Management and content delivery for smart networks: algorithms and modelling - laboratory

Professors

Michela Meo
Daniela Renga

Students

Marco Colocrese
s301227
Francesco Donato
s304810

Politecnico di Torino
June 2022

1 LAB 1

1.1 Introduction

The purpose of this lab is to simulate the operation of a portion of an urban RAN (Radio Access Network) where terrestrial Base Stations are supported by renewable powered UAVs (Unmanned Aerial Vehicles), that are periodically sent to fly over a given area to provide additional capacity during peak periods enabling a better Quality of Service for the communication network. The network is modeled as a queuing system and in this lab activity we investigated its performance under variable configurations, to understand how different parameter settings may affect the system behaviour and how modifying some parameters of the network system may impact on performance metrics. Each aerial BS is equipped with multiple antennas and is modelled as a queuing system with m servers. In the considered queuing system, the customers represent the data packets that arrive at a given drone. The service represents the transmission of the data by the UAV mounted BS. Regarding the simulation, a next event approach is used, in which the system time jumps forward with discrete steps to the time instant at which the next event happens where the future execution of an operation is planned following the event scheduling rules. When an event is triggered, the scheduling of new events is executed. In our analysis of the network operation and performance we focused on changing mainly parameters such as the amount of drones involved, the amount of antennas exploited by each UAV, the buffer size, the analysis under different arrival rates conditions and the variation of the distribution of the service time.

1.2 Methods and parameters setting

The code has been structured in a modular way in order to assess performance analysis under different system conditions by setting a priori all the necessary parameters. The main ones are:

- IS_BUFFER_UNIQUE: in order to decide to have either a multiple server system with one buffer (IS_BUFFER_UNIQUE=1) or a multiple server system where each of them is equipped with a buffer (IS_BUFFER_UNIQUE=0).
- SERVERS_NUMBER: defining the number of servers in the system.
- BUFFER_SIZE: in order to test the system under different buffer sizes.
- SERVICE_TYPE_DISTRIBUTION: determining the service time distribution, which in this case can be *exp, uniform* or *tcp*.

For each buffer size, we observed the variation of the analysed metrics under seventeen different arrival rate values and, for each arrival rate, the metric value was averaged around 20 runs.

We considered *ms* as unit of measure for time as it is relative to what happens during the simulation. The main situations analysed in order to evaluate the system operation and performance were:

1. a single drone with a single antenna and no buffer;
2. a drone equipped with two antennas and a finite buffer;
3. the comparison between a single drone with two antennas and two drones equipped with a single antenna;
4. variation of the distribution of the service time, considering a single drone.

1.2.1 Case 1: single drone with a single antenna and no buffer.

In the first case, the drone is sent to provide additional capacity during peak time. The system performance was investigated under different arrival rates, keeping a fixed value for the average service rate. The parameters set in order to simulate the system were: IS_BUFFER_UNIQUE=1, SERVERS_NUMBER=1 and BUFFER_SIZE=0.

1.2.2 Case 2: single drone with two antennas and a finite buffer.

In the second case, in order to perform the system simulation, the parameters were set in this way: IS_BUFFER_UNIQUE=1, SERVERS_NUMBER=2 and several values for the BUFFER_SIZE.

1.2.3 Case 3: comparison between a single drone with two antennas and two drones equipped with a single antenna.

In the third case, the parameters setting for the single drone case was: IS_BUFFER_UNIQUE=1 and SERVERS_NUMBER=2, while regarding the two drones case: IS_BUFFER_UNIQUE=0. The packets arriving are assigned to either one drone or the other according to a random choice.
In both cases, the simulation was performed assuming an infinite buffer size, hence by setting the BUFFER_SIZE parameter to a significantly large value.

1.2.4 Case 4: variation of the distribution of the service time.

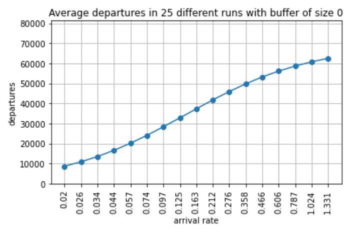
In the last case, we simulated a single drone system trying to vary the distribution of the service time, considering the case of the M/G/1, assuming two different *uniform* distributions with the same mean but different variances and a *tcp* distribution instead of the exponential, observing how the system performance changes. An approximate *tcp* distribution was created using a packet size obtained through two different Gaussian-like distributions centered on 40 bytes and 1460 bytes, each extracted with probability 49% and a uniform distribution between 41 bytes and 1459 bytes with probability 2%. The parameter set in order to evaluate the two different situations is SERVICE_TYPE_DISTRIBUTION.

