Optimal and Heuristic Algorithms for Data Collection by Using an Energy- and Storage-Constrained Drone

Francesco Betti Sorbelli[1] Alfredo Navarra[1] Lorenzo Palazzetti [2] Cristina M. Pinotti[1]
Giuseppe Prencipe[3]

[1]University of Perugia, Perugia, Italy {francesco.bettisorbelli, alfredo.navarra, cristina.pinotti}@unipg.it

[2]University of Florence, Firenze, Italy lorenzo.palazzetti@unifi.it

[3]University of Pisa, Pisa, Italy giuseppe.prencipe@unipi.it

Feb 8th, 2023

Outline

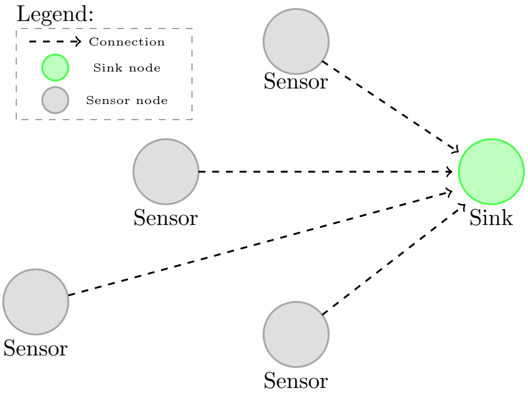
Outline

Data Collection

- **Data collection** is the process by which sensor networks **collect**, and **store data** in a **sink** node for answering to external queries

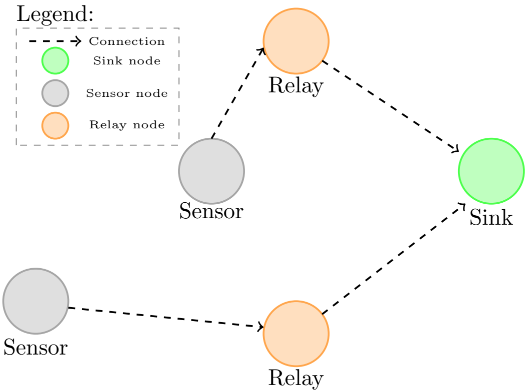
Motivations (1)

- **Single-hop Connections**, as the name implies, assumes direct **one-hop** communication between a sensor and the sink
  - **Not all the sensors are connected via single-hop to the depot/sink that requires the data**



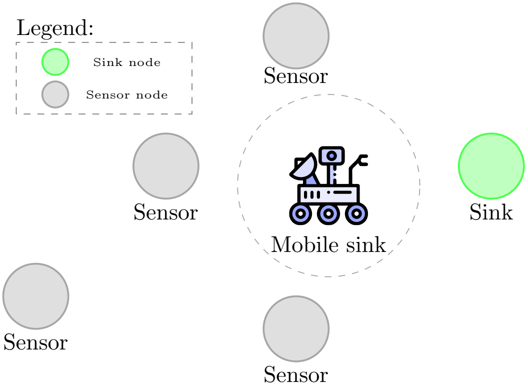
Motivations (2)

- **Multi-hop connection** allows multiple hops for communication among a sensor and the sink
- Packets are forwarded through **in-the-middle** nodes located between a **sensor** and the **sink**
  - **In multi-hop implementation the sensors closer to the depot consume more energy**
  - **Several nodes may be a *bottleneck* for the network**



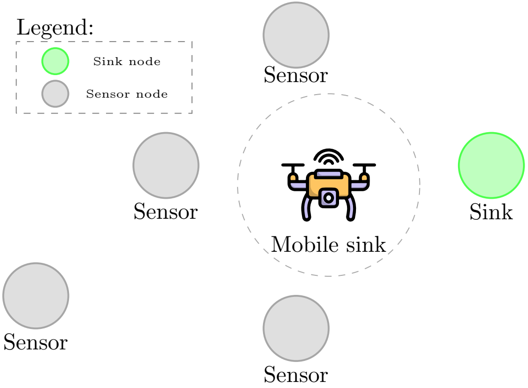
Motivations (3)

- A **ground vehicle** as **mobile sink** allows to collect data by moving close to the sensors
- After collecting, the ground vehicle transfers the data to the sink node, i.e., the robot is a **data carrier**
  - **Ground vehicles movements could be affected by the presence of obstacles**
  - **Ground vehicle speed is relatively low**
  - **Ground vehicles are constrained in the storage (when collecting data)**



Motivations (4)

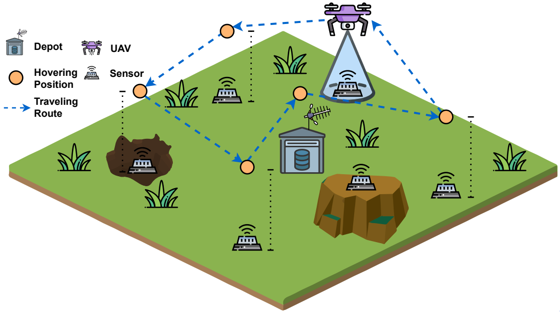
- Using a **flying vehicle** as **mobile sink**
  - **Drone is not affected by eventual obstacles on the terrain**
  - **Drone speed is higher than that of ground vehicles**
  - **Drones are constrained in both the energy (when flying and hovering), and the storage (when collecting data)**



