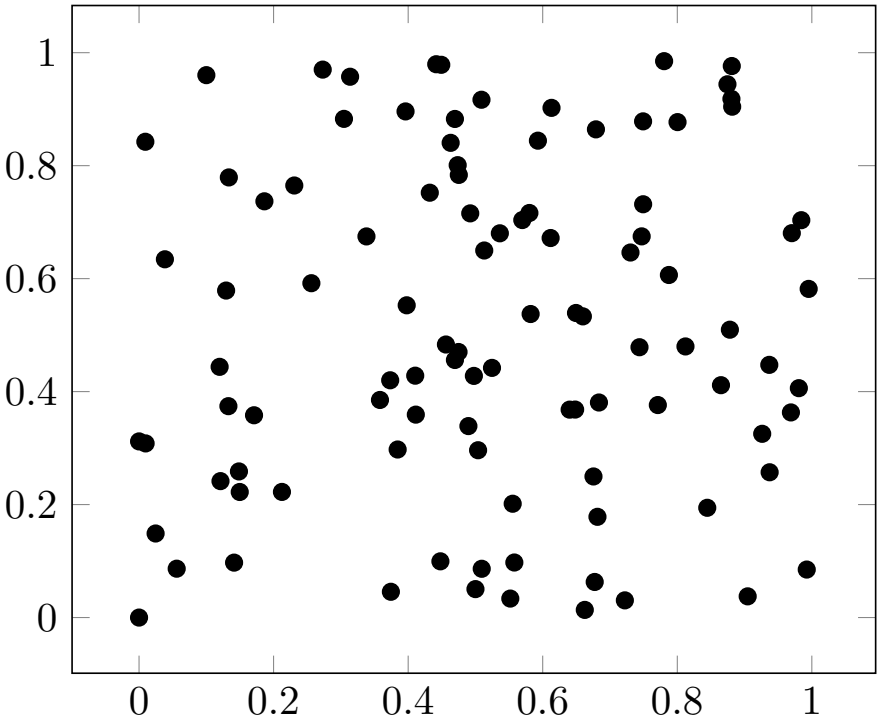


Algorithm $\mathcal{H}_{Mixed}$

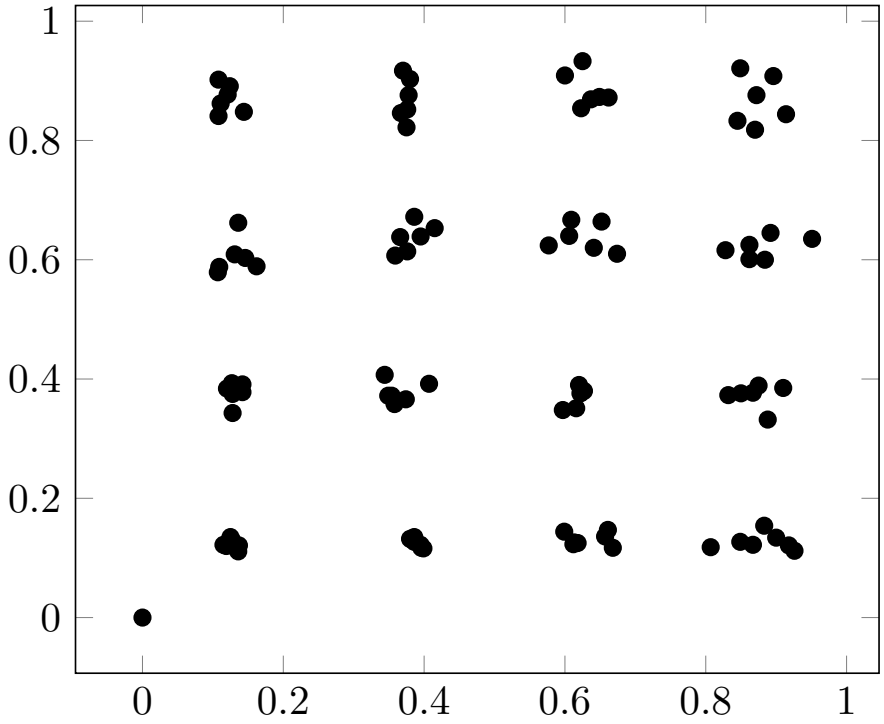


Outline

Experiments



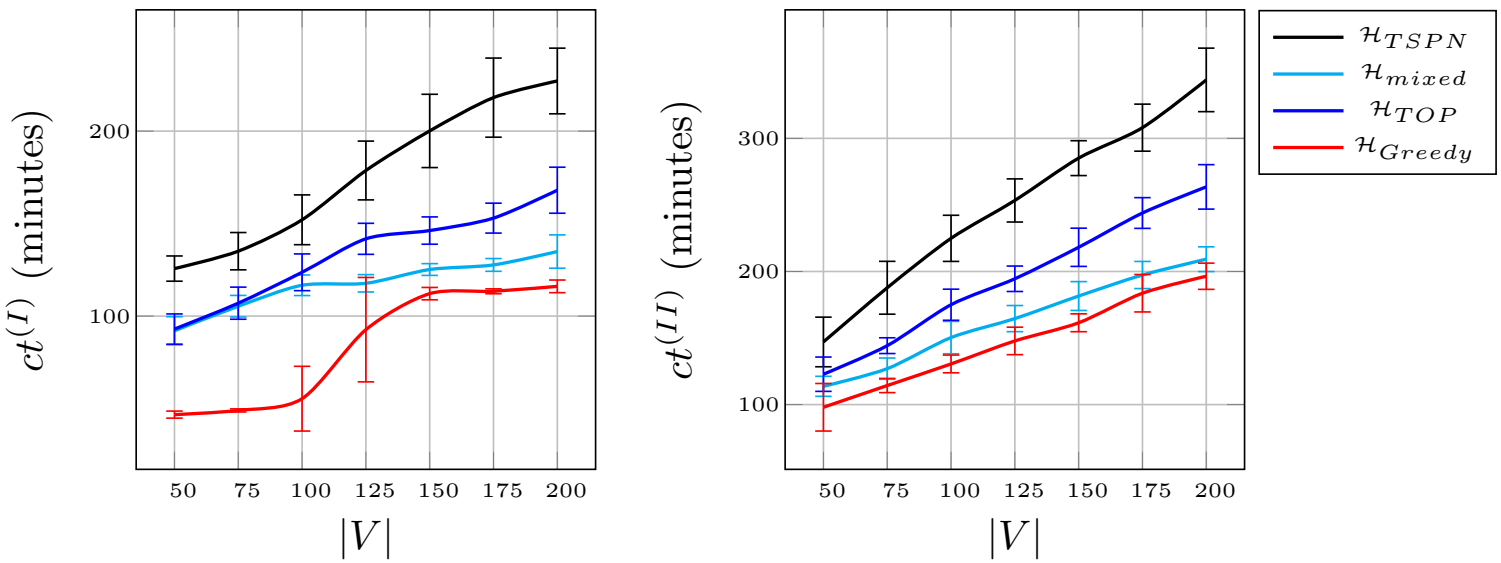