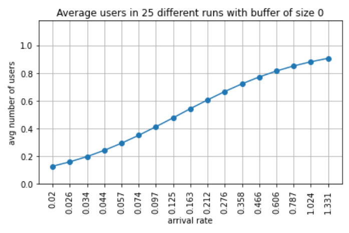
1.3 Results discussion

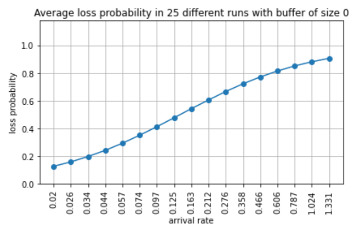
For all the analyzed cases, we plotted: arrivals and departures, losses number and loss probability, average number of users, average time spent in the system and only in the buffer (averaged only on users that were actually put in the buffer before being served) and the average busy time of each server. Not all graphs will be reported, but only the ones we considered useful to be discussed.

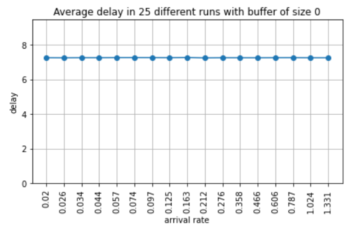
1.3.1 Case 1: single drone, single antenna and no buffer

In fig. 1 it is possible to observe the variation of the relevant performance metrics analysed in the first case scenario. Regarding the number of transmitted packets, it can be seen how it gradually increases as we approach higher arrival rate values, starting from around 10000 packets with an arrival rate of 0.02 packets/ms, and reaching more than 60000 packets with an arrival rate of 1.331 packets/ms (Fig. 1a). As the arrival rate tends to increase, also the average number of users (packets) in the system grows (Fig. 1b), with the maximum mean reachable value equal to one, as the simulation was performed with a single drone featuring a single antenna (server) and no buffer. Also the loss probability and the server busy time follow the same trend (Fig. 1c and Fig. 1e), while the waiting time is always equal to 0 ms as there is no buffer (in Fig. 1d the delay represents the average service time).
Another important step is the analysis of the confidence intervals for one of the metrics just reported: the loss probability. A confidence level of 90% was set, hence α is equal to 0.1 and, with 25 runs, the value of the variable t extrapolated from the *t-student* distribution table is 1.711. For each arrival rate, the mean loss probability over the 25 runs was computed as well as the related confidence interval. As it can be observed in fig. 2a and fig. 2b, by changing the simulation time, the confidence interval tends to significantly reduce, allowing to have a better estimation for the loss probability value, as the number of performed measurement is higher. Overall, the obtained results are consistent with theoretical expected values.


(a) Number of transmitted packets (departures).


(b) Average number of users (packets) in the system.


(c) Loss probability.


(d) Delay.

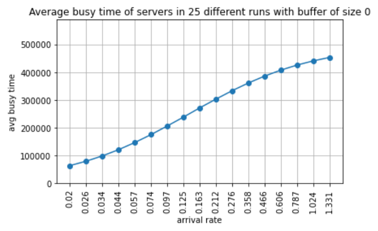
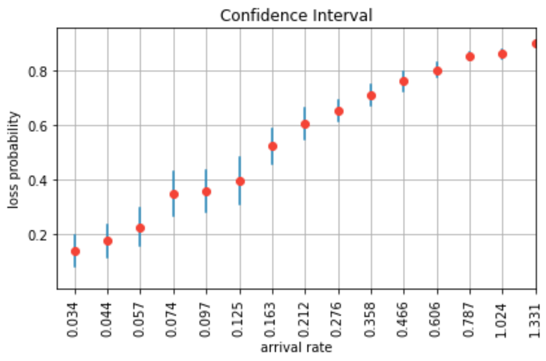

(e) Server busy time.