Data Collection Scenario: The Premise

- The **drone** has to perform a **mission** (route) to/from the **depot**, with the aim to selectively collect the data from the sensors via single-hop close to the sensors
- The drone ***cannot collect*** the data from all the deployed sensors due limited **energy battery** (*flying and hovering have an impact*) and **storage space** (*the larger is the data, the more is its occupancy*)
- The **ground sensors** generate data, and are characterized by a relevance, e.g., *freshness of the data*

⇒ **Goal: collect the most relevant data, while ensuring that the mission energy cost does not exceed the battery budget, and the total collected data does not exceed the storage limit**



Contributions

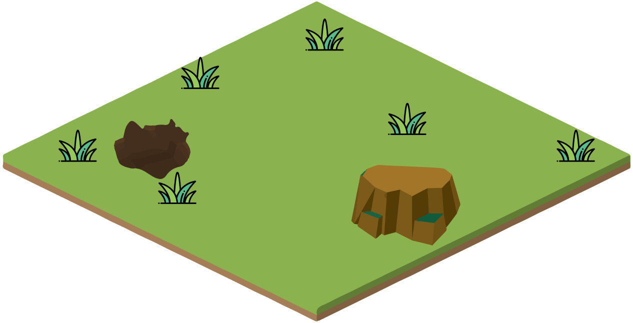
The contributions of this paper are summarized as follows:

- We define a **novel optimization problem**, called *Single-drone Data-collection Maximization Problem* (SDMP), and prove it to be NP-hard
- We devise an **Integer Linear Programming** (ILP) formulation for optimally solving SDMP, as well as an **approximation** plus two **heuristic** algorithms for obtaining sub-optimal solutions
- We **evaluate the performance** of our algorithms on randomly generated **synthetic data**

Outline

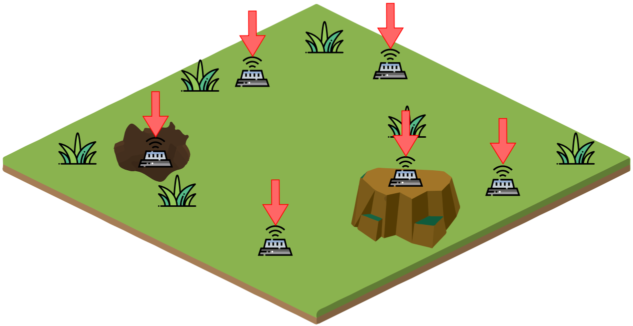
Data Collection Area

- The data collection area is defined by a field F
  - 3D plane
  - the surface is **not flat**
  - whose center is $O = (0, 0, 0)$



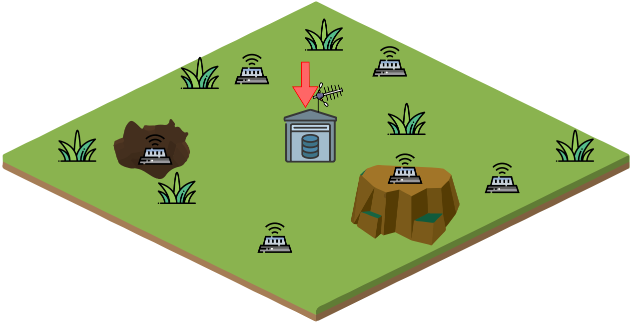
Data Collection Area

- The data collection area is defined by a field F
  - 3D plane
  - the surface is **not flat**
  - whose center is $O = (0, 0, 0)$
- A set $V = \{v_1, \ldots, v_n\}$ of n heterogeneous *ground sensors*



Data Collection Area

- The data collection area is defined by a field F
  - 3D plane
  - the surface is **not flat**
  - whose center is $O = (0, 0, 0)$
- A set $V = \{v_1, \ldots, v_n\}$ of n heterogeneous *ground sensors*
- A **depot** placed at the center $O = (0, 0, 0)$

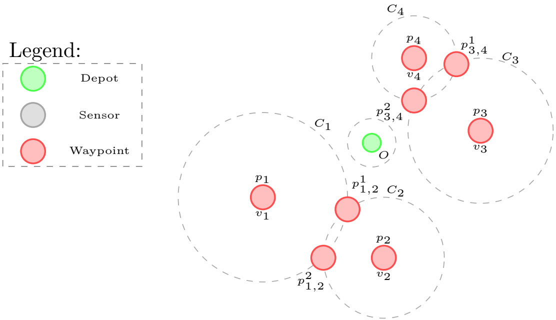


Sensors

- Each sensor $v_i \in V$
  - Is randomly deployed in F
  - Its position is defined by a 3D coordinate (x_i, y_i, z_i) appropriately chosen with respect to O
  - Has a **local storage** of size W_i
  - Perceives physical phenomena, e.g., temperature, pressure, or even pictures or videos, and then generates data to be locally stored
    - Let $0 < w_i \leq W_i$ be the *size* of this generated data
- The data is modeled by a **relevance**, and relevant data should be prioritized when ground sensors have to start the data transferring
  - This is modeled by associating a **reward** $r_i > 0$ to each sensor v_i
  - *The more is the reward, the more relevant is to off-load the data from the sensor*