(d) Uniform distribution



(e) Poisson distribution

Two examples of instances in which $n = 100$ sites are positioned on a squared area

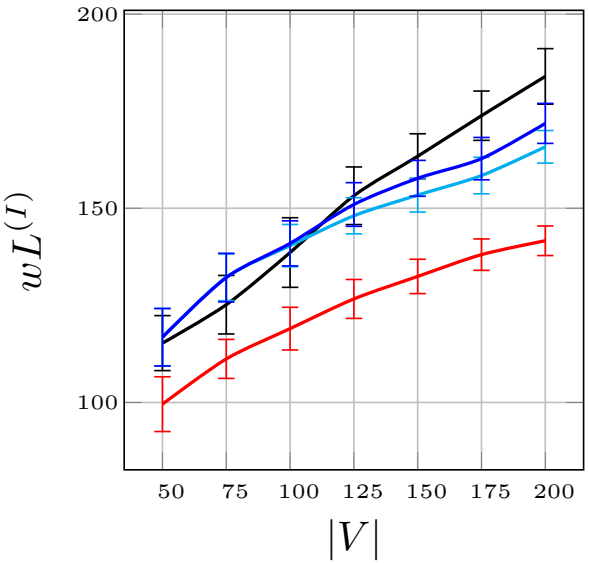
Experiments (cont.)



