Reinforcement learning vs rule-based dynamic movement strategies in UAV assisted networks

Adel Mounir Said, Michel Marot, Chérifa Boucetta, Hossam Afifi, Hassine Moungla, Gatien Roujanski

Reinforcement learning vs rule-based dynamic movement strategies in UAV assisted networks

Adel Mounir *, Michel Marot† Chérifa Boucetta§, Hossam Afifi †, Hassine Moungla †‡, and Gatien Roujanski †
*National Telecommunication Institute - NTI, Cairo, Egypt
†Telecom SudParis, Institut Polytechnique de Paris, SAMOVAR, Palaiseau 91120, France
§LIGM CNRS-UMR 8049, Université Gustave Eiffel, Champs-sur-Marne, France
‡Université de Paris, LIPADE, F-75006 Paris, France

***Abstract*—Since resource allocation of cellular networks is not dynamic, some cells may experience unplanned high traffic demands due to unexpected events. Unmanned aerial vehicles (UAV) can be used to provide the additional bandwidth required for data offloading.**

Considering real-time and non-real-time traffic classes, our work is dedicated to optimize the placement of UAVs in cellular networks by two approaches. A first rule-based, low complexity method, that can be embedded in the UAV, while the other approach uses Reinforcement Learning (RL). It is based on Markov Decision Processes (MDP) for providing optimal results. The energy of the UAV battery and charging time constraints have been taken into account to cover a typical cellular environment consisting of many cells.

We used an open dataset for the Milan cellular network provided by Telecom Italia to evaluate the performance of both proposed models. Considering this dataset, the MDP model outperforms the rule-based algorithm. Nevertheless, the rule-based one requires less processing complexity and can be used immediately without any prior data. This work makes a notable contribution to developing practical and optimal solutions for UAV deployment in modern cellular networks.

***Index Terms*—Anomaly detection, Network outliers, Machine learning, Reinforcement Learning techniques, Q-learning.**

I. INTRODUCTION

The future comes with a vision relying on smart applications because everything is connected to the Internet. Some situations require increasing the cellular resources of certain base stations (BSs) in an optimized manner to satisfy sporadic mobile users traffic requests and provide sustained quality of service (QoS). This is to support the massive amount of data demanded by mobile users of voice calls, multimedia download, chats, etc. It is clear that not all cells have a stable behavior of expected total number of mobile users all the time of the day. Some BS stations, such as BSs in the downtown center or stadiums, are experiencing a sharp increase in mobile user connections over bandwidth (BW) capacity. This can happen when there is a festival or national event, or a match in a stadium. This occurs at certain times on the weekend or due to any other events and it is of known duration. Overcapacity cases of BS are called anomalies as this happens periodically at certain times, so it's not the default behavior all the time. Therefore, it makes no sense to add permanent resources to this BS while they are not needed most of the time, otherwise, it is considered wasted. Thus, an innovative solution is needed to overcome the scarcity of resources in these anomalies situations and prevent service downtime by optimizing the distribution of additional BW to the affected BSs.

Unmanned Aerial Vehicles (UAV) is one of the promising solutions which can cope the limitations of the infrastructure [1] [2]. According to [3], the global market of UAV is increasing rapidly and is expected to reach $21.8 billion by 2027, with compound annual growth rate of 14.1%. Moreover, given the growing importance of UAV, ITU-T issued its functional architecture standard in [4]. It is considered as enablers of various applications including telecommunications [5], which we rely on in this article to increase cellular resources to face anomaly situations. Moreover, UAV can be used in the field of IoT. It can be integrated in an IoT environment not only to support communication between IoT devices but also to enhance its services/applications [6].

The use case for this article is to cover the anomalies of resources requests in a cellular network. Accordingly, there is a need to increase the resources of these cells to meet the sharp increase of the mobile users activities. UAV is proposed to be used to increase data offloading of cells having anomalies.

In this work, two approaches have been proposed to control the deployment of the UAVs. First, a rule-based model is proposed that is determined by describing the parameters and constraints used, and the model's work is adapted to these constraints and parameter values that change over time. Second, the optimization problem will be formulated mathematically into a Markov Decision Process (MDP). MDP is a memory-less stochastic process, and will be used to describe the sequence of random transitions of the environment. Hence, Reinforcement Learning (RL) is proposed to be used to maximize rewards that come from environment's states or transitions. RL is trained as a prediction system by a cellular data set to make the right decision in states related to resource demands in each cell of the cellular network. Note that, strictly speaking, an anomaly is unexpected. Thus, it cannot be learned, and a reinforcement method cannot work to tackle anomaly management. But, rather, we aim at managing demand peaks which may (or not) occur. What we know is that an anomaly has more chance to happen in loaded, periods, which can be learned by any learning algorithm like a reinforcement method. That is why an MDP-based reinforcement method is suitable to manage such anomalies and, actually it gives good results.

UAV battery levels are taken into account, by calculating the energy consumption of send and back actions to/from the cell, as well as the offloading energy. Additionally, the UAV charging time is considered to demonstrate its affect on the models performance. Real Call Detail Records (CDRs) dataset of Milan city, Italy has been used to train the learning model and for simulation.

This work is an extension of our recent previous published paper [7] with additional contributions as follows:

- Study the proposed solutions in a multi-cellular environment.
- Consider a constraint of having number of drones less than the number of cells.
- Consider different traffic type based on time.
- In addition to exploiting idle drones in neighboring cells.

The rest of the paper is organized as follows: related work is surveyed in section II. The system model architecture and Rule-based solution approach and the proposed optimization algorithms are presented in the section III-A. Section V describes the RL solution approach and the proposed optimization algorithm. Thereafter, the performance evaluation of the two proposed models are discussed in section VI. Finally, the paper work conclusion is presented in section VII.

II. Related work

This section reviews previous work related to to UAV in the recent years with respect to its applications and modeling solutions.

Authors in [8] were concerned in proposing a technique to find the best position of the UAV assisting vehicular communications. UAV will act as a relay between vehicles, which constitutes a dynamic environment. This technique is based on using particle swarm optimization and a genetic algorithms. The altitude should be identified to alleviate communication blockage due to surrounding terrain, and this is not practical all the time, else it mandates site survey. Moreover, it'll be better if UAV will have a fixed position to maximize it's battery lifetime.

Another contribution of UAVs in communications [9] where the authors proposed the use of UAV for maritime communications for the future of sixth generation (6G). The work was focusing on channel modeling the maritime communication between UAVs and ships. The proposed model included the sea surface wave equation, also the evaporation duct, and elevated duct over the sea surface. The work is concentrated on the communication channel model, but the authors did not state that this model is applicable to how far from the port where we can find BSs, drones, and ships. Also, battery levels were not taken into account.

The work of [10] proposed a service model of UAVs to be used commercially to deliver goods based on the uncertainty effect of weather, UAV capabilities, spatiotemporal availability, and cost. It is an optimizing solution based on the mentioned parameters to select the most suitable drone to have certain task. This work didn't mention the criteria of UAV controlling and navigation.

Authors of [11] proposed a neural network optimization model to classify scenes out of videos captured by UAVs. They used swarm optimization model. Deep learning is used in this solution to support only the application of the UAV, without the control of the UAV.

Another work of [12] proposed to use also neural networks in training UAV to fly autonomously and detect the terrain changes. As authors mentioned, this work is good for save searches of natural resources in rugged areas and for the safety of underground tourist activities.

The work of [13] proposed to build two layers of UAVs for maritime communications in order to provide powerful computing through the limited power of the UAVs. Hence, minimize the delay in communications and computation. The top layer have centralized UAVs works as mobile edge computing (MEC), while the bottom layer consists of a group of UAVs. The proposed model is based on MDP and deep reinforcement learning to optimize the trajectories of the UAVs. This work architecture has no connection to the cellular network, so it looks like a closed system. Also, there many UAVs in use, cost is not mentioned as an effective parameter.

In [14] proposed to use deep neural network and multi head attention to predict heart disease. The main features needed for the prediction model are extracted from the patient electronic health record. The quality of the proposed model is measured against SVM, CNN, and ANN.

We have previous studies in the same context. In our previous work of [15] a dynamic network anomaly detection model is proposed using a long-term memory (LSTM) machine learning algorithm to detect cells overloaded unexpected events. Then, an optimization model based on maximizing the service time of the UAV and the number of users covered was proposed. In [16], we studied the impact of e-Health applications on cellular networks with determining the cells of peak requests.

Moreover, our previous work [17] proposed the use of deep learning algorithm to predict the normal cellular network behaviour and its capacity. Hence, three different algorithms are used for three different criteria for sending the required UAV, traffic priority, buffer and delay limitations.

The following table I summarizes the related work into focus and methodologies used.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum.

Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

In addition to that, this paper contributions compared to our previous work are different and superior in many comparison faces. In the previous work[7], two optimization model based on integer linear programming (ILP) and long-term memory (LSTM) are proposed. ILP has been proposed to provide optimal results but with a penalty of computational complexity in terms of processing time that increases exponentially as the problem size increases, limited relaxation techniques for solving integer problems, and fine-tuning-based ILP models with small changes that make them less robust in dynamic environments such as the cellular network. In addition, it is a difficult continuous improvement technique. Therefore, it is proposed that LSTM learns the optimal results of ILP and provides a much faster learning technique than ILP but with suboptimal accuracy. Accordingly, in this current work, we propose a rule-based solution, which has many advantages in terms of fastness due to its low processing and simplicity. Also, it has a steady and interpretative response with a high accuracy. On the other hand, rule-based has no learning capability, which makes continuity of improvement limited, especially with continuous changes in the network. On the other hand, Reinforcement Learning (RL) gives optimal results and Markov Decision Process provides a mathematical framework for modeling decision making in situations where outcomes are uncertain and exhibit a certain level of stochasticity. Thus, the MDP-based reinforcement learning method is suitable for managing such anomalies, and it gives good results. In addition, the previous work proposed there is one drone to deploy in case of anomaly and without regarding the neighbour cells (dynamic allocation).

III. System architecture and problem statement

A. System Architecture

Before moving on to the system architecture, we will use the term drone along the rest of the paper. Figure 1 presents the proposed RL model architecture. It consists of cellular network infrastructure of multi-cells of each has its BS. There is a coordinator (RL agent) responsible of managing drones and each BS contacts when an anomaly occurs. The RL model have number of inputs that constitute the RL state.

These inputs are each BS resources in terms of BW, the number of drones in the cellular network, and the energy

TABLE I: Related Work Summary

Ref	Year	Focus	Proposed Model (Methodology)	Key Points
[15]	2019	UAV deployment optimization for cellular coverage	Long short-term memory (LSTM) for anomaly detection, optimization for service time and user coverage	Detection of network anomalies and optimization of UAV deployment for coverage during events
[16]	2019	Impact of e-Health applications on cellular networks	Deep learning for predicting network behavior and capacity	Prediction of normal network behavior using deep learning, optimization for UAV deployment based on traffic priority and constraints
[8]	2020	UAV positioning for vehicular communications	Particle swarm optimization and genetic algorithms	Optimal UAV position considering dynamic environment, altitude, and battery life
[9]	2021	UAVs for maritime communications	Channel modeling incorporating sea surface wave equation, evaporation duct, and elevated duct	Concentrated on channel modeling for maritime communications, didn't address practical applicability or battery concerns
[10]	2021	Commercial drone service optimization	Optimization based on weather, UAV capabilities, availability, and cost	Selection of suitable drone for tasks considering various uncertainties, lacking criteria for UAV control and navigation
[11]	2021	Scene classification using UAV-captured videos	Multi-objective particle swarm optimization for neural network optimization	Neural network model for scene classification without UAV control implementation
[12]	2022	Autonomous UAV flight and terrain change detection	Neural networks for autonomous flight and terrain change detection	Suitable for terrain exploration and safety applications, no mention of control mechanisms
[13]	2022	UAVs for maritime edge computing	Deep reinforcement learning for trajectory optimization	Two-layer UAV architecture for computing and communication optimization, closed system with no cellular network integration
[14]	2023	Predicting cardiovascular disease in diabetic patients	Deep neural network with multi head attention	Extracting features from electronic health records for prediction, compared against SVM, CNN, and ANN models

characteristics of each drone. The drone energy characteristics include its cellular resources capacity in terms of BW, battery level, energy consumption to send/back the drone to/from the cell, and energy consumption in the offloading. These energy characteristics are necessary for the drone's management that will be proposed in algorithm 2. In addition, the activities recorded in CDR and of existing mobile users are important to know the additional required BW and the type of traffic of these activities in terms of real time / non-real time. Accordingly, the expected output of the RL model is the drone's actions the coordinator takes to deploy and eventually offloading. Action taken is based on the current state of the network as will be demonstrated in the algorithm 1.