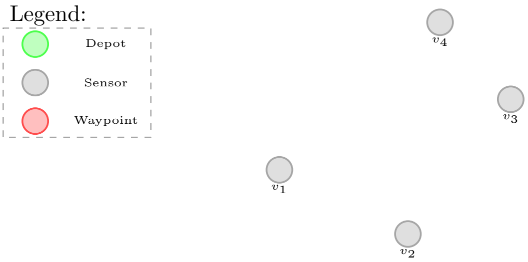
Drone (1)

- The external device which collects the data from the sensors is a **drone** denoted as D
- The drone flies at a fixed altitude h above the ground, and it has a **communication range** with radius R
  - It can collect data from a sensor v_i if $\|D - v_i\|_2 \leq R$, i.e., their **Euclidean distance is within the communication range**
  - For each sensor v_i, we define an **admissible region** C_i in which the drone can actually communicate with it
- The drone is allowed to fly only at specific locations over F, called **waypoints**, represented by a set P



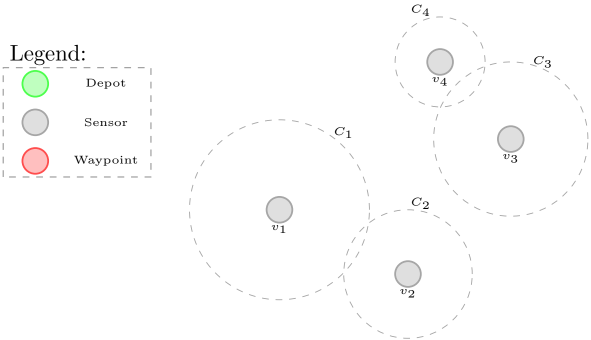
Example

- Given a set of sensor $V = \{v_1, v_2, v_3, v_4\}$ we compute:



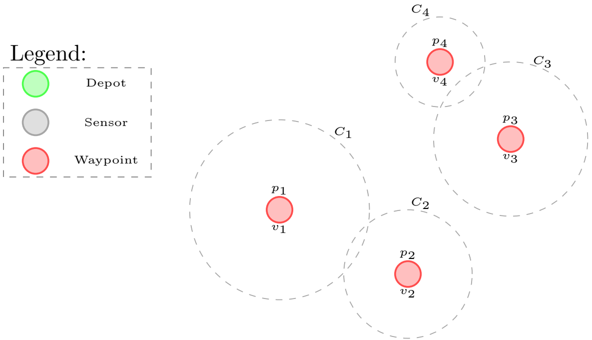
Example

- Given a set of sensor $V = \{v_1, v_2, v_3, v_4\}$ we compute:
  1. The **admissible region** C_i in which the drone can actually communicate with the sensors



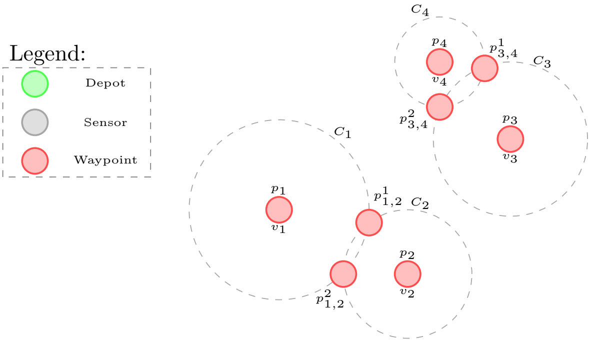
Example

- Given a set of sensor $V = \{v_1, v_2, v_3, v_4\}$ we compute:
  1. The **admissible region** C_i in which the drone can actually communicate with the sensors
  2. **All projected sensors' positions** at height h



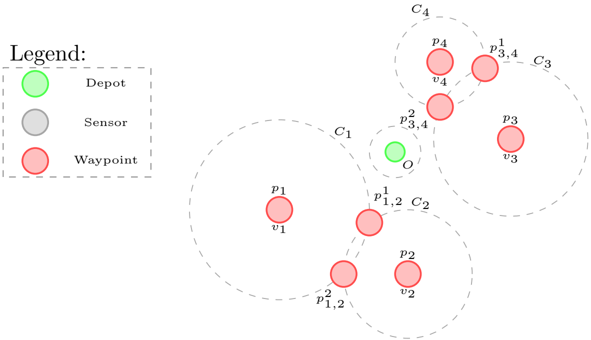
Example

- Given a set of sensor $V = \{v_1, v_2, v_3, v_4\}$ we compute:
  1. The **admissible region** C_i in which the drone can actually communicate with the sensors
  2. **All projected sensors' positions** at height h
  3. For each pair of sensors v_i and v_j, we add in P all the **intersection points** $p^1_{i,j}$ and $p^2_{i,j}$ between C_i and C_j



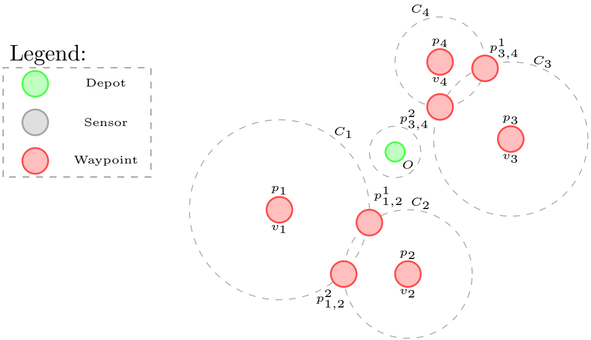
Example

- Given a set of sensor $V = \{v_1, v_2, v_3, v_4\}$ we compute:
  1. The **admissible region** C_i in which the drone can actually communicate with the sensors
  2. **All projected sensors' positions** at height h
  3. For each pair of sensors v_i and v_j, we add in P all the **intersection points** $p^1_{i,j}$ and $p^2_{i,j}$ between C_i and C_j
  4. The **depot** $O = p_0 \in P$



