Computer Network Performance: Project

Giulio Attenni

April 24, 2020

1 Introduction

Purpose of this document is to present the work done for the *Computer Network Performance* course 2019/2020. The goal of the project is to design and implement a model for centralized task assignment problem for swarms of aerial drones.

2 Models

2.1 Model A

The first model is one of the optimization problems suggested in [1], namely *Maximum Coverage under Battery Constraints*.

$$\boxed{\begin{array}{l}
max \sum_{i\in\Psi}\sum_{u\in U, j\in\Psi\cup\{d_u\}} x^u_{ij} \\
s.t. \\
\sum_{i\in\Psi\cup\{d_u\}} x^u_{ij} = \sum_{k\in\Psi\cup\{d_u\}} x^u_{jk}, \forall u\in U, \forall j\in\Psi\cup\{d_u\} \\
\sum_{j\in\Psi} x^u_{d_u j} = 1, \forall u\in U \\
z^u_j - z^u_i \geq x^u_{ij} + |\Psi|\cdot(x^u_{ij}-1), \forall u\in U, \forall i,j\in\Psi \\
z_i \in \{1,...,|\Psi|\}, \forall i\in\Psi \\
\sum_{i,j\in\Psi\cup\{d_u\}} \omega_{ij}\cdot x^i_{ij} \leq b_u, \forall u\in U \\
x^u_{ik} \in \{0,1\}
\end{array}}$$

2.2 Model N

This model is based on the previous one, which has the issue that is not always able to reach the maximum coverage according to the battery limitations. This issue arises from the fact that if a node is reached multiple times counts as multiple targets are reached.

2.2.1 Decision variables

- $\forall i \in \Psi$

$$\delta_i = \begin{cases} 1 & \text{if target } i \text{ is reached} \\ 0 & \text{otherwise} \end{cases}$$

- $\forall i \in \Psi$, $\forall u \in U$, $\forall j \in \Psi \cup \{d_u\}$

$$x^u_{i,j} = \begin{cases} 1 & \text{if } u \text{ goes from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

2.2.2 Objective function

We want to maximize the following objective function:

$$f : \sum_{i \in \Psi} \delta_i \tag{1}$$

since we want to maximize the actual target coverage, i.e. the number of reached targets.

2.2.3 Constraints

- We still need all the constraints defined in model A.
- In order to make the δ_i's equal to 1 only if the target is covered:
  $\forall i \in \Psi, \forall u \in U, \forall j \in \Psi \cup \{d_u\}$

  $$\delta_i \leq \sum_{u \in U} \sum_{j \in \Psi \cup \{d_u\}} x^u_{ij}$$

  Each δ_i is 1 if and only if the target i is covered because:

  - if the target i is not covered then for each drone u, and each points j

    $$x^u_{ij} = 0 \implies \sum_{u \in U} \sum_{j \in \Psi \cup \{d_u\}} x^u_{ij} = 0$$

    thus $\delta_i = 0$
  - if the target i is covered then for some drone u, and some point j

    $$x^u_{ij} = 1 \implies \sum_{u \in U} \sum_{j \in \Psi \cup \{d_u\}} x^u_{ij} \geq 1$$

    thus $\delta_i = 1$, because it is a maximization problem.

2.2.4 The model

$$Maximize : \sum_{i\in\Psi} \delta_i$$

$$s.t.$$

$$\sum_{i\in\Psi\cup\{d_u\}} x^u_{ij} = \sum_{k\in\Psi\cup\{d_u\}} x^u_{jk} \qquad \forall u \in U, \forall j \in \Psi \cup \{d_u\}$$

$$\sum_{j\in\Psi} x^u_{d_u j} = 1 \qquad \forall u \in U$$

$$z^u_j - z^u_i \geq x^u_{ij} + |\Psi| \cdot (x^u_{ij} - 1) \qquad \forall u \in U, \forall i, j \in \Psi$$

$$z_i \in \{1, ..., |\Psi|\} \qquad \forall i \in \Psi$$

$$\sum_{i,j\in\Psi\cup\{d_u\}} \omega_{ij} \cdot x^i_{ij} \leq b_u \qquad \forall u \in U$$