There are four potential drone's actions; sending a drone to the cell where there is an anomaly, offloading to compensate for the required BW, returning the drone to the charging station when its charge is depleted, and idle in the cell when it is no longer needed in its existing cell. The multi-cellular environment, traffic priorities and drone's energy are taken into account.

In this work, the optimization solution is based on prioritize cells with larger number of mobile users. Moreover, there is a constraint to have number of drones less than the number of cells in the cellular network. Also, the time under study is flexible to be an hour or a day without losing the generality of the algorithm.

B. Problem Statement

We want to optimize drones deployment to satisfy extra bandwidth demand, or, in other words, minimize as possible the amount of data demand not served because of lack of capacity in cells. There are different possible objective performance criteria. Denoting $D_{i,R}(t)$ the *remaining* of BW/Channels requested of cell(i) at time(t) and $C_i(t)$ the bandwidth capacity per base station of cell(i) at time(t), possible objective functions could be:

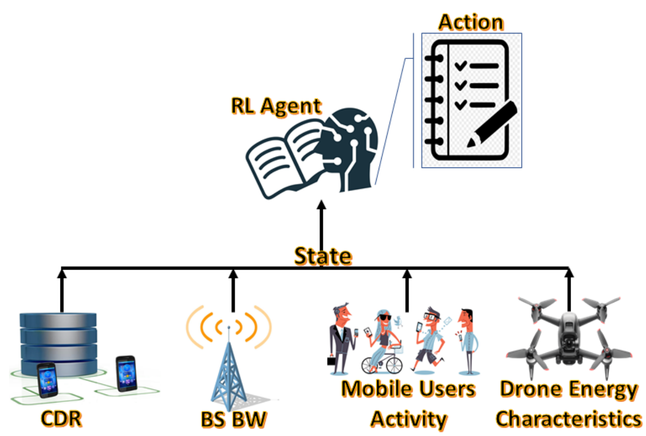

Fig. 1: RL architecture

$$\min \sum_{i\in\Omega_{\text{cells}}} D_{i,R}(t); \tag{1}$$

$$\min \sum_{i\in\Omega_{\text{cells}}} \frac{D_{i,R}(t)}{D_i(t)}; \tag{2}$$

$$\min \sum_{i\in\Omega_{\text{cells}}} \frac{D_{i,R}(t)}{C_i(t)}; \tag{3}$$

$$\min \sum_{i\in\Omega_{\text{cells}}} \mathbb{1}_{\{\frac{D_{i,R}(t)}{C_i(t)}>\xi\}}. \tag{4}$$

Equation (1) aims to minimize the total extra bandwidth needed in the cellular network. The problem is that it may make no sense to consider absolute value if the network is heterogeneous. That is why it would be better to target relative extra demand compared to either the total cell's demand (equation (2)) or to the total cell's capacity (equation (3)). Nevertheless, this may lead to a dispersion of the drone resource: in some situations, all the cells may be badly and equally served because of the sum which mixes the individual performance of each cell. Another approach consists in minimizing the number of cells which are not served with a sufficient QoS. For instance, the number of cells for which the extra demand relatively to the cell capacity is above a certain threshold. This is the objective function in equation (4).

Of course, optimizing the resource aims of using it in the best way but it raises fairness issue. Do we want to have an excellent service for certain cells and a (possibly very) bad for others? or do we prefer to accept good though not excellent service for some cells and a rather good one for the others?. These questions lead to an optimized criteria such as:

$$\min \sum_{i \in \Omega_{\text{cells}}} \left| D_{i,R}(t) - \frac{1}{|\Omega_{\text{cells}}|} \sum_{k \in \Omega_{\text{cells}}} D_{k,R}(t) \right|; \tag{5}$$

$$\min \sum_{i \in \Omega_{\text{cells}}} \left| \frac{D_{i,R}(t)}{D_i(t)} - \frac{1}{|\Omega_{\text{cells}}|} \sum_{k \in \Omega_{\text{cells}}} \frac{D_{k,R}(t)}{D_k(t)} \right|; \tag{6}$$

$$\min \sum_{i \in \Omega_{\text{cells}}} \left| \frac{D_{i,R}(t)}{C_i(t)} - \frac{1}{|\Omega_{\text{cells}}|} \sum_{k \in \Omega_{\text{cells}}} \frac{D_{k,R}(t)}{C_k(t)} \right|. \tag{7}$$

All of these approaches make sense and are rather a matter of choice according to the operator's objectives. Here, we focus on minimizing the average of extra-demand, above a certain bandwidth threshold, which is not satisfied given that their is an extra-demand, i.e. We are targeting the objective function

$$\min \frac{\sum_{i \in \Omega_{\text{cells}}} D_{i,R}(t)}{\sum_{i \in \Omega_{\text{cells}}} \mathbb{1}_{\{C_i(t)\} > \xi\}}}, \tag{8}$$

hence the rewards chosen in the part V. Nevertheless, to reach a certain fairness, in each round of the algorithm, the cells are sorted descending by decreasing the demand and the highest loaded ones are served first. Thus, the algorithm has a primary objective which is the demand satisfaction and a secondary objective of relative fairness: the cells that cannot be served due to lack of resources are those for which the relative demand is the smallest.

IV. Rule-Based Optimization

In this part, we explain the rule-based optimization algorithms proposed in this work according to the mentioned use case in the system architecture model. There are two algorithms of each optimize different criteria. The variables used in these algorithms are described in Table II. The rule-based optimization algorithm depends on specific parameters for each cell which are the available bandwidth of the cell and the total requested bandwidth. In addition, the drone offloading capacity to determine number of needed drones, the number of idle drones in the neighbor cells and its battery energy level, the number of available drones in the charging station, and the number of serving drone and its battery energy level. All these parameters are time dependent.

TABLE II: Table of notation

Parameter	Description
h	The offloading capacity of drone
a	Action taken by the RL agent
$E(t)$	Number of idle drones in neighbor cells at time(t)
$S_i(t)$	Number of serving drones in cell(i) at time(t)
$L(t)$	Number of available drones in charging station at time(t)
$D_i(t)$	Total BW requested by users of cell(i) at time(t)
$C_i(t)$	BW per base station of cell(i) at time(t)
$D_{i,R}(t)$	Needed extra BW in cell(i) at time(t)
$N_i(t)$	Number of needed drones in cell(i) at time(t)

The first algorithm optimizes and manages the cell resources. As shown in algorithm 1 steps, the list of input parameters is; day time, number of cells, total number of drones, number of drones in the charging station, the BW of each cell, the total requested BW and number of idle/serving drones in each cell. Also, there are no idle/running drones in cells with optimization model initialization. The output is the action taken in each cell at the time under study.

For better understanding, we will describe algorithm 1 in bolt steps as follows:

- The first part of the algorithm, at every time(t), step 4 collects the BW traffic request for each cell(i).
- Accordingly, in step 5, we calculate the required extra BW, which indicates the detection of anomalies in the cell. This calculation is based on the BW capacity $C_i(t)$ and the BW capacity (h) of each of the current existing drones $S_i(t)$ in the cell.
- Next, in step 7, we sort the cells in descending order based on the BW required to prioritize providing extra BW for cells with a large number of users activities. The BW list contains the sorted cells and their BW needs.
- The second part of the algorithm works on the list of sorted cells.
- In step 9, the BW required for each cell is checked to ensure that it is greater than certain threshold (ρ) and there is a need for resources in that cell(i).
- Consequently, we calculate in step 10 the number of required drones resulting from dividing the BW needed by the drone's BW capacity.
- Step 11 is a condition of two options has been added to check whether the traffic is realtime or not.
  - Consider a realtime traffic:
    - The first step before sending a drone to work in the cells in need is to check the battery level of the idle drones in the list of neighbor cells. Accordingly, drones with insufficient power to operate will return to the charging station for recharging. This is based on algorithm 2 that will be described in the next part.

Algorithm 1: Cellular Resources Optimization (Rule-based)

input : $C_i(t)$ is the available BW of Cell(i) at time(t)
input : $D_i(t)$ is the total requested BW of Cell(i) at time(t)
input : $E(t)$ is the number of idle drones in neighbor cells
input : $S_i(t)$ is the number of serving drones in cell(i) at time(t)
input : h is the drone's BW capacity
input : $L(t)$ is the number of available charged drones in garage
output: The best action to be taken for each Cell(i) at time(t)

1 **for** *each time*(t) **do**
2   **for** *each cell*(i) **do**
3   Call Algorithm 3 to check each current SERVING Drone energy
4   Collect BW request of each cell(i)
5     $D_{i,R}(t) = D_i(t) - C_i(t) - S_i(t) * h$ // Calculate BW needed in cell(i) at time(t)
6   **end**
7   BW List = Sort Cells descending according to BW needed
8   **for** *each cell in BW List*(j) **do**
9   **if** $BW_j(t) > \rho$ **then** // Cell Needs extra BW
10       $N_j(t) = \lceil BW_j(t)/h \rceil$
11   **if** *There is real-time traffic* **then**
12   Call Algorithm 2 to check each IDLE Drone energy
13   **if** $E(t) \geq N_j(t)$ **then** // Enough Drones available from neighbors
14           $a = 1$ // Send all needed drones
15           $E(t) = E(t) - N_j(t)$ // Update available Drones in the neighbor cells
16           $S_j(t) = S_j(t) + N_j(t)$ // Update current SERVING Drones in the cell
17   Call algorithm 3 to CHECK each SERVING Drone energy
18           $a = 1$; // offloading
19   **else if** $E(t) < N_j(t)$ *AND* $L(t) \geq N_j(t)$ **then** // "Enough Drones available from neighbors/charging station"
20           $a = 1$ // Send all needed drones
21           $E(t) = 0$ // Update available Drones in the neighbor cells
22           $L(t) = L(t) - (N_j(t) - E(t))$ // Update available Drones in the charging station
23           $S_j(t) = S_j(t) + N_j(t)$ // Update current SERVING Drones in the cell
24   Call algorithm 3 to CHECK each SERVING Drone energy
25           $a = 1$; // offloading
26   **end**
27   **else if** $E(t) < N_j(t)$ *AND* $L(t) < N_j(t)$ **then** // "No enough Drones available
28           $a = 1$; // Send available drones
29           $E(t) = 0$; // Update available Drones in the neighbor cells
30           $L(t) = 0$; // Update available Drones in the charging station
31           $S_j(t) = S_j(t) + E(t) + L1(t)$ // Update current SERVING Drones in the cell
32   Call algorithm 3 to CHECK each SERVING Drone energy
33           $a = 1$; // offloading
34   **end**
35   **end**
36   **end**
37   **else**
38   Buffering
39   **end**
40   **end**
41   **else if** $BW_j(t) < 0$ **then** // There is unneeded BW in the cell
42       $N_j(t) = \lceil (BW_(j)(t))/h \rceil$ // Calculate number of unneeded Drones
43       $E(t) = E(t) + N_j(t)$; // Update IDLE Drones in neighbor cells
44       $S_j(t) = S_j(t) - N_j(t)$ // Update current SERVING Drones in the cell
45   **end**
46   **end**
47 **end**