Example

- Given a set of sensor $V = \{v_1, v_2, v_3, v_4\}$ we compute:
  1. The **admissible region** C_i in which the drone can actually communicate with the sensors
  2. **All projected sensors' positions** at height h
  3. For each pair of sensors v_i and v_j, we add in P all the **intersection points** $p^1_{i,j}$ and $p^2_{i,j}$ between C_i and C_j
  4. The **depot** $O = p_0 \in P$



- **Given n sensors, the number of waypoints $m = |P| \le n + n(n-1) + 1$**

Drone (2)

- The drone is constrained by the **limited energy** of its **battery** of capacity E
- The energy is consumed when the drone:
  - **Moves** between waypoints
  - **Hovers** at a position
- The sensors can start the **data transferring** procedure only when the drone **hovers at waypoints**
- The drone collects data from multiple sensors **sequentially**, one at a time
- We do not allow for a **partial transferring** $\Rightarrow$ **drone is allowed to off-load data from a subset sensor**
- We neglect **communication issues**, e.g., shadowing, fading, or multipath propagation
- The drone is constrained by the **limited storage** by its **storage capacity** S

Mission

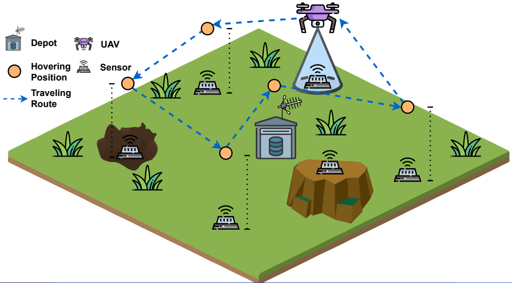
- The drone's **mission** M is formed by a sequence of distinct waypoints to be visited **to/from** the depot
- A drone's mission is characterized:
  - The **total mission cost** $\mathcal{C}_M$ in terms of energy consumed during the **flight** (**flying** and **hovering**)
  - The **total used storage** $\mathcal{U}_M$ by the drone in terms of transferred data from the selected sensors
  - The **total obtained reward** $\mathcal{R}_Q$ by the drone after having transferred the whole data from the selected sensors (Where Q is the subset of sensors selected for the mission)
- A mission is **valid** if $\mathcal{C}_M \leq E$ and $\mathcal{U}_M \leq S$
- **We assume that any mission formed by *only a single waypoint* is both energy- and storage-feasible for the drone**

Single-drone Data-collection Maximization Problem (SDMP)

- Given
  - The set of sensors V each characterized by
    - a reward r_i
    - an amount of generated data w_i
  - The drone's **energy** and **storage** budgets E and S, respectively

- The **objective** is to determine the optimal mission M^* and the optimal selection of sensors Q^* such that

$$(M^*, Q^*) = \arg\max_{M,Q} \mathcal{R}_Q \;:\; \mathcal{C}_M \leq E,\, \mathcal{U}_M \leq S.$$



NP-Hardness

Theorem

The SDMP *is* NP*-hard.*

- The proof comes from the fact that the **Orienteering Problem** is a **special case** for this problem

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj}\right) \leq E \tag{9}$$

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \tag{9}$$

1. **maximizes the overall reward**

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \quad (1)$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \quad (2)$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \quad (3)$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \quad (4)$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \quad (5)$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \quad (6)$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \quad (7)$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \quad (8)$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \quad (9)$$

2. states that each sensor can transfer its data no more than one time

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \tag{9}$$

3. forces that the drone's route begins and ends at the depot

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \tag{9}$$

4. **forbids self loops**

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \qquad (1)$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \qquad (2)$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \qquad (3)$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \qquad (4)$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \qquad (5)$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \qquad (6)$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \qquad (7)$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \qquad (8)$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj}\right) \leq E \qquad (9)$$

5. **guarantees that the generated path is a simple cycle which contains the selected sensors**

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \tag{9}$$

6. **ensures that no more than a single loop is allowed**

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj}\right) \leq E \tag{9}$$

7. **indicates the temporal order of the visited waypoints, i.e., $u_l < u_j$ if l is visited before j**

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \quad (1)$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \quad (2)$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \quad (3)$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \quad (4)$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \quad (5)$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \quad (6)$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \quad (7)$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \quad (8)$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \quad (9)$$

8. **guarantees the storage constraint**

Integer Linear Programming Formulation (ILP)