$$x^u_{ij} \in \{0, 1\} \qquad \forall i \in \Psi, \forall u \in U, \forall j \in \Psi \cup \{d_u\}$$

$$\delta_i \leq \sum_{u\in U} \sum_{j\in\Psi\cup\{d_u\}} x^u_{ij} \qquad \forall i \in \Psi$$

$$\delta_i \in \{0, 1\} \qquad \forall i \in \Psi$$

3 Implementation

Each model is defined as a MILP problem, so in order to find the optimal task for each drone we solve such problem through IBM Cplex Optimizer. Using Cplex Python API we dynamically define the optimization problem given the input dataset.

3.1 Datasets

Each dataset contains 3 input files: `drones.in`, `targets.in` and `travelling_costs.in`.

1. Each row of `drones.in`, except the first, contains the following information about a drone of the swarm: identifier, depot point identifier, battery capacity.

2. Each row of `targets.in`, except the first, contains the following information about a target: identifier, inspection time.

3. Each row of `travelling_costs.in`, except the first, contains the following information: departure point identifier, arrival point identifier, cost of travelling from departure point to arrival point.

```
data
└─ a
```

```
│
│  ├─ drones.in
│  ├─ targets.in
│  └─ travelling_costs.in
├─ b
│  ├─ drones.in
│  ├─ targets.in
│  └─ travelling_costs.in
└─ c
   ├─ drones.in
   ├─ targets.in
   └─ travelling_costs.in
```

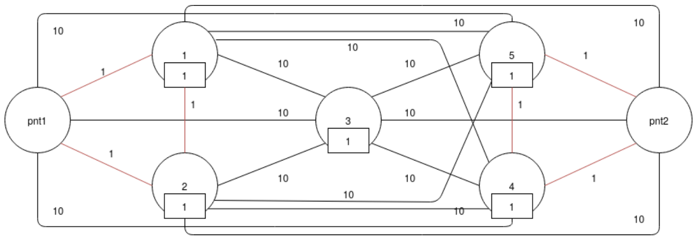


Figure 1: Dataset `a` seen as a graph.

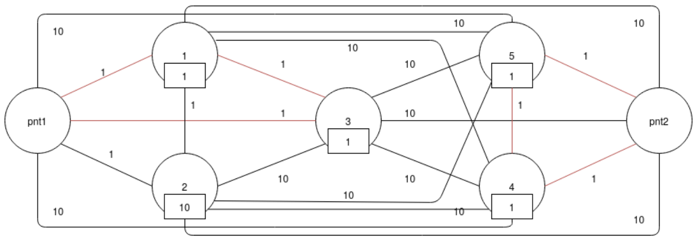


Figure 2: Dataset `b` seen as a graph.

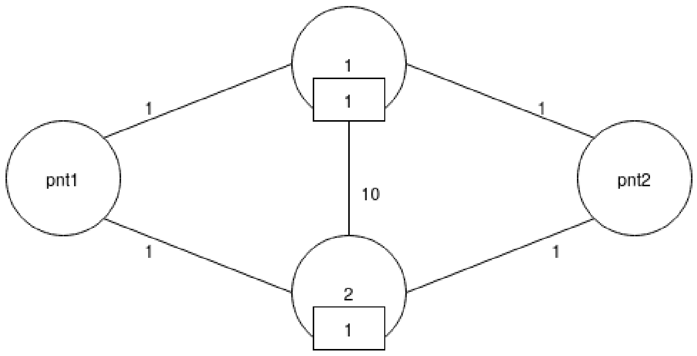
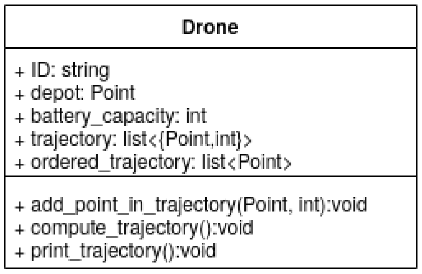
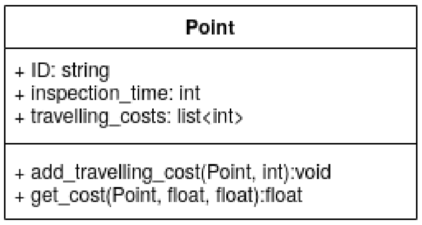


Figure 3: Dataset `c` seen as a graph.