- $*$ Next, there are three conditions of different possible options, if there are charged drones available in the neighbor cells or in the neighbor cells and the charging station or if no charged drones are currently available.
- $*$ Consider the first option of having charged drones available in neighboring cells as in step 13.
  - · Hence, in step 14, the needed drones are sent from neighbor cells.
  - · After that, in step 15, the number of available drones in neighbor cells is updated.
  - · Next, in step 16, number of drones that are currently serving in that cell is updated.
  - · There is an important factor is to check the energy level of each current serving drone before applying the offloading action as in step 17. This is to ensure that each drone has enough energy to perform the offloading action and return to the charging station, or else it will be missed. The energy checking process is based on algorithm 3, which will be described next in this section.
  - · Finally, the current serving drones will make offloading as in step 18.
- $*$ Consider the second option in step 19, when there are not enough drones in the neighbor cells, but they can be completed or sent all from the charging station.
  - · In this case, in step 20, the needed drones are sent from neighbor cells and/or from the charging station.
  - · According to that, in steps 21 and 22, the number of available drones in neighbor cells and charging station are updated consequently.
  - · Next, in step 23, the number of drones that are currently serving in that cell is updated.
  - · As we did before, we will check first the energy level of each current serving drone's before applying the offloading action based on algorithm 3 as in step 24.
  - · Finally, in step 25, these sent drones will make offloading.
- $*$ Consider the last option in step 27, when there are not enough charged drones available neither in the neighboring cells nor in the charging station.
  - · In this case, in step 28, the available drones are sent from neighbor cells and/or from the charging station. To be noted, these drones are less than the cell's need.
  - · Then, the number of available drones in neighbor cells and charging station are updated to be zero as in steps 29 and 30 consequently.
  - · Next, the number of drones currently serving in that cell is updated as in step 31.
  - · Now, in step 32, we check first the energy

level of each current serving drone based on algorithm 3 before applying offloading action.
          · Finally, in step 33, these sent drones will make offloading.
    – Consider the case of a non realtime traffic
        ∗ The best solution in this case, is to buffer the traffic until free resources being available or it reaches its maximum time limit as there is no obligation to send it now as in step 38.
- Conversely, in step 41, when there is no unnecessary BW in the cell, and if there are drones existing, so:
    – In step 42, we calculate the number of unneeded drones.
    – Wherefore, the status of these drones will be changed to idle as in step 43.
    – Finally, in step 44, the number of currently serving drones in that cell is updated.

To be noted, the scenario of non-real time traffic has not been taken into account here and will be contributed in the future work.

Algorithm 2: Idle Drones Energy Management (Rule-based)

```
for each drone in neighbor cell list E(t) do
    // Check drone's battery level
    if Idle drone energy (B_R) < (offloading energy + return energy)
     then                                          // Battery level
        a = 0 // Back the drone to the charging
            station for recharging
        E(t) = E(t) − 1
        L(t + 1) = L(t) + 1
    end
end
```

Algorithm 2 is concerned with managing the energy of idle drones in the list of neighbor cells. The algorithm begins by comparing the energy of each idle drone in the list of neighbor cells with the energy needed for offloading and return actions. Accordingly, if the energy of the idle drone is less, then it is returned to the charging station for recharging. Next, the number of drones in neighbor cells list is updated on the current time(t). Also, the number of fully charged drones is updated at the charging station but based on the advanced time$(t+1)$ as charging takes time considering that the drone charging time equals to one time period.

Algorithm 3 is concerned with managing and optimizing the power of the drones as well as compensating for the returned drones. It will be described in bolt steps for better understanding as follows:

- In each cell, check the battery level for each current drone.
- When the battery level of any drone reaches less than the level required for the offloading and return actions, then:
    – This drone will be sent back to the charging station for recharging.
    – Then, in order to compensate for the returned drone, we have three conditions for different options that are processed sequentially.

Algorithm 3: Drones Energy Management (Rule-based)

```
for each serving drone in list S_i(t) do
    // Check drones' battery level
    if Serving drone energy (B_R) < (offloading energy + return energy)
     then                                          // Battery level
        a = 0 // Back the drone to charging station
            for recharging
        if E(t) > 0 then
            a = 1 // Send idle neighbor Drone to
                compensate returned one
            E(t) = E(t) − 1 // Update available Drones
                in the neighbor cells
        else if E(t) = 0 AND L(t) > 0 then
            a = 1 // Send idle charging station
                Drone to compensate returned one
            L(t + 1) = L(t) − 1 // Update available
                Drones in the charging station
        end
        else
            S_i(t) = S_i(t) − 1 // Update current
                SERVING Drones in the cell
            L(t) = 0 // Update available Drones in
                the charging station
        end
    end
end
end
```

        ∗ Option 1: Send a replacement drone from neighbor cells, if there is one available there.
        ∗ After that, the number of available drones in neighbor cells is updated.
        ∗ If option 1 is not valid, we proceed to the next option
        ∗ Option 2: Send a replacement drone from the charging station if there is one available there.
        ∗ Thus, the number of available drones in charging station is updated.
        ∗ In case of the previous two options are not valid, we proceed to the last one.
        ∗ Option 3: There are no replacement drone to be sent, if there is no available drones neither in the neighbor cells nor in the charging station.
        ∗ Thus, number of drones that are currently serving in that cell is updated.
        ∗ Finally, the number of available drones in charging station is updated.

V. Reinforcement Learning Solution

MDP is used to formulate the cellular network resources optimization problem, which describes the probabilities of the state of the environment. Thus, it is proposed to use RL to maximize the reward of the system and provide optimal decisions [18]. The RL model consists of an agent which acts as a controller taking action based on the states of the cellular network environment and these states will be rewarded accordingly. These actions control the drones to optimize its energy and the cell resources. This agent learns the best actions to take from the rewards received based on the states of the environment. We have defined the different conditions that the RL needs as they will be described later in algorithm 4.

- **State:** It is the system monitoring that the RL agent receives and accordingly action will be taken. In our solution, the case observes 6 conditions described in the algorithm 3. These conditions are as follows:
  - Availability of charged drones in the cell
  - Need for extra BW
  - Availability of idle drones in the neighbor cells
  - Availability of charged drones in charging station
  - Traffic type based on time
  - Energy of the drone
- **Action:** The chosen action defines the next states. The action space considered in this context is:
  - *Send:* Agent sends a drone to BS.
  - *offloading:* The agent order the drone to use its communication resources to work in BS.
  - *Charge:* The agent sends a drone back to the charging station for recharging.
  - *Idle:* The agent asks the drone to be idle in the existing cell.
- **Reward:** It is a result given to the RL agent when it takes a right action that is appropriate for a particular state. The result is a binary value that can be a reward or a penalty that is calculated after an action is taken. The purpose of the reward to let the RL agent learn the correct action to take for each state.

MDP is defined as a tuple [i;s;a;r] which is defined as follows:

- i: stands for the cell id under study.
- s: stands for the set of states considered in this model by increasing the number of serving drones in cell(i) at time(t).
- a: stands for actions taken by the agent.
- r: stands for the immediate reward that the agent will get according to taking decision a.

RL algorithm is used to solve the MDP process by Q-learning technique based on the following updated function.

$$
\begin{aligned}
Q_i(s(t), a(t)) &= (1-\alpha) * Q_i(s(t), a(t)) \\
&+\alpha * (r_i(t) + \gamma * max_a Q_i(s(t+1), a))
\end{aligned} \tag{9}
$$

A $Q_i(s(t), a(t))$ matrix is used to store the learned reward/sanction for each state action at time(t). Typically, $Q_i(s(t), a(t))$ is the expected reward for taking an action(a) at time(t), and where:

- α: denotes the learning rate which models here how quickly the Q-values can change with the dynamic feedback.
- γ: refers to the discount factor. It indicates the immediate reward status against the future one (the importance of the future reward).
- $r_i(t)$: represents the expected immediate reward of choosing action(a) at time(t), $r_i(t) = r(s(t), a(t))$
- $max_{a'} Q_i(s', a')$: represents the maximum expected future reward when the system reaches the state($t+1$) after taking action(a).

Algorithm 4: Cellular Resources Optimization Based Reinforcement Learning

```
for each cell(i) do
    Initialize Q_i(s, a) table
    Initialize s
    for each time(t) do
        Initialize the agent
        if there are charged drones in the cell then
            // (Condition 1)
            M_1(i, t) = 1
            else if There are drones in the cell out of charge then
                M_1(i, t) = 0
            end
        end
        if there is a need for extra BW (BW ≥ ρ) then
            // (Condition 2)
            M_2(i, t) = 1
            else if there is extra BW in the cell then
                M_2(i, t) = 0
            end
        end
        if there is an idle drone in a neighbor cell then
            // (Condition 3)
            M_3(i, t) = 1
            else
                M_3(i, t) = 0
            end
        end
        if there is a charged drone in charging station then
            // (Condition 4)
            M_4(i, t) = 1
            else
                M_4(i, t) = 0
            end
        end
        Classify traffic into real-time and non-real-time
        if there is real-time traffic then
            // (Condition 5)
            M_5(i, t) = 1
            else
                M_5(i, t) = 0
            end
        end
        for each drone(j) in the cell do
            Measure the residual energy of each drone (j)
            if drone energy > (offloading energy + return energy) then
                // (Condition 6)
                M_6(i, t) = 1
                else
                    M_6(i, t) = 0
                end
            end
        end
        s_i(t) ← M_{1-6}(i, t)
        while Final state not reached do
            Choose action a(t) for state s_i(t)
            Calculate Q-value
            Execute action a(t) and calculate next cell state s_i(t + 1)
            Update Q_i(s_i(t), a(t)) table
        end
    end
end
```

Before the reward is mentioned, algorithm 4 of the RL optimization will be described. This is because the model reward depends on its' conditions. Algorithm 4 starts by initializing $Q_i(s, a)$ table. Moreover, there are six conditions checked at each cell of the network, which are as follows:

- Condition 1: If there are charged drones in the cell, the value of $M_1(i, t) = 1$ and if they have a low energy and have to return to the charging station for recharging, then the value in this case of $M_1(i, t) = 0$.
- Condition 2: If there is a need for extra BW in this cell, then the value of $M_2(i, t) = 1$, Conversely, if there is an

abundance of BW in the cell, then the value of $M_2(i,t) = 0$.

- Condition 3: If there are idle drones in the neighbor cells, then the value of $M_3(i,t) = 1$, otherwise there are no idle drones in the neighbor cells and the value of $M_3(i,t) = 0$.
- Condition 4: If there are charged drones in the charging station, then $M_4(i,t) = 1$, otherwise the charging station is empty and $M_4(i,t) = 0$.
- : Following to that, traffic of the cell is classified into real-time and non-real-time.
- Condition 5: If there is real-time traffic, the value of $M_5(i,t) = 1$, and in the case of non-real-time traffic, the value of $M_5(i,t) = 0$.
- Condition 6 is formed based on the energy of the drone. Therefore, the energy of each drone of the cell is measured.
- Condition 6: If the measured energy is enough for offloading and there will be enough left to return to the charging station, then the value of $M_6(i,t) = 1$, otherwise the value of $M_6(i,t) = 0$.
- The state of the system reflects the value of each condition.
- As not all the states are reached, the agent choose action for each state, calculate the Q-value, execute the action, and build the $Q_i(s_i(t), a(t))$ table.