- The ILP formulation is:

$$\max \sum_{i=1}^{n} \sum_{j=0}^{m} r_i x_{ij} \tag{1}$$

subject to:

$$\sum_{j=0}^{m} x_{ij} \leq 1, \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{j=1}^{m} y_{0j} = \sum_{l=1}^{m} y_{l0} = 1, \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{3}$$

$$y_{jj} = 0, \qquad \forall j \in \mathcal{M} \tag{4}$$

$$\sum_{l=1}^{m} y_{lk} = \sum_{j=1}^{m} y_{kj} = \max_{i \in \mathcal{V}} x_{ik}, \qquad \forall k \in \mathcal{M} \setminus \{0\} \tag{5}$$

$$u_l - u_j + 1 \leq (m-1)(1 - y_{lj}), \qquad \forall l, j \in \mathcal{M} \setminus \{0\} \tag{6}$$

$$1 \leq u_l \leq m, \qquad \forall l \in \mathcal{M} \setminus \{0\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{j=0}^{m} w_i x_{ij} \leq S \tag{8}$$

$$\sum_{j=0}^{m} \left(\sum_{i=1}^{n} h_i x_{ij} + \sum_{l=0}^{m} f_{lj} y_{lj} \right) \leq E \tag{9}$$

9. **guarantees the energy constraint**

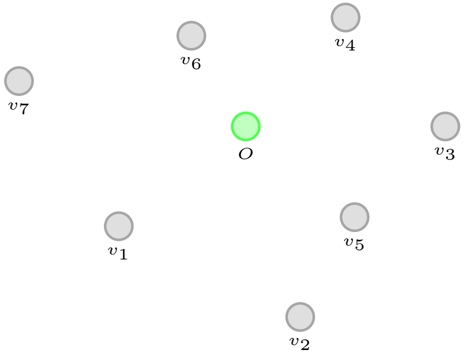
Outline

Reward-Storage-first Energy-then Optimization (RSEO) (1)

- We devise an **approximation** algorithm that sub-optimally solves SDMP, called *Reward-Storage-first Energy-then Optimization* (RSEO)

Algorithm 1: The RSEO Algorithm

$V' \leftarrow$ *knapsack*(V, S)
$P' \leftarrow$ *min-set-cover*(V', P)
$M \leftarrow$ *traveling-salesman*(P')
while $\mathcal{C}_M > E$ **do**
  $p \leftarrow \arg\min_{p_i \in M} \mathcal{R}_{p_i}, M \leftarrow M \setminus p$
return M



- **Approximation Ratio:** $\frac{0.5}{\mu \log |V'|}$

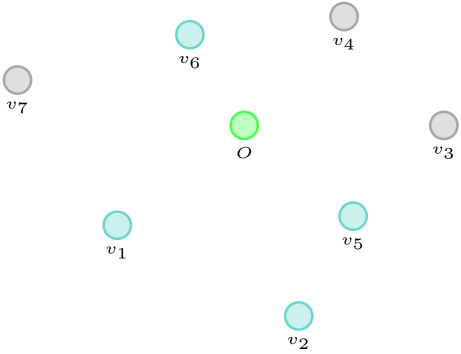
$\Rightarrow$ **Time complexity:** $\mathcal{O}(|P'|^3)$

Reward-Storage-first Energy-then Optimization (RSEO) (1)

- We devise an **approximation** algorithm that sub-optimally solves SDMP, called *Reward-Storage-first Energy-then Optimization* (RSEO)

Algorithm 1: The RSEO Algorithm

$V' \leftarrow$ *knapsack*(V, S)
$P' \leftarrow$ *min-set-cover*(V', P)
$M \leftarrow$ *traveling-salesman*(P')
while $\mathcal{C}_M > E$ **do**
  $p \leftarrow \arg\min_{p_i \in M} \mathcal{R}_{p_i}, M \leftarrow M \setminus p$
return M



- **Approximation Ratio:** $\frac{0.5}{\mu \log |V'|}$

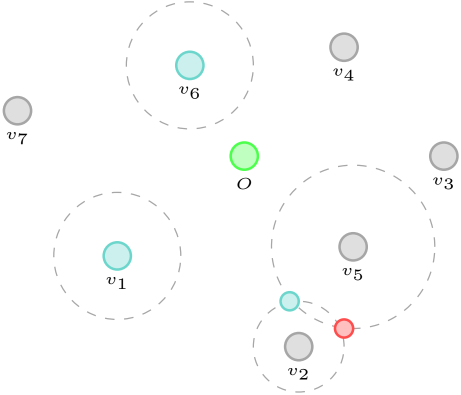
$\Rightarrow$ **Time complexity:** $\mathcal{O}(|P'|^3)$

Reward-Storage-first Energy-then Optimization (RSEO) (1)

- We devise an **approximation** algorithm that sub-optimally solves SDMP, called *Reward-Storage-first Energy-then Optimization* (RSEO)

Algorithm 1: The RSEO Algorithm

$V' \leftarrow$ *knapsack*(V, S)
$P' \leftarrow$ *min-set-cover*(V', P)
$M \leftarrow$ *traveling-salesman*(P')
while $\mathcal{C}_M > E$ **do**
  $p \leftarrow \arg\min_{p_i \in M} \mathcal{R}_{p_i}, M \leftarrow M \setminus p$
return M



- **Approximation Ratio:** $\frac{0.5}{\mu \log |V'|}$

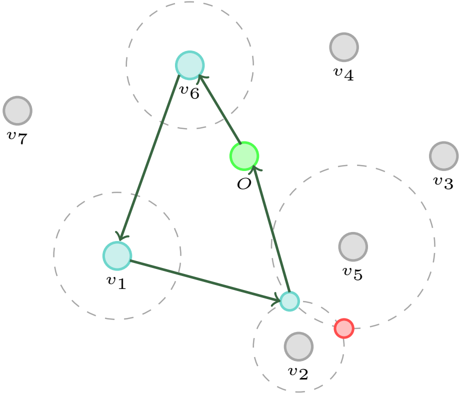
$\Rightarrow$ **Time complexity:** $\mathcal{O}(|P'|^3)$