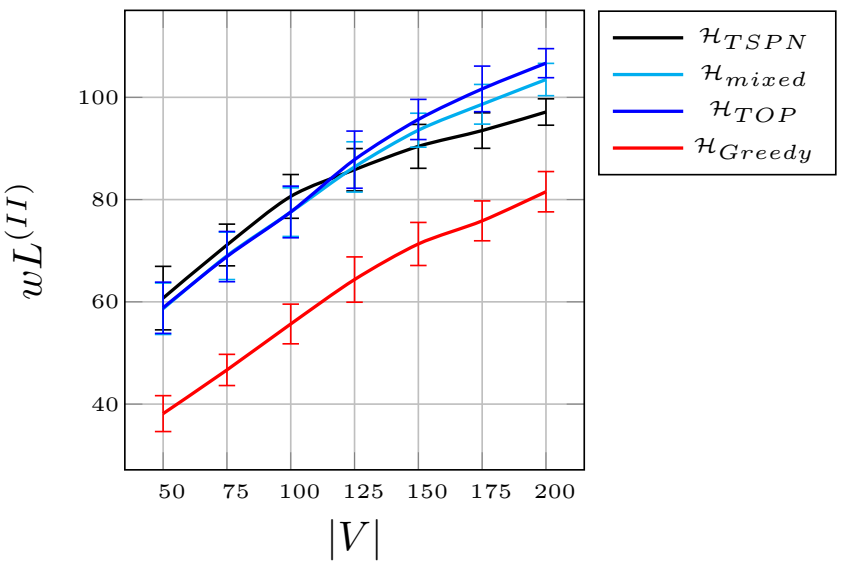
(f) $q = 20, B = 50$ (g) $q = 10, B = 50, p = 0.25$

Average completion time in the first and second scenario as a function of the number of nodes in the graph

Experiments (cont.)



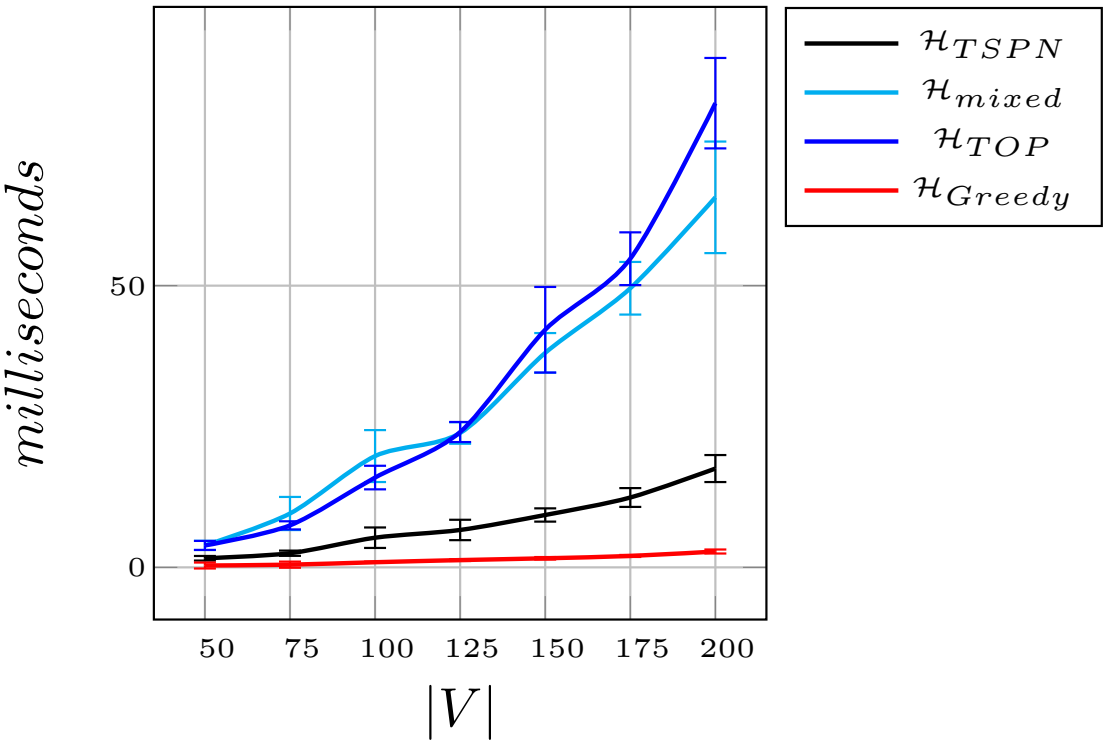
(h) $q = 20, B = 50$



(i) $q = 10, B = 50, p = 0.25$

Average weighted latency in the first and second scenario as a function of the number of nodes in the graph

Experiments (cont.)