Figure 1: *Variation of relevant performance metrics in case 1.*


(a) Short simulation.

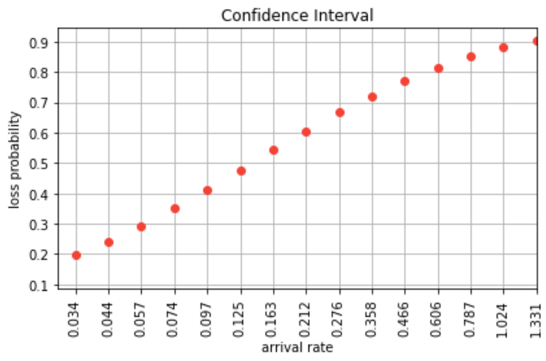
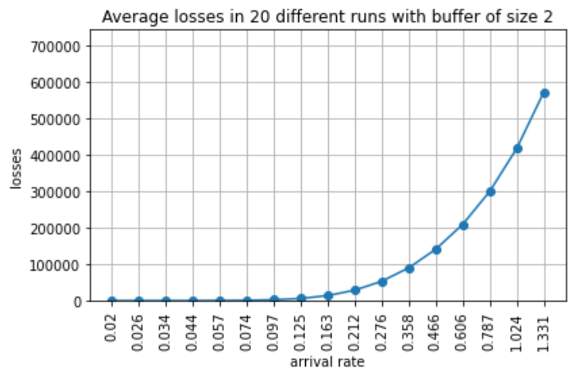

(b) Long simulation.

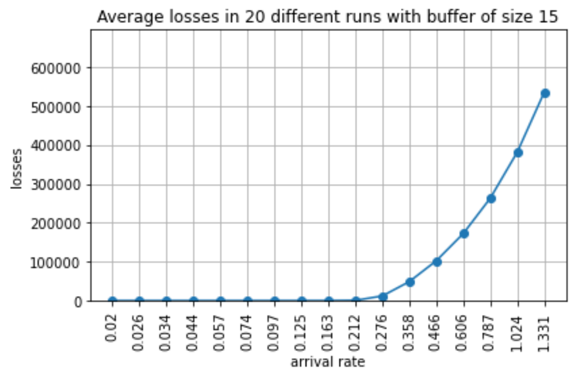
Figure 2: *Loss probability confidence intervals in case 1.*

1.3.2 Case 2: single drone, two antennas and finite buffer

Let us consider losses and loss probability. Observing the plots in fig. 3 and in fig. 4, we can observe that the losses trend has an exponential growth in the analyzed range of arrival rates, proportional to the number of arriving packets that increases exponentially and due to the fact the the buffer easily arrives to saturation.
Considering the case with buffer size=15, we can observe different statistics in fig. 5. In particular, it is clear how the system behaves with arrival rates 0.276 packets/ms and 0.358 packets/μs: in this situation servers saturate and start being always busy (fig. 5a). As a consequence, the departure rate (fig. 5c) saturates also when the load continues to grow, in fact the losses curve (fig. 3a) starts its exponential behavior and the related probability (fig. 4b) has an important growth.
From a QoS point of view, it can be interesting to observe the shape of the average delay in queue fig. 5b) in the immediately previous interval (from arrival rates of 0.163 packets/ms to 0.276 packets/ms): the busy time has a strong increase and this causes a relevant growth of the average delay in the buffer (reaching almost 20ms); this observation is probably the one with the biggest impact for the end user to correctly model and project the system.

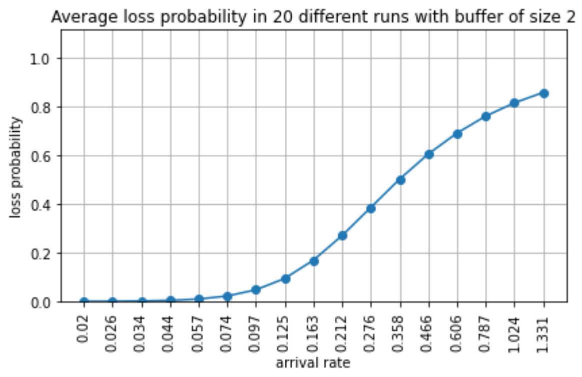


(a) Buffer of size 2.