3.2 UML



```
Drone
+ ID: string
+ depot: Point
+ battery_capacity: int
+ trajectory: list<{Point,int}>
+ ordered_trajectory: list<Point>
+ add_point_in_trajectory(Point, int):void
+ compute_trajectory():void
+ print_trajectory():void
```

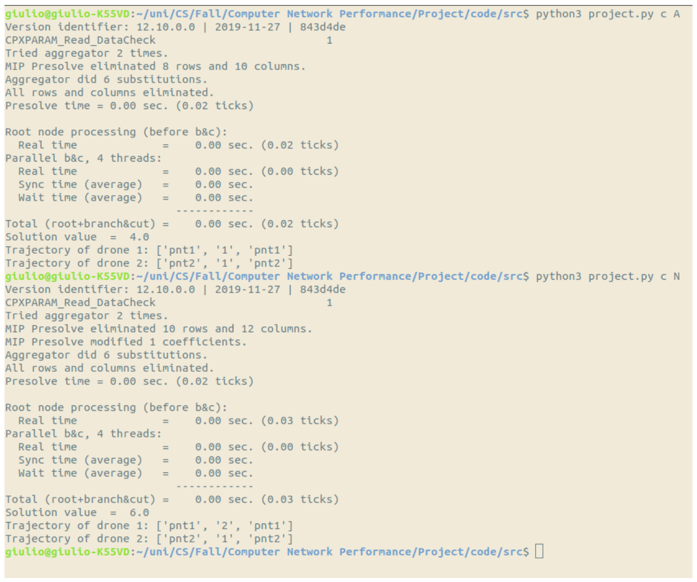


```
Point
+ ID: string
+ inspection_time: int
+ travelling_costs: list<int>
+ add_travelling_cost(Point, int):void
+ get_cost(Point, float, float):float
```

4 Execution

Execute project.py in `AttenniProject/src`. It takes two arguments

- the name of the input dataset folder (`a` or `b` or `c`)
- the name of the model (`A` or `N`)



```
giulio@giulio-K55VD:~/uni/CS/Fall/Computer Network Performance/Project/code/src$ python3 project.py c A
Version identifier: 12.10.0.0 | 2019-11-27 | 843d4de
CPXPARAM_Read_DataCheck                          1
Tried aggregator 2 times.
MIP Presolve eliminated 8 rows and 10 columns.
Aggregator did 6 substitutions.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.02 ticks)
Parallel b&c, 4 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.02 ticks)
Solution value  =  4.0
Trajectory of drone 1: ['pnt1', '1', 'pnt1']
Trajectory of drone 2: ['pnt2', '1', 'pnt2']
giulio@giulio-K55VD:~/uni/CS/Fall/Computer Network Performance/Project/code/src$ python3 project.py c N
Version identifier: 12.10.0.0 | 2019-11-27 | 843d4de
CPXPARAM_Read_DataCheck                          1
Tried aggregator 2 times.
MIP Presolve eliminated 10 rows and 12 columns.
MIP Presolve modified 1 coefficients.
Aggregator did 6 substitutions.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.02 ticks)

Root node processing (before b&c):
  Real time             =    0.00 sec. (0.03 ticks)
Parallel b&c, 4 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.00 sec. (0.03 ticks)
Solution value  =  6.0
Trajectory of drone 1: ['pnt1', '2', 'pnt1']
Trajectory of drone 2: ['pnt2', '1', 'pnt2']
giulio@giulio-K55VD:~/uni/CS/Fall/Computer Network Performance/Project/code/src$ 
```

Figure 4: Execution of both models on dataset `c`.

References

[1] Computer Network Performance Projects. Project topic: task assignment and path planning for swarms of aerial drones, Novella Bartolini `https://twiki.di.uniroma1.it/pub/PDSDR/AA1819Bartolini/CNP_projects.pdf`