Reward-Storage-first Energy-then Optimization (RSEO) (1)

- We devise an **approximation** algorithm that sub-optimally solves SDMP, called *Reward-Storage-first Energy-then Optimization* (RSEO)

Algorithm 1: The RSEO Algorithm

$V' \leftarrow$ *knapsack*(V, S)
$P' \leftarrow$ *min-set-cover*(V', P)
$M \leftarrow$ *traveling-salesman*(P')
while $\mathcal{C}_M > E$ **do**
  $p \leftarrow \arg\min_{p_i \in M} \mathcal{R}_{p_i}, M \leftarrow M \setminus p$
return M



- **Approximation Ratio:** $\frac{0.5}{\mu \log |V'|}$

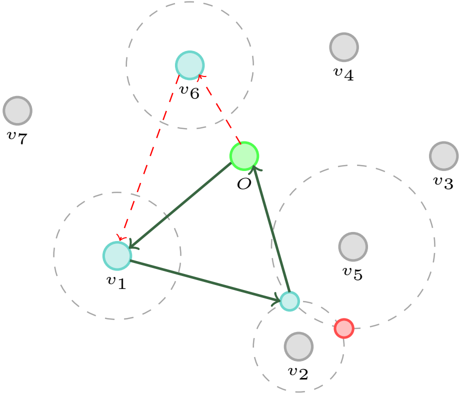
$\Rightarrow$ **Time complexity:** $\mathcal{O}(|P'|^3)$

Reward-Storage-first Energy-then Optimization (RSEO) (1)

- We devise an **approximation** algorithm that sub-optimally solves SDMP, called *Reward-Storage-first Energy-then Optimization* (RSEO)

Algorithm 1: The RSEO Algorithm

$V' \leftarrow$ *knapsack*(V, S)
$P' \leftarrow$ *min-set-cover*(V', P)
$M \leftarrow$ *traveling-salesman*(P')
while $\mathcal{C}_M > E$ **do**
  $p \leftarrow \arg\min_{p_i \in M} \mathcal{R}_{p_i}, M \leftarrow M \setminus p$
return M



- **Approximation Ratio:** $\frac{0.5}{\mu \log |V'|}$

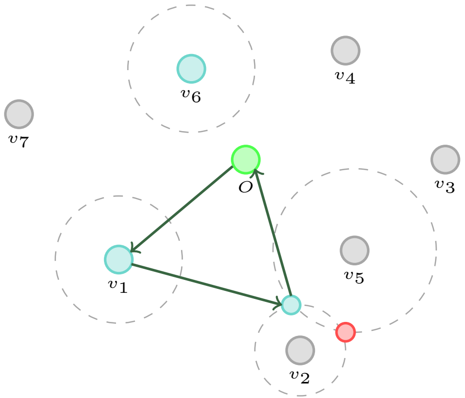
$\Rightarrow$ **Time complexity:** $\mathcal{O}(|P'|^3)$

Reward-Storage-first Energy-then Optimization (RSEO) (1)

- We devise an **approximation** algorithm that sub-optimally solves SDMP, called *Reward-Storage-first Energy-then Optimization* (RSEO)

Algorithm 1: The RSEO Algorithm

1 $V' \leftarrow$ *knapsack*(V, S)
2 $P' \leftarrow$ *min-set-cover*(V', P)
3 $M \leftarrow$ *traveling-salesman*(P')
4 **while** $\mathcal{C}_M > E$ **do**
5 $\quad p \leftarrow \arg\min_{p_i \in M} \mathcal{R}_{p_i}, M \leftarrow M \setminus p$
6 **return** M



- **Approximation Ratio:** $\frac{0.5}{\mu \log |V'|}$

$\Rightarrow$ **Time complexity:** $\mathcal{O}(|P'|^3)$

Reward-Storage-first Energy-then Optimization (RSEO) (2)

Theorem

RSEO *solves* SDMP *with an approximation ratio of* $\frac{\psi}{\mu\phi}$ *where* $\mu\phi$ *is the number of waypoints returned by a* ϕ*-approximation algorithm for the min-set-cover whose optimal solution has* μ *elements, which cover the sensors selected by a* ψ*-approximation algorithm for the knapsack.*

Time complexity: the overall time complexity of RSEO can be depicted as follows:

- We rely **fractional knapsack algorithm** which requires $\mathcal{O}(n \log n)$
- We rely on a **greedy strategy** for the *min-set-cover* which requires $\mathcal{O}(m|V'|)$
- We rely on the $\frac{3}{2}$-approximation algorithm for the TSP which takes $\mathcal{O}(|P'|^3)$
- Considering that at each iteration we **remove one vertex**, the time required by the loop is $\mathcal{O}(|P'| \log |P'|)$

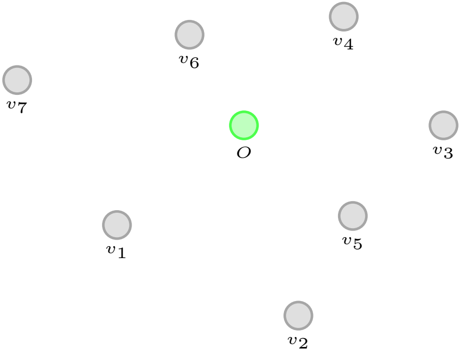
$\Rightarrow$ $\mathcal{O}(n \log n + m|V'| + |P'|^3 + |P'| \log |P'|) = \mathcal{O}(|P'|^3)$