Now, we mention the reward equation, which will be calculated after each action as follows:

$$r_i(t) = \frac{\sum_{i \in \Omega_{\text{cells}}} D_{i,R}(t)}{\sum_{i \in \Omega_{\text{cells}}} \mathbb{1}_{\{C_i(t)\} > \xi\}}} \tag{10}$$

Figure 2 shows the finite automate with the different states and actions. The reward isn't present in this figure since we are assuming that the agent is taking the optimal action and the award is maximal here. For one day and one-hour granularity if the agent is making only optimal actions the reward will be equal to 24.

If we consider $t = 2$, our state is as follows [2,1,80,0], which means that there is an anomaly, the drone battery is equal to 80% and the drone is deployed. The optimal action, in this case, is offloading.

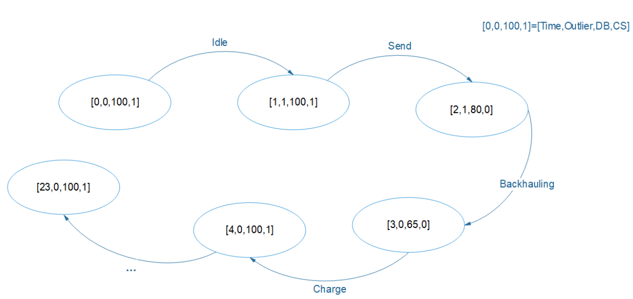

Fig. 2: Finite automate for one day

VI. Performance Evaluation

A. Dataset Description

The proposed anomaly detection framework can be applied to any dataset of users' activity times-series. In essence, we consider the CDRs provided by *Telecom Italia* as part of the big data challenge context [19]. It is a rich and open multi-source aggregation of telecommunications, weather, news, social networks and electricity data. This dataset contains over 319 million user-activity records for a 10,000 squares having a $235m \times 235m$ size spread across Milan, Italy. The records are for a two-months duration from November 1st, 2013 to January 1st, 2014 and divided into 10-minutes timestamps. Personal users data are removed to preserve privacy. Hence, the raw data contains five user-specific activity features: incoming and outgoing SMS, incoming and outgoing call, Internet usage and the geographical location (Cell ID).

In our study, we select the downtown (city center) of Milan city and presented by cell ID = 5060 and we study the Internet activity of users. We consider one type of data: RealTime which is uploaded immediately on the network.

B. Results and discussion

Rule-based algorithms are designed based on predefined rules and conditions that are often simpler in terms of processing complexity compared to sophisticated algorithms. On the other hand, in terms of performance, rule-based algorithms are suitable for well-defined rules and patterns. They can perform well in scenarios where the optimization problem can be easily expressed in a set of rules. However, they may not suitable for problems with complex, ambiguous, or evolving patterns.

We tested both the rule-based and the reinforcement algorithms on the Milan dataset. The reinforcement one (below called as "MDP") has been trained on the November part of the dataset, while it is tested on the part of December. In this section, we present the comparison of the rule based, the reinforcement and a third policy of drone assignment: the random drone allocation where the set of all available drones (i.e. with sufficient battery level) are assigned to a subset of the demanding cells. This subset is chosen the following way: if, at a certain time, there are no sufficient drones to satisfy all the cells with extra demand, but only a subset of these cells, a subset is chosen randomly and uniformly among the demanding cells. Of course, if there are sufficient drones, all the demanding cells are served. Note that, either for the reinforcement method, or the random one, when more cells demand extra bandwidth than there are available drones, a drone assigned to a demanding cell at $t-1$ will remain at the same cell if the same cell has still a demand at t. If it is not the case, it may move to a neighboring cell, according to the demand and policy (either MDP or random), but not further but to go to the charging station. Thus, if there is a choice to do between different drones allocation, it resides in the neighboring cell possibilities. We observe that often there this set of possible choices may be restricted. It depends on the bandwidth threshold from which any more demand is considered as an extra one, i.e. requiring a drone to satisfy it. Also note that the battery level of all the drones resets at 3:00 am each day, as that is the time when there is generally no activity and its the moment to take profit of this inactivity to reset the drones. Of course, this battery reset is done whatever the drone assignment algorithm: MDP, random or rule-based.

As we said, the rule-based is simpler to implement than the reinforcement method, more suitable for redundant situations but in contrast less adaptive to system evolution. Besides these facts, we show in this part dedicated to performance evaluation, that, of course, when the number of drones or the battery capacity increases or when the charging time decreases, the losses decrease whatever the algorithm. But, in contrast to the rule-based we exhibit here how the MDP is able to minimize the *future* losses, while the rule-based takes decision with only the information available at current time t. For example, the below section VI-B2 about the impact of the number of drones discusses this point. Also, it is interesting to note that the rule-based is less sensitive to an increase in charging time compare to MDP, especially in December when the demand is higher.

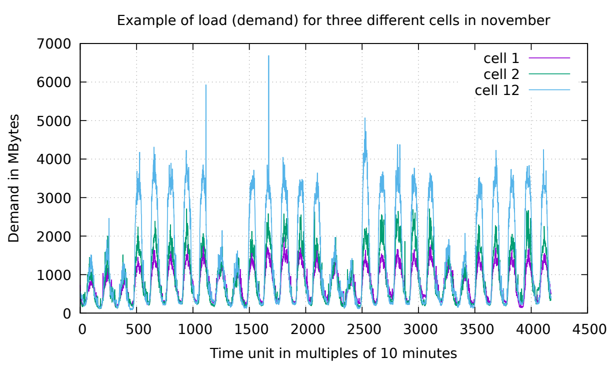

Fig. 3: Time series of the bandwidth demand in function of the time (in multiples of 10 minutes) for the month of November and three different cells

1) Shape of the demand: To understand the behaviour of the algorithms when comparing them, it is necessary to understand the shape of the traffic. Figure 3 presents the time series of the demand for three typical cells in function of the time for the month of November, while figure 4 presents the same for the month of December. Figure 5 is the same but zoomed on a specific day. We can observe that the demand is heterogeneous, depending on the cells. It is also random with peak and off hours, which means that generally speaking, when a demand starts at certain time on a particular cell, it will probably continue for a certain duration even if it may be not the case, randomly.

For this evaluation, we observe the influence of four parameters: bandwidth threshold, number of drones, battery capacity and charging time. We plots the average losses *given* that there is a demand. For each demanding cell which requires

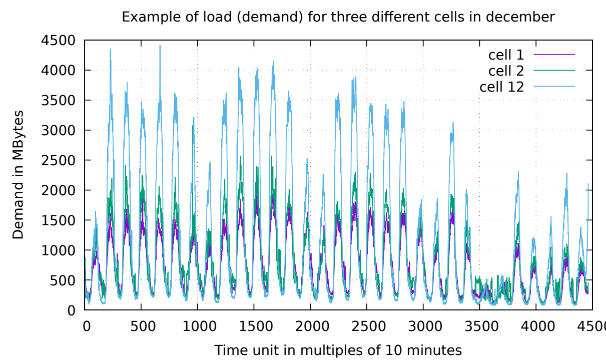

Fig. 4: Time series of the bandwidth demand in function of the time (in multiples of 10 minutes) for the month of December and three different cells

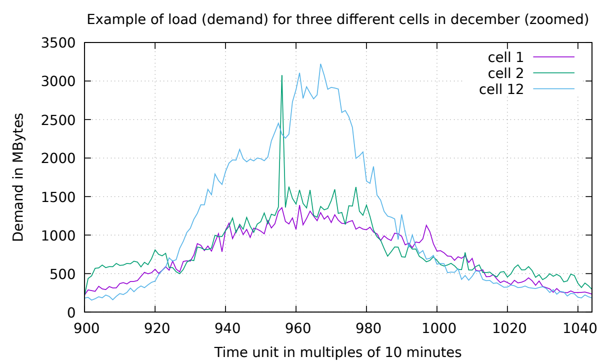

Fig. 5: Time series of the bandwidth demand in function of the time (in multiples of 10 minutes) for the month of December and three different cells, zoomed

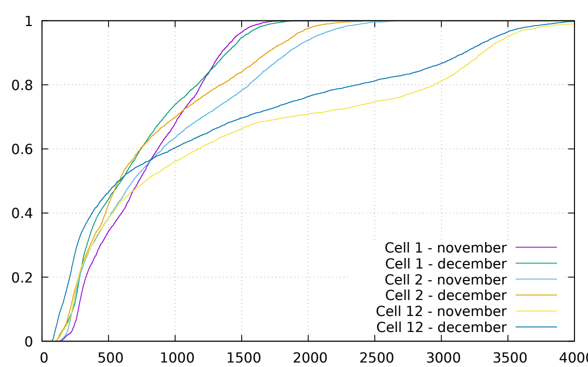

Fig. 6: Cumulative probability distribution of the bandwidth demand for three different cells for both November and December

more demand than the bandwidth threshold, if the cell is served by a drone all the demand is satisfied. Whereas, on the contrary, if it is not the case, the corresponding data are lost. At each time, the average loss rate is the amount of lost data divided by the number of cells requiring extra capacity without considering other cells. For example, at a given time t, if there are 6 cells demanding more than the bandwidth threshold and only two drones are available, if the assignment of these two drones is such as 4 cells are not satisfied, then cells loss is a total of L MBytes, the average loss rate is $L/4$ and not $L/15$ (15 is the total number of cells).

In figure 6, the cumulative distribution function of the bandwidth demand is given for three different cells in November and December. Generally speaking, the demand is higher in December than in November. This is important to understand the performance results presented hereafter.

At last, figure 7 represents the total demand of all the cells in function of the time for the month of December, but restricted to the time interval [23h;23h59]. We can observe a peak at the end of the period. This may be explained by the new year's evening. It has an impact on the average demand in function of the hour in the day, which is presented below.

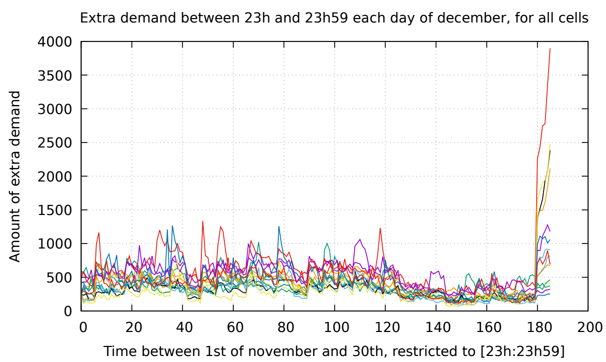

Fig. 7: Amount of extra demand in function of time between the 1st and the 31th of December, restricted to [23h;23h59], for all cellular cells

2) Impact of the number of drones: Figure 8 shows the average data loss in function of the number of drones for a bandwidth threshold of 2500Mbytes, a battery capacity of 100 minutes, and 30 minutes charging time, for the three policies (MDP, rule-based and random), in November and December. Of course, when the number of drones increases, the amount of lost data decreases. The lowest loss is observed for the MDP in November. It is expected since MDP is trained on the data set of November and it should converge to the optimal drone allocation. The rule-based in November is worst than the MDP but also worst than the random allocation. Although surprising, this is understandable since once a drone is allocated to a cell, it won't be moved if there is still a demand in the same cell the next time instants. But, if a cell may be a good choice at a given time t, it may not be the case anymore in the next time instants. The MDP learns how to take this feature into account by optimising the *future* expected reward, that is why it is better than the rule-based. The fact that the random policy is better is due to the fact that sometime it chooses the best allocation that the MDP would also choose. In other words, the rule-based is affected by its inability to predict the future.