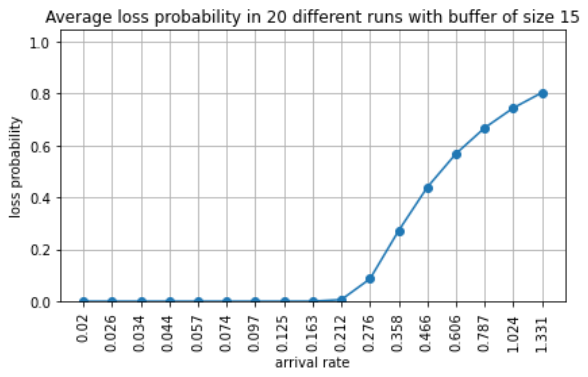


(b) Buffer of size 15.

Figure 3: *Losses over arrival rates in case 2.*



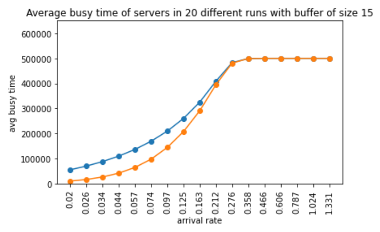
(a) Buffer of size 2.



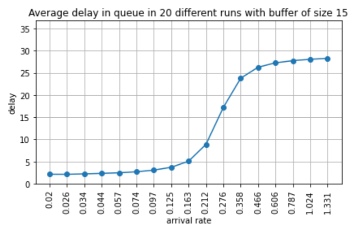
(b) Buffer of size 15.

Figure 4: *Loss probability over arrival rates in case 2.*

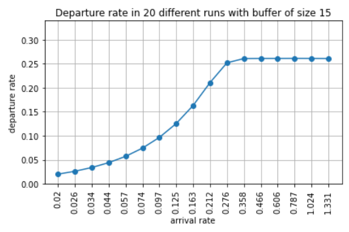
The parameter that varies this behavior is the buffer size, that in practice moves the curve to higher arrival rates. This phenomenon can be studied to choose the correct buffer size as it could be preferable to have slightly more losses with respect to an increase on the waiting time in the queue experienced by users.
The trend of loss probability gives almost the same information, but has a different graphical shape as losses are normalized on the total arriving packets.



(a) Average servers busy time.



(b) Average delay in queue.

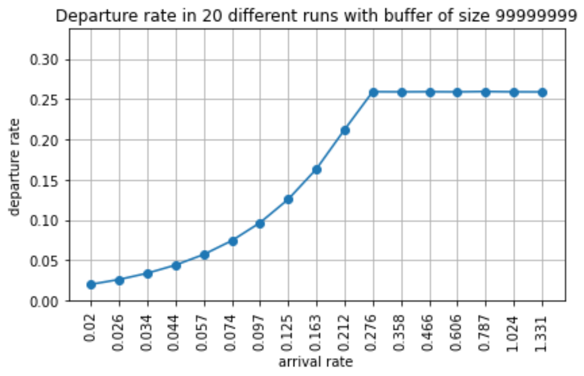


(c) Departure rate.

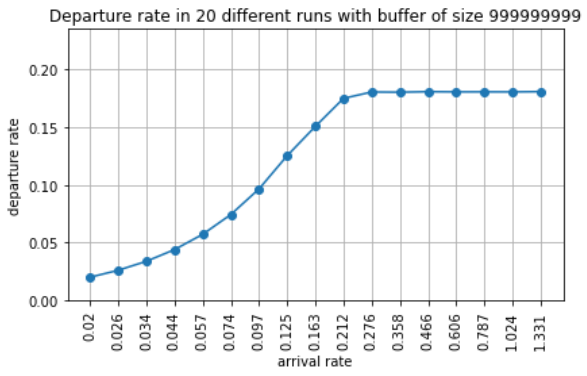
Figure 5: *Statistics with buffer size=15 in case 2.*