Execution time (average per round) $q = 10, B = 50$, in the second scenario with $p = 0.25$ with sites uniformly distributed

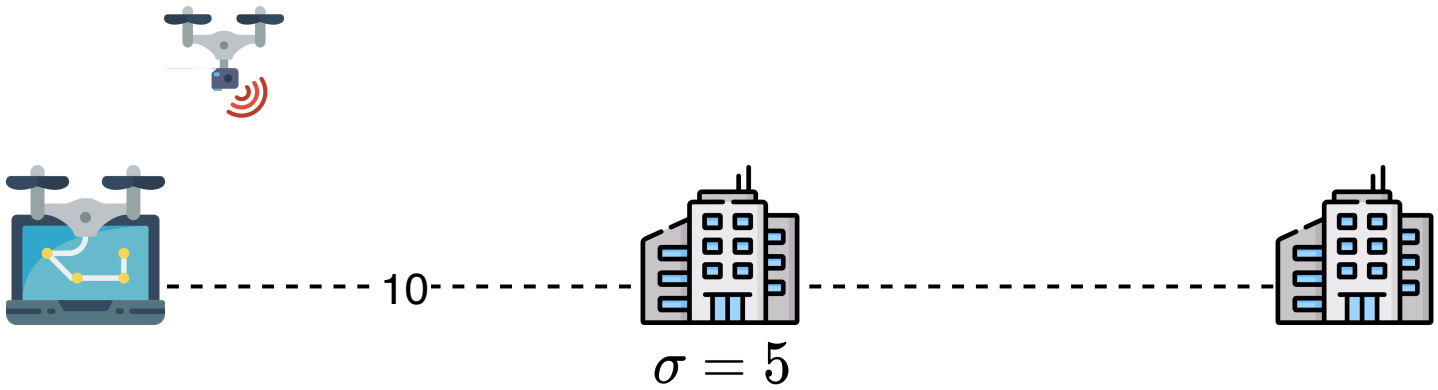
Outline

Future Work

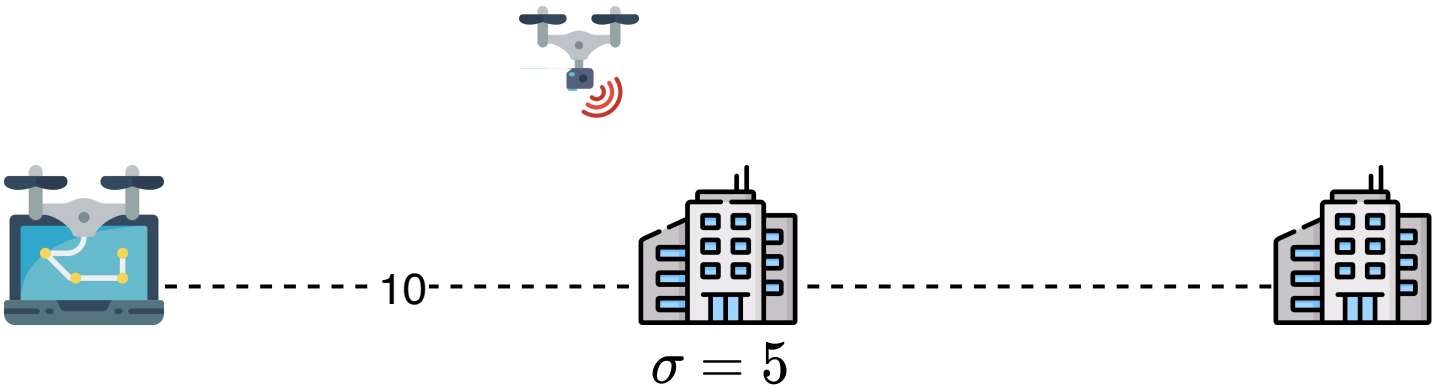
Future works:

- Consider "partially overflight" nodes
- Introducing cooperation
  - UAV-UAV
  - Human-UAV
- Considering more operations centre
- Determining a tight approximation ratio