Looking at December, the results are a little different. Actually, the MDP is still better than the random policy but the rule-based is almost as good as the MDP when the number of drones is small. This is explained by the fact that the demand in December differs from the demand in November. So, the behaviour learned from the data of December is not always suitable anytime. Also, as shown above, the demand in December is slightly higher than in November. These two factors make the MDP less efficient.

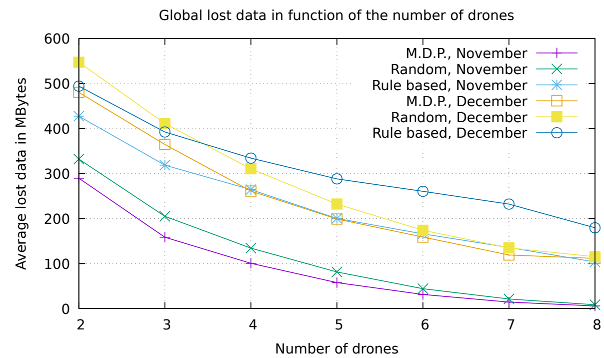

Fig. 8: Average number of lost data in function of the number of drones for 2500Mbytes threshold, a battery capacity of 100 minutes and 30 minutes charging time

Figures 27 and 28 (given in Annex) show the same results as figure 8, but for work and OFF days respectively. The results are more or less the same. The main difference is due to the amount of data changing between the working days and OFF days. For working days, there is relatively uniform demand, while on weekends it is larger and more chaotic. Actually, the average amount of demand, *given that* there is a demand above the bandwidth threshold, in the working and OFF days is shown in Annex in figures 29 and 30 for November and figures 31 and 32 for December in function of the time of day. In addition, there is less data measured for weekends, so that the averages are calculated in light of demand using less data. That is why the behaviour of the rule based is more chaotic. Note that having one more drone may be useful sometimes to satisfy certain bandwidth but in some pathological cases it may degrade the performance. For instance, a drone $d1$ may stay in cell $c1$ without the need to serve it because there is no demand at $c1$ and another (additional) drone $d2$ is serving a neighboring cell $c2$ (this cell $c2$ will be served by $d1$ if $d2$ is not there). Thus, $d1$ will be available a longer time on the field. But, it will need to return to the charging station at a certain time which is later than in the case where $d2$ does not exist (if there is one drone less). Hence, it may be not available at the time where it could be needed another cell $c3$ where the demand may be high and the additional drone $d2$ may remain occupied in a less important cell. This problem is still due to the fact that the rule-based algorithm cannot predict when and where the important demand will occur.

Another interesting result is the average loss as a function of the time of day. The Figure 9 shows the average data loss given there is a demand in function of the time of day, with a 2500Mbytes bandwidth threshold, a battery capacity of 100 minutes, 30 minutes charging time, for 2 and 7 drones, for the month of November, together with the average extra demand, that is the average demand above 2500Mbytes, *given* that there is an extra demand. Of course, there are less losses with 7 drones than with 2 drones. Also, the MDP is better than the random policy which itself is better than the rule-based algorithm. What is interesting is to observe jagged results. These oscillations are due to the periodic lack of drones because of needs to charge the battery. Remember that all the drones are synchronised at the beginning of the day since their state of charge is reset at 3:00 am. However, these results show a clear improvement in the use of drones to alleviate the data loss compared to no additional resources (case of the curve titled "total extra demand" which corresponds to the average extra demand given that there is a demand). Figures 29 and 30 present the same results but are limited to working days and OFF days. The same kind of results can be observed, depending on the amount of data, which is typically less for working days (remember we consider a conditional demand given that there is a demand above the bandwidth threshold).

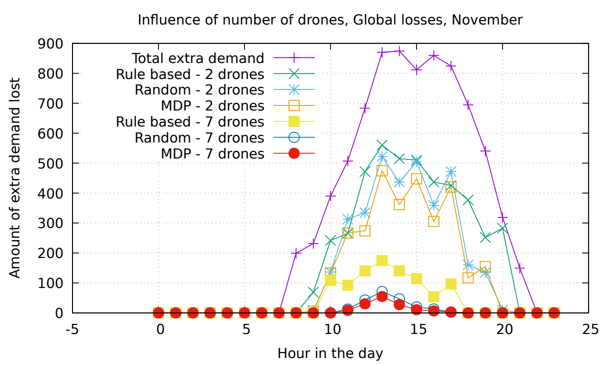

Fig. 9: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, a battery capacity of 100 minutes, a charging time of 30 minutes and for different number of drones

Figures 10 and figures 31 (in Annex) and 32 (also in Annex) give the same results but for the month of December. The demand pic at 23h is due to the end of year's evening (not that this day is a work day). These results exhibit clearly the trade off that can be found between placing drones in cells with immediate reward or in cells for which immediate reward may be low but becomes higher in the future. Remember that a drone does not move as long as there is a demand for the cell in place, whatever the drone allocation policy. Considering the case of the global average loss rate (Figure 10), for 7 drones, the average loss rate is smaller at the beginning of the day for the rule-based than for the MDP but it gets high as the time of day progresses, and globally the loss rate is smaller for the MDP than the rule-based.

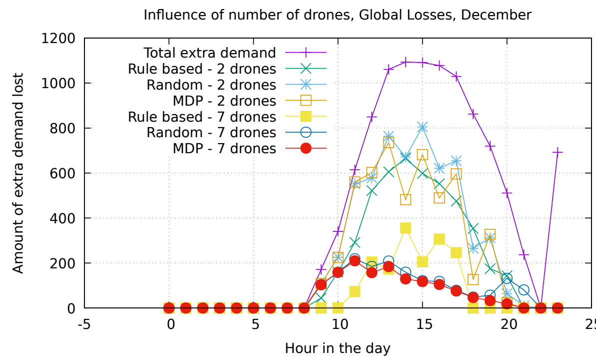

Fig. 10: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, a battery capacity of 100 minutes, a charging time of 30 minutes and for different number of drones

3) Impact of the battery capacity: The impact of the battery capacity is investigated in figure 11 which represents the average loss rate given there is a demand, for a 2500Mbytes bandwidth capacity, 30 minutes charging time and 5 drones. The battery capacity is given in multiples of 10 minutes. The higher the battery capacity, the fewer times a drone has to return to the charging station and the higher the number of drones, on average, so the loss rate is lower. For the same reasons as before, i.e. a higher demand and a test on a dataset that is slightly different from the training, the results are slightly worst in December than in November.

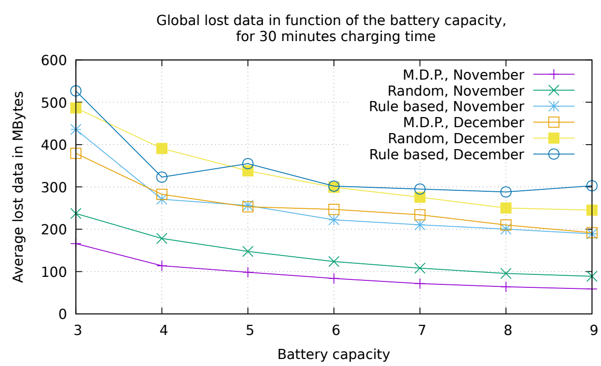

Fig. 11: Average number of lost data in function of the battery capacity for 2500Mbytes threshold, 5 drones and 30 minutes charging time

Figure 12 presents the same results as figure 11 but for a charging time of 100 minutes instead of 30 minutes. Of course, the loss rate is higher due to the increased charging time which means less active drones on average. But it is interesting to note that the rule-based is less sensitive to an increase in charging time, especially in December when the demand is higher. Therefore, it gives good results compared to the two other assignment methods. Generally speaking, we can conclude that although the rule-based is affected by the later bad consequence of an assignment, this is alleviated when the demand is high compared to the resources.

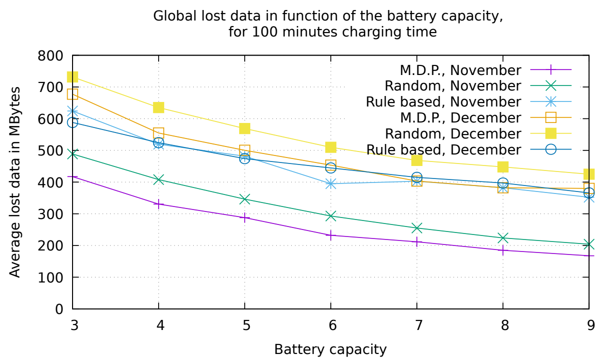

Fig. 12: Average number of lost data in function of the battery capacity for 2500Mbytes threshold, 5 drones and a charging time of 100 minutes

The same average loss rate in function of the battery capacity respectively for 30 minutes charging time for working days and OFF days, and for 100 minutes charging time for working days and OFF days can be found in Annex figures 33, 34, 35 and 36.

The evolution of the loss rate in function of the time is also presented in Annex: the figures 37 and 39 present the loss rate in function of the time of day for 30 minutes charging time for working days, respectively for November and December, while figures 38 and 40 are for OFF days. The same results are presented figures 41, 43, 42 and 44 for 100 minutes charging time. Of course, a longer charging time leads to worst loss rates.

4) Impact of the charging time: Figure 13 presents the global loss rate in function of the charging time for a 2500Mbytes bandwidth threshold, 8 drones and a battery capacity of 100 minutes. Although there is a clear impact of the charging time on MDP and the random drone allocation policies, it is less the case for the rule based. Here, there are a lot of drones, 8, which makes that the loss rate of the rule based depends mainly on the spatial drone availability, which is a function of their first placements and thus a function of the data. The ability of the MDP to predict future rewards is a clear advantage for it. Note that the considered loss rates are low (compared to the similar previous curves like figures 8, 11 and 12). Figures 14 and 15 show the same thing for ON and OFF days.

Figures 16 and 18 show the evolution of the loss rate in function of the time of day respectively in November and December, for working days, 8 drones, a 2500Mbytes bandwidth threshold and a battery capacity of 100 minutes while figures 17 and 19 are for OFF days in November and December respectively.