1.3.3 Case 3: infinite buffer

In this case we compared a single drone equipped with two antennas (case 3a) with a system with two drones, each equipped with a single antenna (case 3b). Assuming an infinite buffer size, regarding the departures, it can be seen that in the case 3a the departure rate (fig. 6a) is on average higher with respect to the case 3b (fig. 6b) as when one of the servers is busy, the following packet is served by the idle one, hence the capacity is optimally managed as there is one queue and the average time spent in the system. As a consequence, the delay (fig. 7a) is smaller than the one of the case 3b (fig. 7b) proportionally to the number of servers (it must be noted that, the larger the buffer size, the larger is the potential delay, in fact with an infinite buffer size it tends to reach very high values). Moreover, in the case 3b it could happen that while some packet is waiting in one of the two queues, the other one is empty with the server idle, hence making it to be useless and wasting resources.
It can also be observed that the departure rate saturates when the arrival rate is around 0.276 packets/ms because of the maximum service rate achievable. As a consequence, the average number of users in the system exponentially grows (fig. 8a and fig. 8b). Regarding the average busy time of the servers, it still saturates because of the arrival rate reaching a value that starts to be significant with respect to the fixed service rate. The value at which it saturates is 500000 ms, which corresponds to the simulation time, meaning that the servers are always busy from a certain point. Moreover, it can be observed how in the case 3a the second server is exploited when packets in the queue find the first one to be busy, while in the second case a packet is randomly served by one of the two servers (see fig. 9a and fig. 9b).
The main differences between considering an infinite buffer size for the drones and considering a finite one is that, in the second case, the delay saturates to a value that is proportional to the size of the buffer and the average amount of users in the system is less as there is a limited capacity, also causing losses.



(a) case 3a.

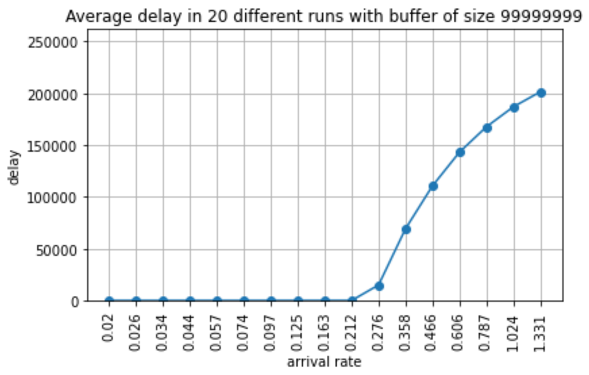


(b) case 3b.

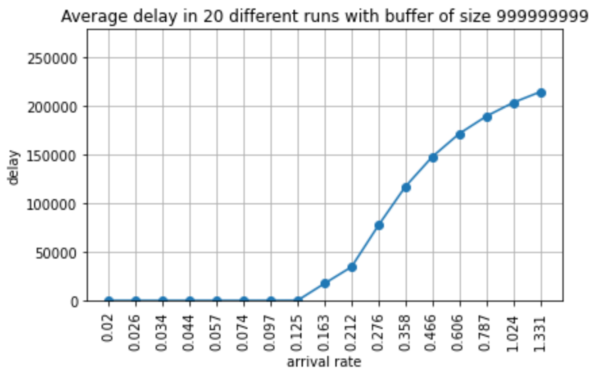
Figure 6: *Departure rate in case 3.*

1.3.4 Case 4: M/G/1

Considering a single drone, we tried to vary the distribution of the service time, modelling an M/G/1 case (so with one server) to compare the obtained results with the ones using a 'classical' exponential distribution. In the simulations obtained not using a buffer, we did not appreciate differences. In the cases with buffers of difference sizes the trend is almost always the same: the distributions with higher variance (one of the two uniform that was implemented between 0 and 2*mean service time) and the *tcp* distribution show slightly worse performances. In particular, as confirmed by the Pollaczek-Khintchin formula, with higher variance, we obtained higher delays in the buffer and higher losses. This behavior was observable using arrival rates that do not show a 'static behavior' as in the cases with load close to zero or with high load (close to one), for which the metrics tended to saturate to the same values for all distributions. In tables 1 and 2 some of these important results are reported.