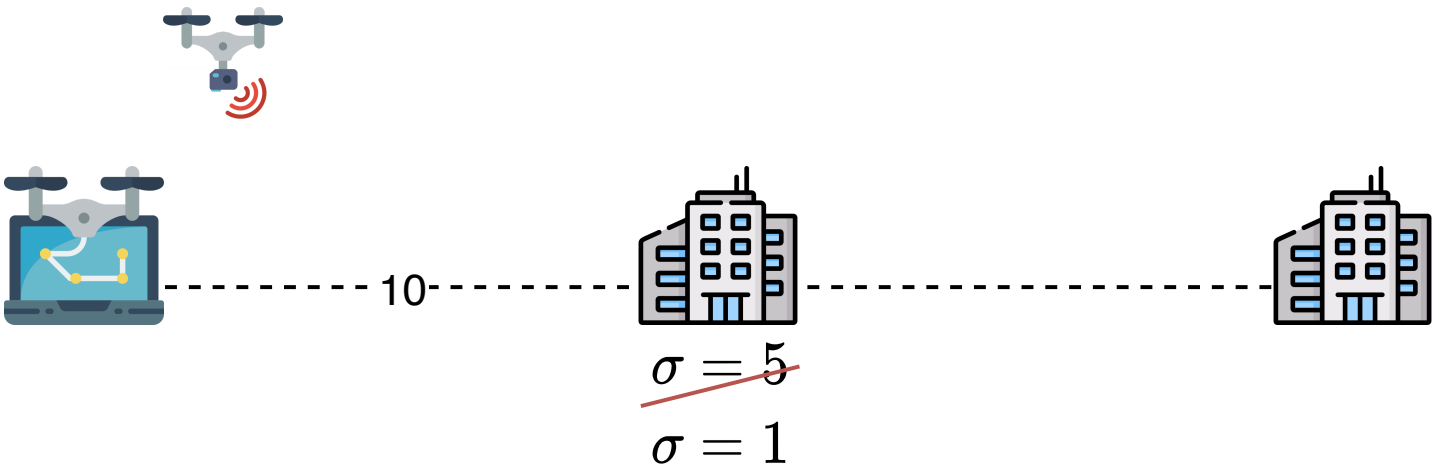
“Partially overflight” nodes



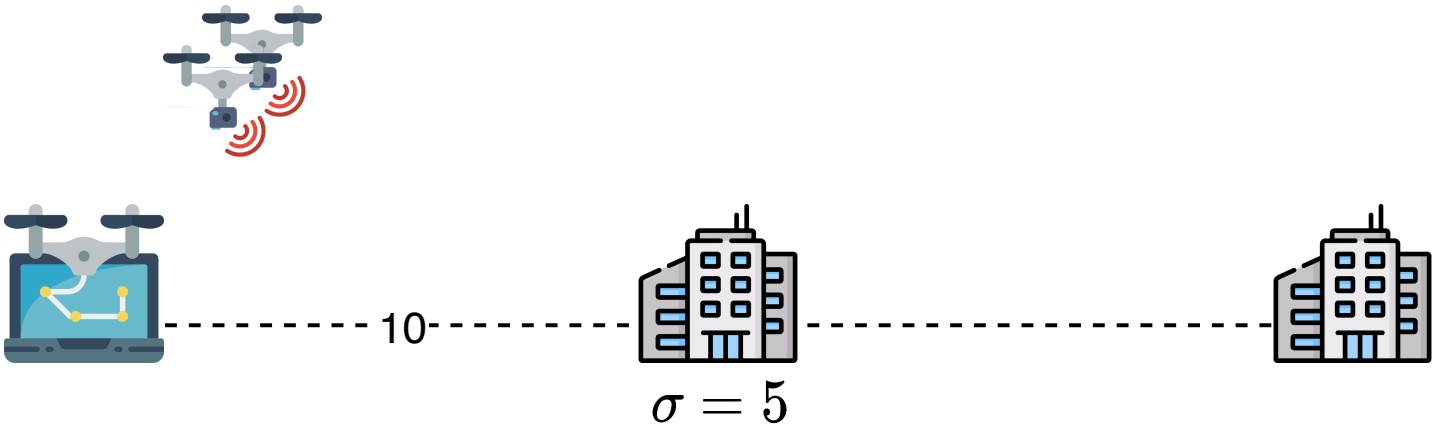
“Partially overflight” nodes



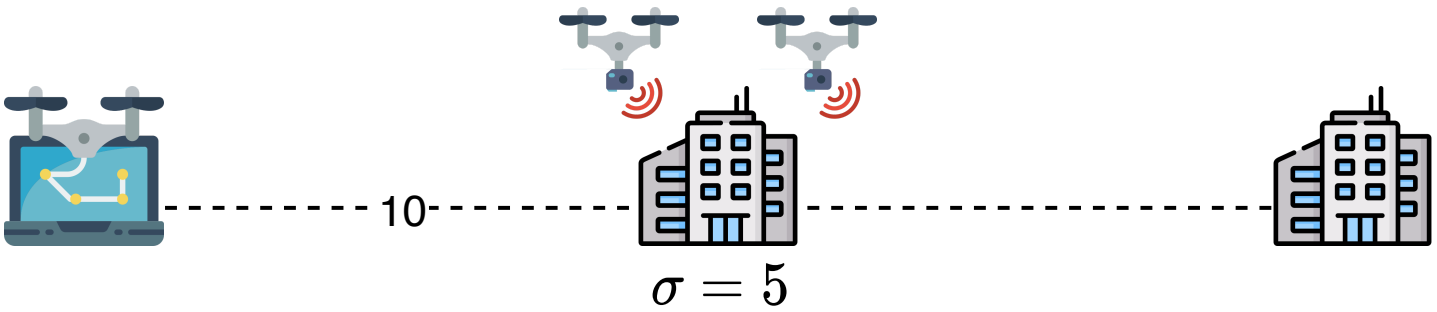