5) Impact of the bandwidth threshold: Figure 20 shows the global loss rate in function of the bandwidth threshold, for 8 drones, 100 minutes battery capacity and 30 minutes

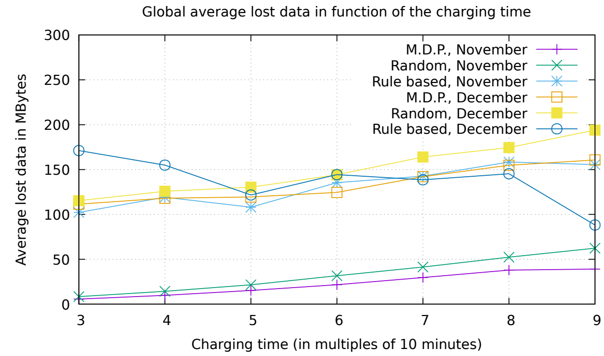

Fig. 13: Global average number of lost data in function of the charging time for 2500Mbytes threshold, 8 drones and battery capacity 100 minutes

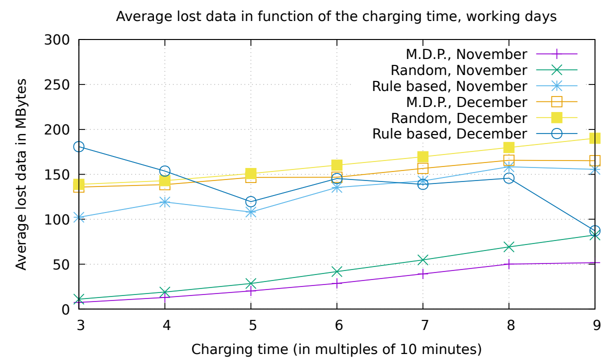

Fig. 14: Average number of lost data for working days, in function of the charging time for 2500Mbytes threshold, 8 drones and battery capacity 100 minutes

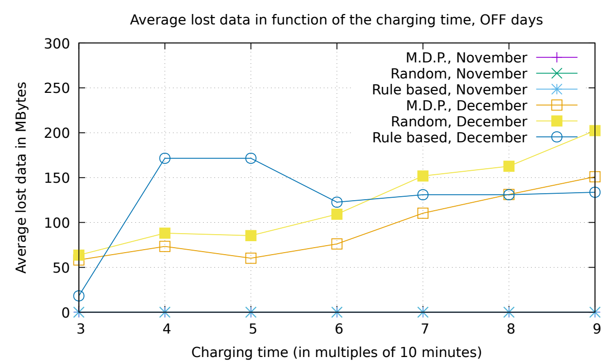

Fig. 15: Average number of lost data for OFF days, in function of the charging time for 2500Mbytes threshold, 8 drones and battery capacity 100 minutes

charging time. Without surprise, the MDP comes out with the best performance in November, then the random policy and finally the rule-based algorithm. In December, the MDP and the rule-based exhibit the same performance except for the high 2500Mbytes threshold where it is worse. The fact

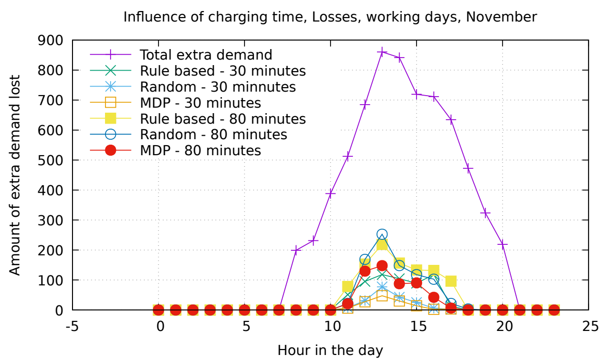

Fig. 16: Average number of lost data for working days in November in function of the time of day for 2500Mbytes threshold, 8 drones, a battery capacity of 100 minutes and different charging times

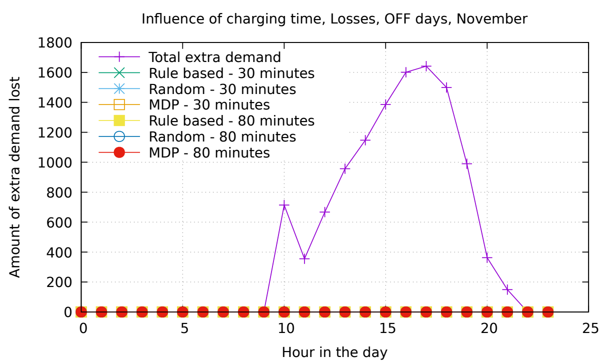

Fig. 17: Average number of lost data for OFF days in November in function of the time of day for 2500Mbytes threshold, 8 drones, a battery capacity of 100 minutes and different charging times

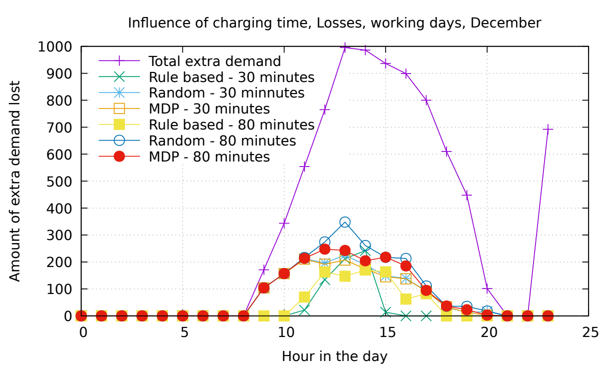

Fig. 18: Average number of lost data for working days in December in function of the time of day for 2500Mbytes threshold, 8 drones, a battery capacity of 100 minutes and different charging times

that the losses of the rule-based increase at 2500Mbytes in December can only be attributed to the shape of the peaks of

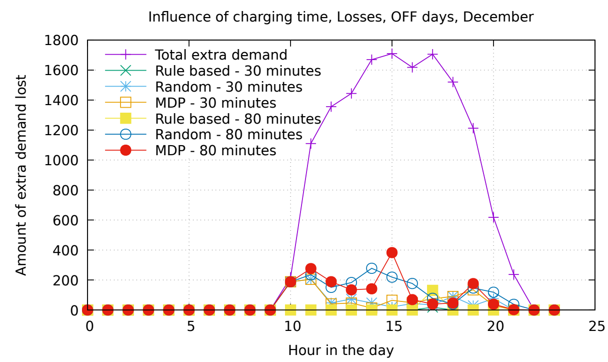

Fig. 19: Average number of lost data for OFF days in December in function of the time of day for 2500Mbytes threshold, 8 drones, a battery capacity of 100 minutes and different charging times

demand in the working days (i.e. the number of simultaneous peaks and their intensity), which can be confirmed knowing that the rule-based algorithms reacts poorly to a "bad future" and by observing the figures 23, 25, 24 and 26 representing the same quantity but in function of the time of day. Actually, while the losses of the rule-based are low for the OFF days, higher during the working days and higher in December than in November, the demand is higher during the OFF days than during the working days.

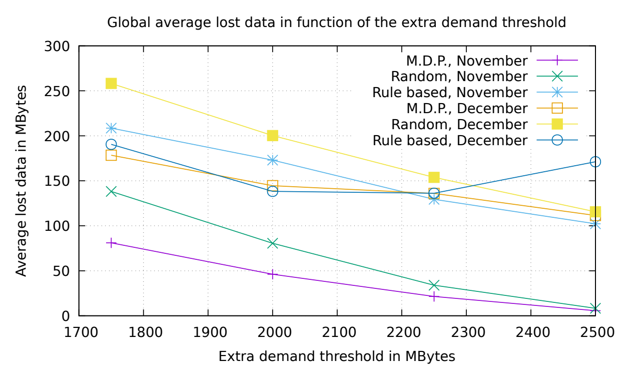

Fig. 20: Global average number of lost data in function of the capacity threshold in MBytes, 8 drones, 30 minutes charging time and battery capacity 100 minutes

6) Tabular summary of quantitative results: In order to highlight the advantages of our proposed approaches with the experimental data, in function of the context, we summarize here the numerical results given above in figures 8, 11, 13 and 20. For each changed parameter, the number of drones, the battery capacity, the charging time or the extra demand bandwidth threshold, we compute the average of the loss rate and we compare the averages in table III.

Also, in table IV are represented the relative differences in percentages of these averages with the average loss rate of the random algorithm. As already noticed previously, the

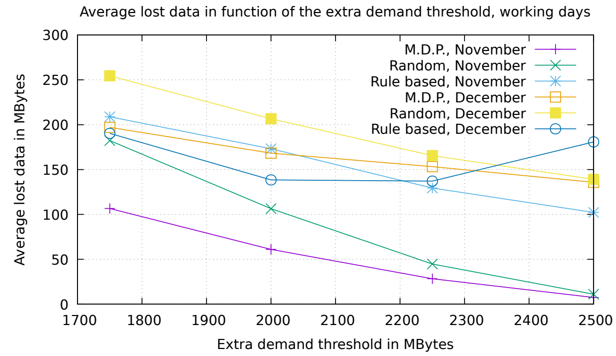

Fig. 21: Average number of lost data for working days, in function of the capacity threshold in MBytes, 8 drones, 30 minutes charging time and battery capacity 100 minutes

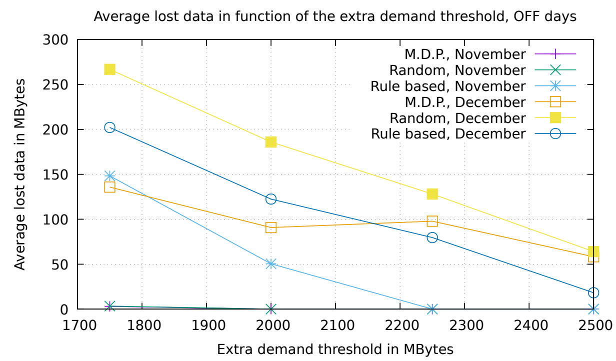

Fig. 22: Average number of lost data for OFF days, in function of the capacity threshold in MBytes, 8 drones, 30 minutes charging time and battery capacity 100 minutes

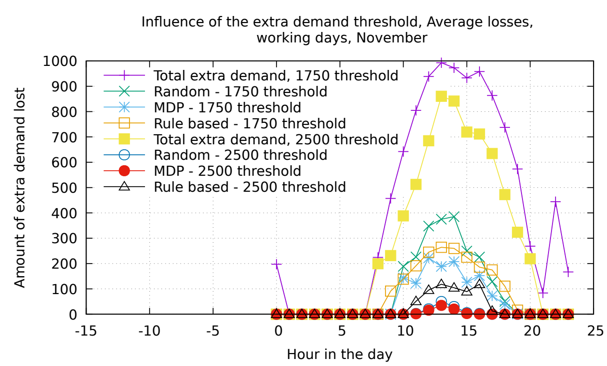

Fig. 23: Average number of lost data for working days in November in function of the time of day for 8 drones, a battery capacity of 100 minutes, 30 minutes charging time and different capacity thresholds

percentage of losses is negative for the MDP, meaning that it decreases the losses compared to a random allocation. Also, this loss decreasing is attenuated for the month of december where the learning performs less good. In contrast,

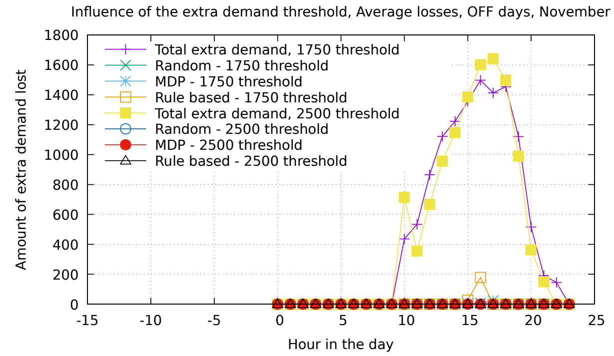

Fig. 24: Average number of lost data for OFF days in November in function of the time of day for 8 drones, a battery capacity of 100 minutes, 30 minutes charging time and different capacity thresholds

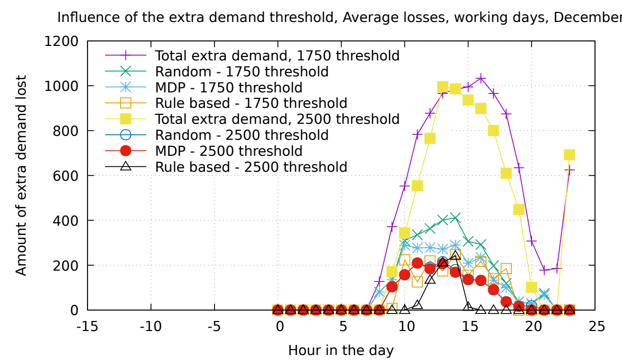

Fig. 25: Average number of lost data for working days in December in function of the time of day for 8 drones, a battery capacity of 100 minutes, 30 minutes charging time and different capacity thresholds