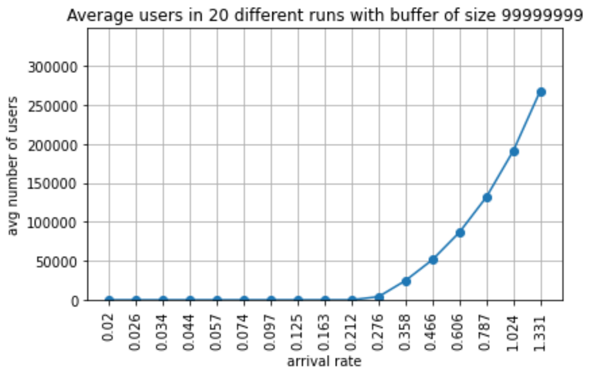


(a) case 3a.

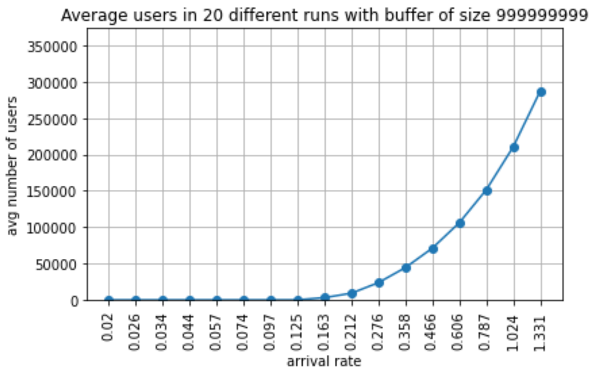


(b) case 3b.

Figure 7: *Delay in case 3.*

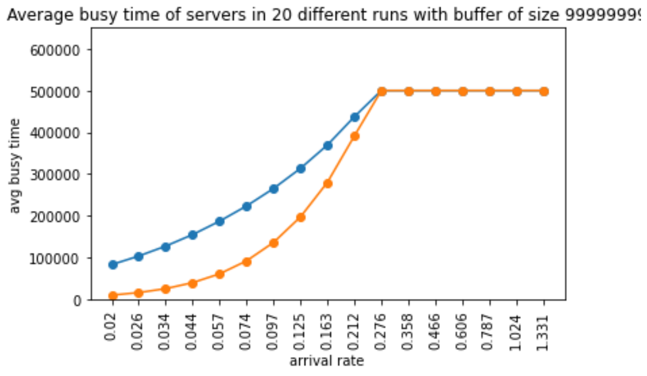


(a) case 3a.

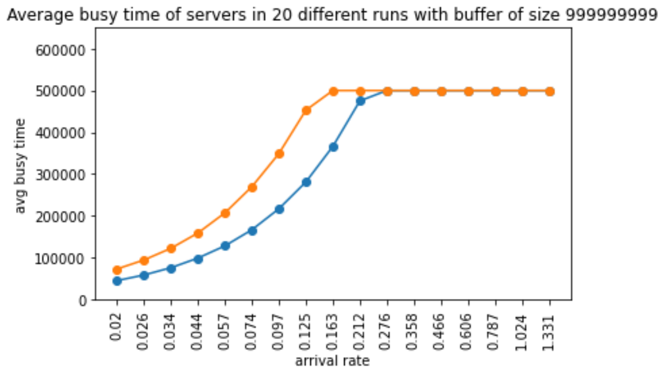


(b) case 3b.

Figure 8: *Average number of users in the system in case 3.*



(a) case 3a.



(b) case 3b.

Figure 9: *Average busy time of servers in case 3.*

Distribution	Arrival rate [packets/ms]	Delay in queue [ms]
Low variance uniform	0.07	13.9
	0.09	27.9
High variance uniform	0.07	14.7
	0.09	28.3
Tcp	0.07	15.8
	0.09	30.1

Table 1: Comparison of delay in the queue among the three distributions with buffer = 50.

Distribution	Arrival rate [packets/ms]	Total losses [packets]
Low variance uniform	0.045	697
	0.075	4600
High variance uniform	0.045	818
	0.075	5026
Tcp	0.045	1025
	0.075	5688

Table 2: Comparison of losses among the three distributions with buffer = 2.