Max ratio Reward-Energy (MRE)

- We devise an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Energy* (MRE)

Algorithm 2: The MRE Algorithm

$M \leftarrow \varnothing, \hat{P} \leftarrow \{p_0, p_1, \ldots, p_n\}$
while $\hat{P} \neq \varnothing$ **do**
  $p \leftarrow$ *best-waypoint-ratio-reward-to-energy*$(M, \hat{P})$
  if *is-augmentable*(M, p) **then**
    $M \leftarrow M \cup p$
  $\hat{P} \leftarrow \hat{P} \setminus p$
return M



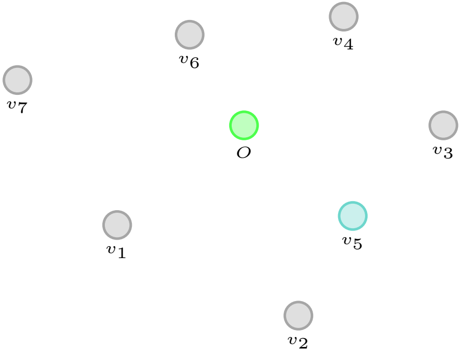
$\Rightarrow$ Time complexity: $\mathcal{O}(n^2)$

Max ratio Reward-Energy (MRE)

- We devise an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Energy* (MRE)

Algorithm 2: The MRE Algorithm

```
M ← ∅, P̂ ← {p_0, p_1, ..., p_n}
while P̂ ≠ ∅ do
    p ← best-waypoint-ratio-reward-to-energy(M, P̂)
    if is-augmentable(M, p) then
        M ← M ∪ p
    P̂ ← P̂ \ p
return M
```



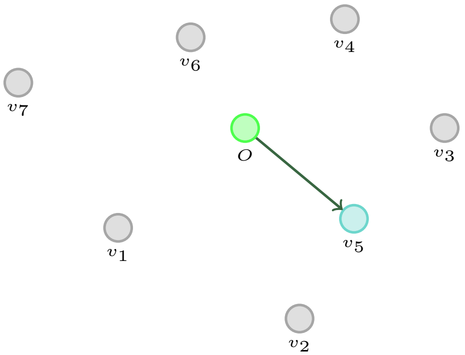
$\Rightarrow$ Time complexity: $\mathcal{O}(n^2)$

Max ratio Reward-Energy (MRE)

- We devise an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Energy* (MRE)

Algorithm 2: The MRE Algorithm

```
M ← ∅, P̂ ← {p_0, p_1, ..., p_n}
while P̂ ≠ ∅ do
    p ← best-waypoint-ratio-reward-to-energy(M, P̂)
    if is-augmentable(M, p) then
        M ← M ∪ p
    P̂ ← P̂ \ p
return M
```



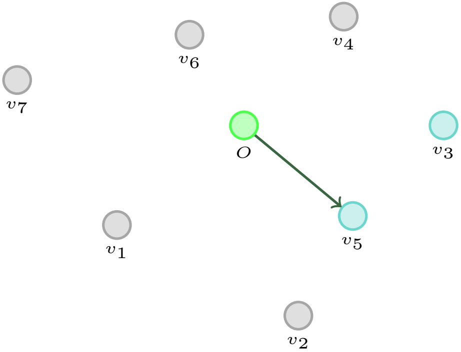
$\Rightarrow$ Time complexity: $\mathcal{O}(n^2)$

Max ratio Reward-Energy (MRE)

- We devise an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Energy* (MRE)

Algorithm 2: The MRE Algorithm

$M \leftarrow \varnothing, \hat{P} \leftarrow \{p_0, p_1, \ldots, p_n\}$
while $\hat{P} \neq \varnothing$ **do**
    $p \leftarrow$ *best-waypoint-ratio-reward-to-energy*$(M, \hat{P})$
    if *is-augmentable*(M, p) **then**
        $M \leftarrow M \cup p$
    $\hat{P} \leftarrow \hat{P} \setminus p$
return M



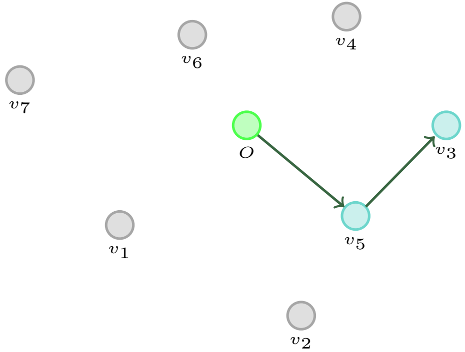
$\Rightarrow$ Time complexity: $\mathcal{O}(n^2)$

Max ratio Reward-Energy (MRE)

- We devise an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Energy* (MRE)

Algorithm 2: The MRE Algorithm

$M \leftarrow \varnothing, \hat{P} \leftarrow \{p_0, p_1, \ldots, p_n\}$
while $\hat{P} \neq \varnothing$ **do**
    $p \leftarrow$ *best-waypoint-ratio-reward-to-energy*$(M, \hat{P})$
    if *is-augmentable*(M, p) **then**
        $M \leftarrow M \cup p$
    $\hat{P} \leftarrow \hat{P} \setminus p$
return M