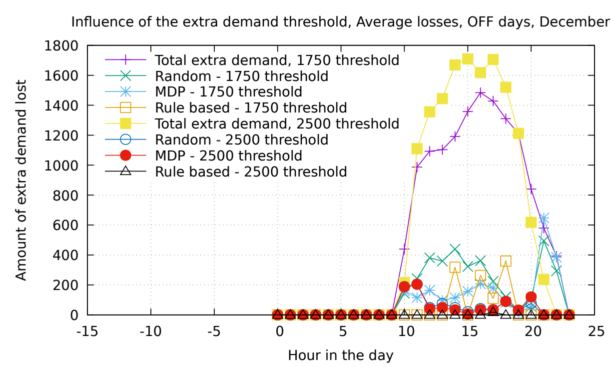

Fig. 26: Average number of lost data for OFF days in December in function of the time of day for 8 drones, a battery capacity of 100 minutes, 30 minutes charging time and different capacity thresholds

the rule-based performance is not good for the month of november while it is not really bad for the month of december

	Random		MDP		Rule-based	
	Nov.	Dec.	Nov.	Dec.	Nov.	Dec.
Nber of drones	118	275	94	242	231	314
Battery capacity	140	327	94	257	255	342
Charging time	33	150	23	133	132	138
Demand threshold	65	182	39	143	153	159

TABLE III: Summary of the average loss rates presented on the figures above, compared for the different algorithms

	MDP		Rule-based	
	Nov.	Dec.	Nov.	Dec.
Nber of drones	-21%	-12%	96%	13%
Battery capacity	-33%	-21%	82%	4.7%
Charging time	-31%	-11%	297%	-8%
Demand threshold	-40.9%	-21.7%	135%	-12.6%

TABLE IV: Percentage comparison of the average losses for MDP and rule-based with respect to the random algorithm

without being excellent. This means that the rule-based is well suited for situation where learning is not easy nor possible. Remember also that its simplicity making it easy to implement is a key advantage for the rule-based.

VII. Conclusion and Future Work

We have proposed two approaches of optimal deployment algorithms for drones to satisfy extra-bandwidth demand. The first one is a rule-based algorithm which is not complex to implement and which can be embedded in small devices. In order to target the optimality of the drones placement, we also implemented a reinforcement method based on MDP. These placement methods have been evaluated on a real dataset of two month cellular demand in the city of Milan, Italy. The reinforcement algorithm has been trained over the month of November and its efficiency tested both with the data of November and with December in order to observe its sensitivity to the testing dataset. Of course, the performance is best in November than in December but it remains in most of the cases the best compared to the rule-based algorithm, even if both are quite close. The rule-based algorithm is penalized by the consequence on the near future of a possible bad placement at a given time, while the reinforcement method tries to learn the best future reward which is the lost demand rate.

During this work, a drone serves one cell at a time, even if it would have sufficient capacity to other neighboring cells. In a near future, we aim at investigating the effect of the cooperation between drones to share the load of several cells simultaneously between them. Hence, we could consider exploring the use of federated learning which is a distributed machine learning technique that enables continual model training in distributed wireless systems. It leverages a cooperative fusion approach where networked agents, connected via ultra-reliable, low latency communications, act as distributed learners that periodically exchange their locally trained model parameters. Moreover, the work could be enhanced by studying the actual charging time of drones and its residual charge.

References

[1] N. S. Labib, M. R. Brust, G. Danoy, and P. Bouvry, "The Rise of Drones in Internet of Things:A Survey on the Evolution, Prospects and Challenges of Unmanned Aerial Vehicles," *IEEE Access*, vol. 9, pp. 115 466–115 487, 2021.

[2] "A taxonomy of blockchain-enabled softwarization for secure uav network," *Computer Communications*, vol. 161, pp. 304–323, 2020.

[3] GlobalNewswire. Unmanned aerial vehicles (uav) market is expected to reach $21.8 billion by 2027- market size, share, forecasts, & trends analysis report with covid-19 impact by meticulous research. [Online]. Available: https://www.globenewswire.com/en/news-release/2021/09/23/2302421/0/en/Unmanned-Aerial-Vehicles-UAV-Market-is-Expected-to-Reach-21-8-Billion-by-2027-Market-Size-Share-Forecasts-Trends-Analysis-Report-with-COVID-19-Impact-by-Meticulous-Research.html

[4] R. ITU-T, "Functional architecture for unmanned aerial vehicles and unmanned aerial vehicle controllers using IMT-2020 networks," *Recommendation ITU-T Y.4421*, no. Y.4421, 2021.

[5] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, 2019.

[6] N. Cheng, S. Wu, X. Wang, Z. Yin, C. Li, W. Chen, and F. Chen, "Ai for uav-assisted iot applications: A comprehensive review," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 438–14 461, 2023.

[7] A. Dridi, M. Laroui, C. Boucetta, H. Afifi, and H. Moungla, "Reinforcement Learning Vs ILP Optimization in IoT support of Drone assisted Cellular Networks," in *ICC 2022 - IEEE International Conference on Communications*, 2022.

[8] S. A. Hadiwardoyo, C. T. Calafate, J.-C. Cano, K. Krinkin, D. Klionskiy, E. Hernández-Orallo, and P. Manzoni, "Three Dimensional UAV Positioning for Dynamic UAV-to-Car Communications," *Sensors*, 2020.

[9] Y. Liu, C.-X. Wang, H. Chang, Y. He, and J. Bian, "A Novel Non-Stationary 6G UAV Channel Model for Maritime Communications," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 10, pp. 2992–3005, 2021.

[10] A. Hamdi, F. D. Salim, D. Y. Kim, A. G. Neiat, and A. Bouguettaya, "Drone-as-a-Service Composition Under Uncertainty," *IEEE Transactions on Services Computing*, 2021.

[11] A. Rajagopal, G. P. Joshi, A. Ramachandran, R. Subhalakshmi, M. Khari, S. Jha, K. Shankar, and J. You8, "A Deep Learning Model Based on Multi-Objective Particle Swarm Optimization for Scene Classification in Unmanned Aerial Vehicles," *IEEE Access*, vol. 8, pp. 135 383–135 393, 2021.

[12] F. A. Almalki and M. C. Angelides, "Autonomous flying IoT: A synergy of machine learning, digital elevation, and 3D structure change detection," *Computer Communications*, vol. 190, p. 154–165, 2022.

[13] Y. Liu, J. Yan, and X. Zhao, "Deep Reinforcement Learning Based Latency Minimization for Mobile Edge Computing With Virtualization in Maritime UAV Communication Network," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4225–4236, 2022.

[14] K. L. B. Ramesh, "Multi head deep neural network prediction methodology for high-risk cardiovascular disease on diabetes mellitus," *Computer Modeling in Engineering & Sciences*, vol. 137, no. 3, pp. 2513–2528, 2023. [Online]. Available: http://www.techscience.com/CMES/v137n3/53744

[15] C. Boucetta, A. Dridi, H. Moungla, H. Afifi, and A. E. Kamal, "Optimizing drone deployment for cellular communication coverage during crowded events," in *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*, 2019, pp. 622–627.

[16] C. Boucetta, B. Nour, S. E. Hammami, H. Moungla, and H. Afifi, "Adaptive Range-based Anomaly Detection in Drone-assisted Cellular Networks," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 1239–1244.

[17] C. Boucetta, A. Dridi, H. Afifi, A. E. Kamal, and H. Moungla, "Heuristic Optimization Algorithms for QoS Management in UAV Assisted Cellular Networks," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 01–06.

[18] A. Kumari and S. Tanwar, "A reinforcement-learning-based secure demand response scheme for smart grid system," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2180–2191, 2022.

[19] T. Italia, "Telecommunications - SMS, Call, Internet - MI," 2015. [Online]. Available: https://doi.org/10.7910/DVN/EGZHFV

VIII. ANNEX

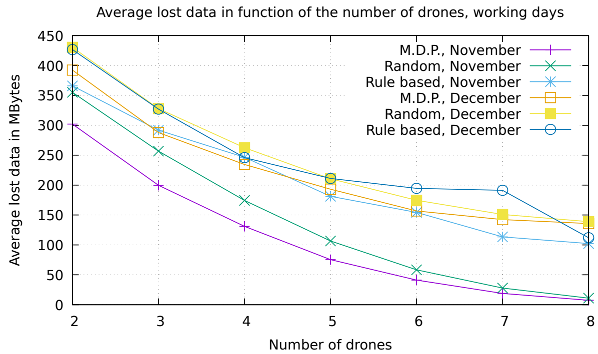

Fig. 27: Average number of lost data in function of the number of drones for 2500Mbytes threshold, a battery capacity of 100 minutes and 30 minutes charging time, for working days

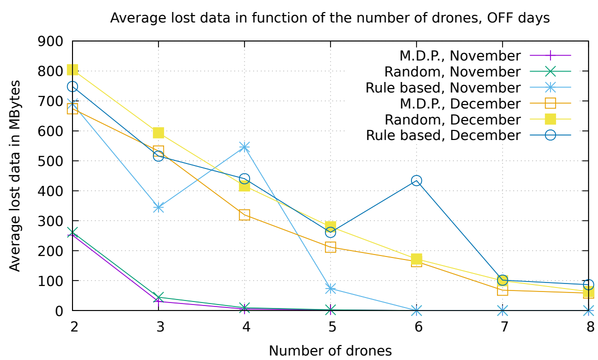

Fig. 28: Average number of lost data in function of the number of drones for 2500Mbytes threshold, a battery capacity of 100 minutes and 30 minutes charging time, for OFF days

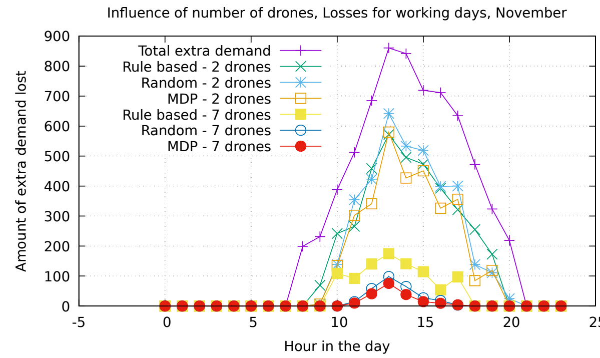

Fig. 29: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, a battery capacity of 100 minutes, a charging time of 30 minutes and for different number of drones, working days

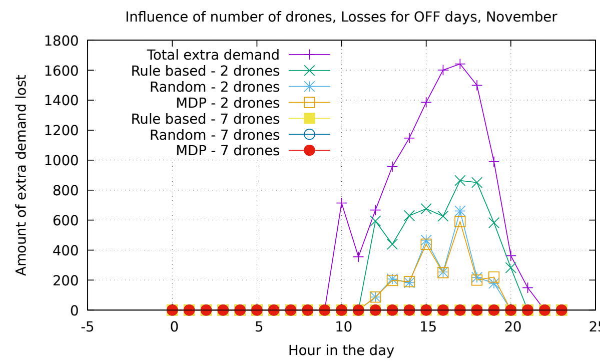

Fig. 30: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, a battery capacity of 100 minutes, a charging time of 30 minutes and for different number of drones, OFF days