“Partially overflight” nodes



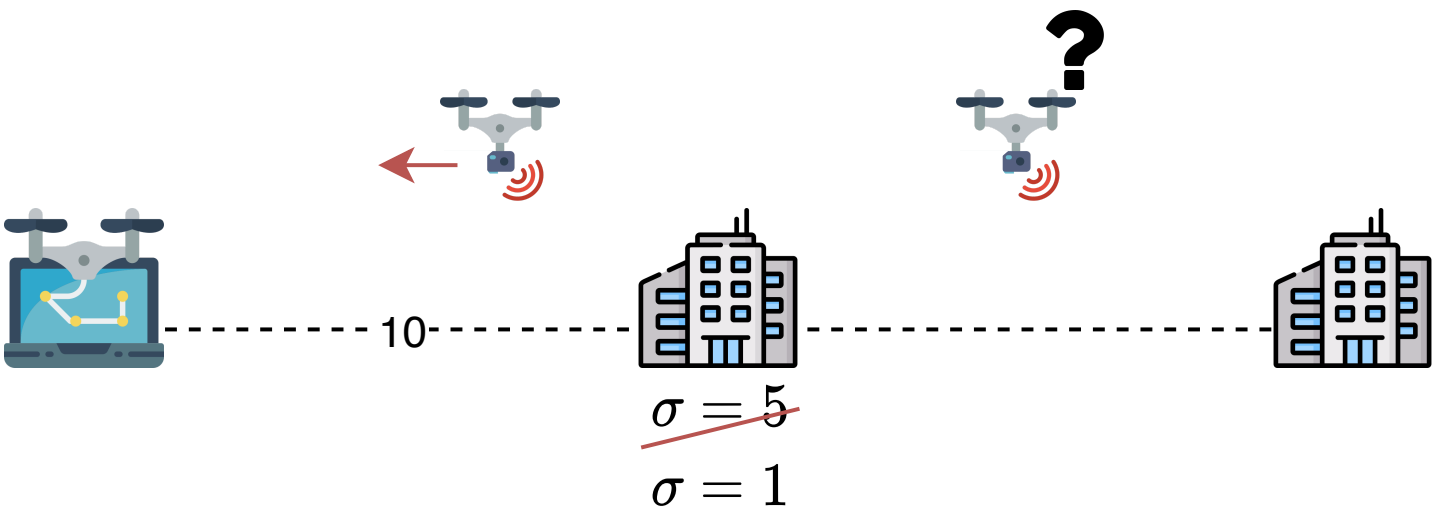
Cooperation: Scenario 1



Cooperation: Scenario 1



Cooperation: Scenario 2



Multiple operations centre