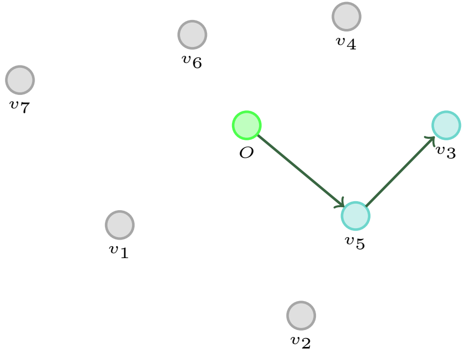
$\Rightarrow$ Time complexity: $\mathcal{O}(n^2)$

Max ratio Reward-Energy (MRE)

- We devise an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Energy* (MRE)

Algorithm 2: The MRE Algorithm

$M \leftarrow \varnothing, \hat{P} \leftarrow \{p_0, p_1, \ldots, p_n\}$
while $\hat{P} \neq \varnothing$ **do**
  $p \leftarrow$ *best-waypoint-ratio-reward-to-energy*$(M, \hat{P})$
  if *is-augmentable*(M, p) **then**
    $M \leftarrow M \cup p$
  $\hat{P} \leftarrow \hat{P} \setminus p$
return M



$\Rightarrow$ Time complexity: $\mathcal{O}(n^2)$

Max ratio Reward-Storage (MRS)

- We propose an algorithm that sub-optimally solves SDMP, called *Max ratio Reward-Storage* (MRS)
- MRS, similar to MRE, relies on the largest ratio overall **reward to storage**

Algorithm 3: The MRE Algorithm

$M \leftarrow \varnothing, \hat{P} \leftarrow \{p_0, p_1, \ldots, p_n\}$
while $\hat{P} \neq \varnothing$ **do**
  $p \leftarrow$ ***best-waypoint-ratio-reward-to-storage***$(M, \hat{P})$
  if *is-augmentable*(M, p) **then**
    $M \leftarrow M \cup p$
  $\hat{P} \leftarrow \hat{P} \setminus p$
return M

$\Rightarrow$ Since MRS works as MRE, its time complexity is $\mathcal{O}(n^2)$

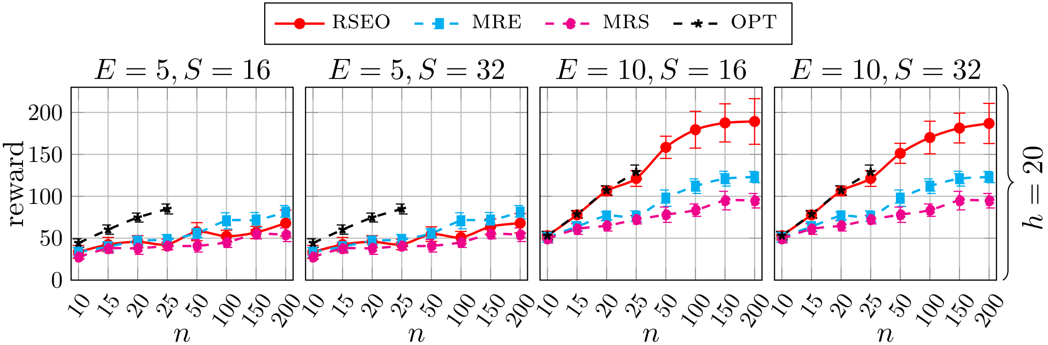
Outline

Settings

- ILP formulation is implemented using CPLEX
- Coding: Python language

Description	Variable	Unit	Value
Field sides	F	km	5
Number of sensors	n	–	$\{10, \ldots, 200\}$
Sensor height	z_i	m	$[-5, 5]$
Data to transfer	w_i	MB	$[100, 1024]$
Reward	r_i	–	$[1, 10]$
Drone altitude	h	m	$\{20, 40\}$
Communication range	R	m	50
Energy consumption for flying	–	J/m	200
Energy consumption for hovering	–	J/s	700
Data transfer rate	–	MB/s	9
Energy budget	E	MJ	$\{5, 10\}$
Storage budget	S	GB	$\{16, 32\}$

A Few Results



- The **best** performing algorithm is RSEO
- RSEO performs worse when it has **low energy** budget with respect to the **size of the field**
- MRE's results are quite **stable**
- The **worst** performing algorithm is MRS
- Storage constraint seems less **stringent**

Outline

Conclusion and Future Work

- In this work:
  - We formalized an optimization problem, the ***Single-drone Data-collection Maximization Problem*** (SDMP)
  - We showed SDMP is NP-Hard
  - We proposed an ILP formulation
  - We devised an approximation and two heuristic algorithms
  - We evaluated the performance of our algorithms on randomly generated data

- Future work:
  - To investigate the SDMP considering real communication issues
  - To explore a multiple drone scenario, where a fleet of drones have to cooperate in order to collect data
    - $\rightarrow$ **under review**: "*Wireless IoT Sensors Data Collection Reward Maximization by Leveraging Multiple Energy- and Storage-Constrained UAVs*", **Journal of Computer and System Sciences**
  - To build a real test-bed with a single drone which aims to evaluate a real case performance

Thank you for your attention!