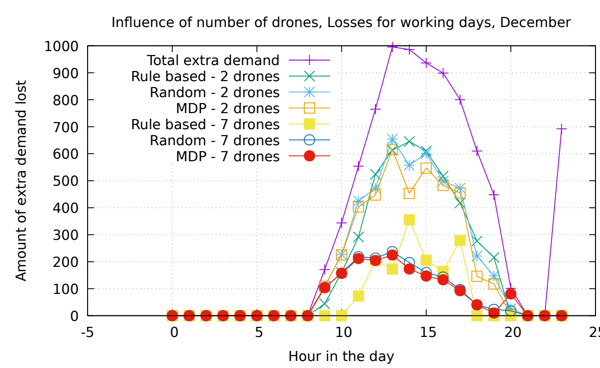

Fig. 31: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, a battery capacity of 100 minutes, a charging time of 30 minutes and for different number of drones, working days

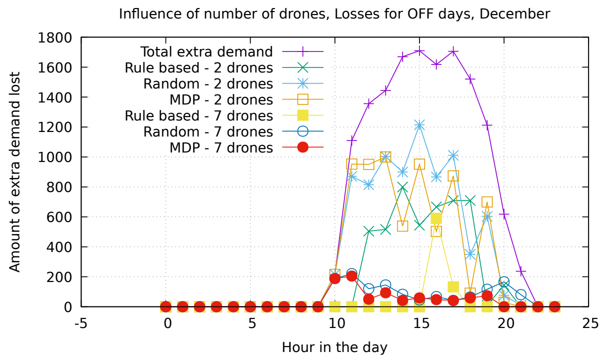

Fig. 32: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, a battery capacity of 100 minutes, a charging time of 30 minutes and for different number of drones, OFF days

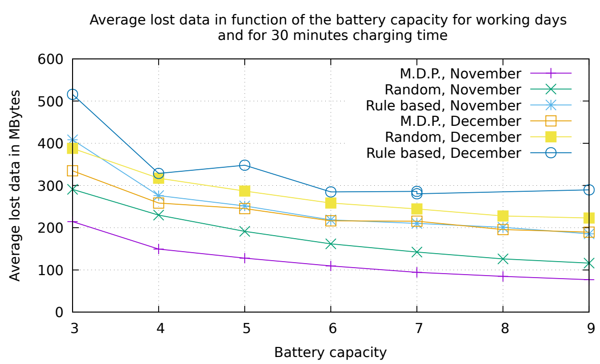

Fig. 33: Average number of lost data in function of the battery capacity for a 2500Mbytes threshold, 5 drones and 30 minutes charging time, working days

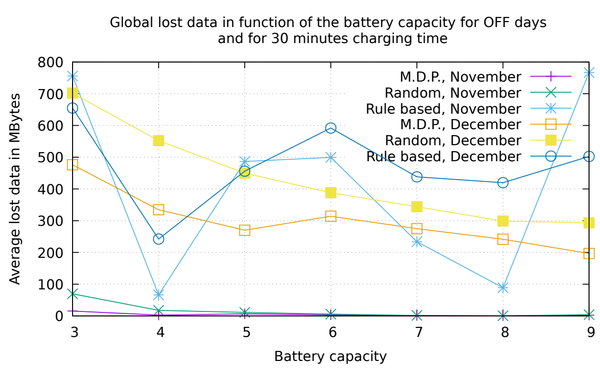

Fig. 34: Average number of lost data in function of the battery capacity for a 2500Mbytes threshold, 5 drones and 30 minutes charging time, OFF days

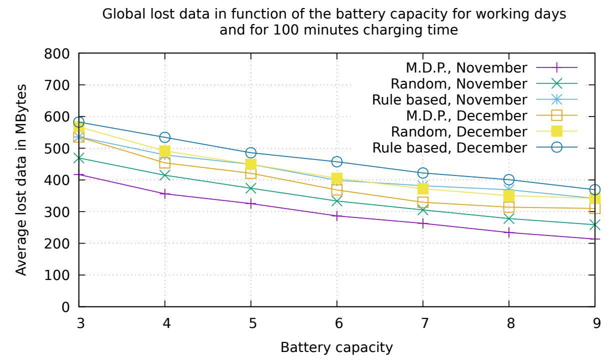

Fig. 35: Average number of lost data in function of the battery capacity for a 2500Mbytes threshold, 5 drones and a charging time of 100 minutes, working days

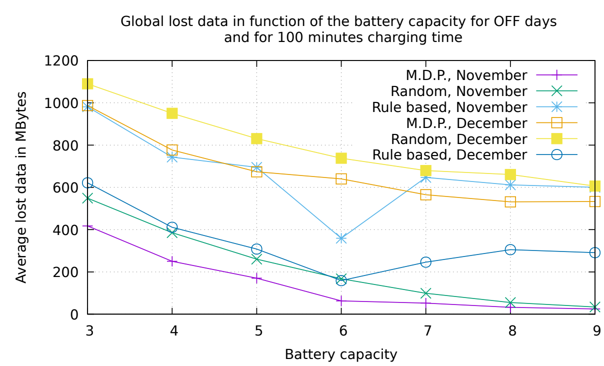

Fig. 36: Average number of lost data in function of the battery capacity for a 2500Mbytes threshold, 5 drones and a charging time of 100 minutes, OFF days

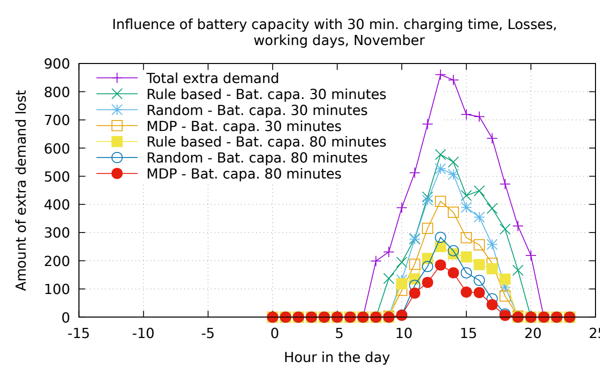

Fig. 37: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 30 minutes and for different battery capacities, working days

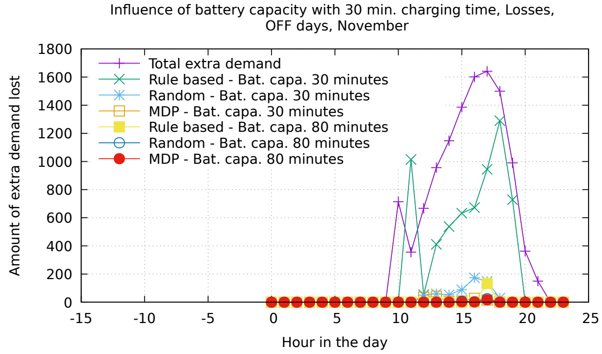

Fig. 38: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 30 minutes and for different battery capacities, OFF days

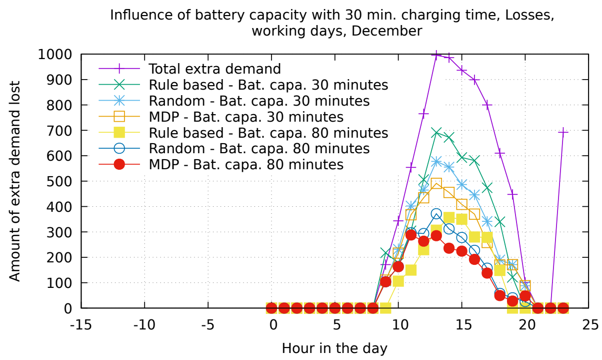

Fig. 39: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 30 minutes and for different battery capacities, working days

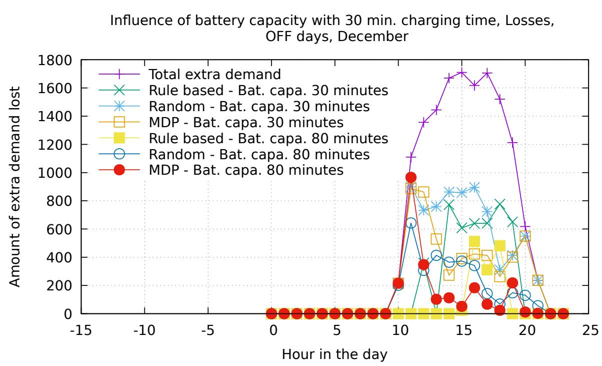

Fig. 40: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 30 minutes and for different battery capacities, OFF days

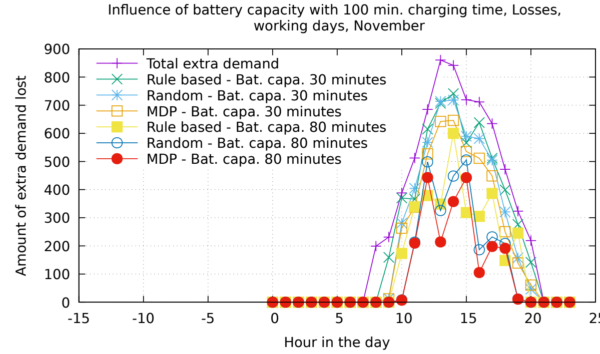

Fig. 41: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 100 minutes and for different battery capacities, working days

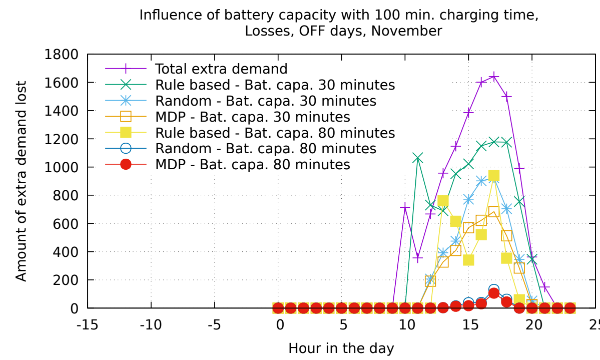

Fig. 42: Average number of lost data in November in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 100 minutes and for different battery capacities, OFF days

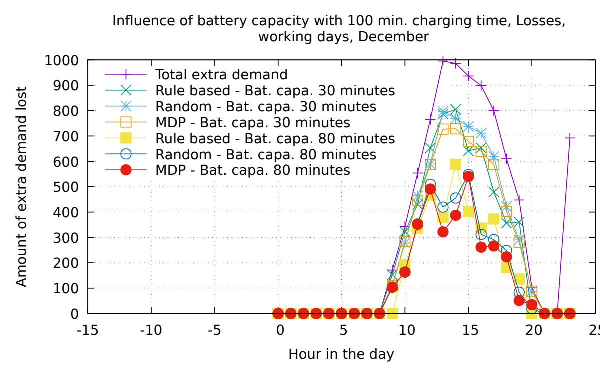

Fig. 43: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 100 minutes and for different battery capacities, working days

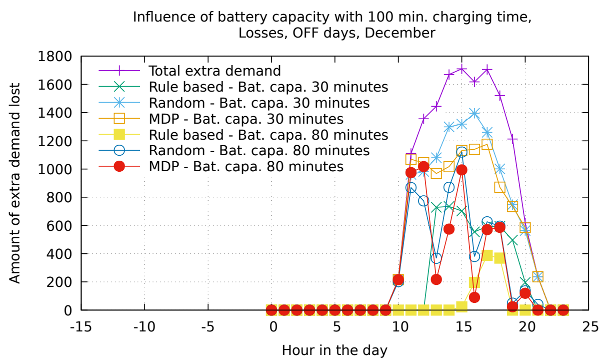

Fig. 44: Average number of lost data in December in function of the time of day for 2500Mbytes threshold, 5 drones, charging time of 100 minutes and for different battery capacities, OFF days