2 LAB 2

2.1 Methods and parameters setting

The general approach is based on the `Server` Class, used both for the Base Station and for the drones. The main differences among them is that the BS cannot be shut down and has priority in the usage: the packets are firstly tried to be assigned to it, only if needed drones are used. To simulate a realistic scenario we assigned to the BS a buffer of 100, 2 antennas and a service rate of 2μ.
We used a fixed seed to perform more meaningful comparisons.
Differently from lab 1, we considered μs as unit of measure for time, but in any case the time unit has a relative meaning to what happens during the simulation and should not be intended in its absolute meaning. The analyzed time is slotted to have statistics related to each slot to implement the UAV scheduling strategies. As it will be explicated, the dimension of these time slots is customizable. The code has been thought not only to respond to the requested task, but also to be modular with respect to many parameters usable to personalize the simulation in an easy way. Some of them are:

- `SERVERS_TYPES`: it is a list used to specify the type of servers, that is used in the Server Class to characterize some properties such as the battery duration, the service rate, the buffer size;
- `ANTENNAS_NUMBER`: it is a list that includes the number of antennas for each server;
- `BUFFER_SIZE`: to choose the size of drones buffer and `BSbuffer` for the BS one;
- `HOURS_NUMBER`, `FIRST_HOUR`: to choose the time window to be analyzed and others to define the increasing/decreasing factor of the load during them;
- `STATISTICS_FREQUENCY`: to set the number of slots per each hour;
- `SCHED_STRAT`: to select the scheduling strategy to be adopted;
- `REAL_TIME`: can be 1 to indicate the reactivity of drones slot-by-slot (they can be deactivated if not needed anymore) or 0 to consider that a drone, after being activate, remains in usage till the battery depletion.

Scheduling is strongly related to time slotting: in each time slot a single drone can be activated or deactivated, and strategies are based on previous slots metrics. To modify the reactivity of the system it is sufficient to increment the number of time slots per each hour in a very simple way. We implemented 3 scheduling strategies, each of them customizable through some parameters:

1. The first strategy is related to the **losses** and is based on 2 parameters (`LOSS_THRESH` and `STRAT_ONE_PERCENTAGE`, that is actually a ratio not normalized over 100). The idea is to check losses in the 2 previous time slots: assuming to be in the slot *i*, if losses in slot *i-2* are greater than the chosen threshold and the ones in slot *i-1* are greater than the chosen threshold multiplied by another chosen percentage (in most of our simulations 20%) a drone is activated. The choice of observing two consecutive slots is based on the observation that in some scenarios there are some 'instantaneous' peaks, not representative of the wider time period, that could cause the activation of a drone without a real need. The specified percentage is useful to avoid this behaviour with risking to not send a drone when actually needed (i.e. losses in slot *i-2*=10 and *i-1*=5). This can actually also be used to consider increasing losses scenarios as this multiplying factor can be chosen to be greater than 1;

2. The second strategy is related to the **arrival rate**: if it is over a threshold a server is activated, if it is under it, it is deactivated. We chose no to perform the check on two slots as in previous strategy to have a more reactive approach, also considering that the arrival rate changes in a slower manner and less with respect to the losses with a realtive smaller variance;

3. The third strategy is more conservative as it is based on the **buffer occupance** of the *n-1*-th server (the one before the next drone to be activated), as the usage of servers is performed in order starting from the BS going on with the following drones. The check is done on the percentage of the buffer occupance to avoid to overload the drone, leading to a higher delay and evetually losses.

2.2 Results discussion

In this section, the main results obtained by performing the different tasks will be discussed. Our group was selected in order to find scheduling solutions intended to help managing the traffic in a Residential area. For all the analyzed cases, we plotted: arrivals, losses, loss probability, number of transmitted packets (departures), drones activation with respect to the time slots, average delay experienced by each packet (for QoS considerations), users processed by the drones with respect to the time slots, drones' average buffer usage, drones' antennas usage and total amount of clients served by each drone. Not all of the graphs will be reported, but only the ones we considered useful to be discussed.

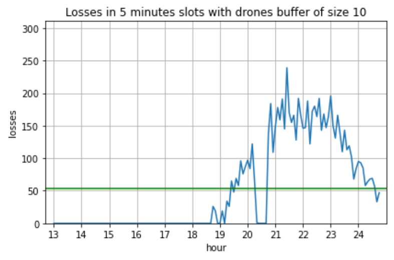
2.2.1 Task 1

For the first task we implemented different scheduling strategies in order to offload some mobile traffic managed by a BS in a certain Residential area. For hosting the BS we considered one drone (with one antenna) equipped with a battery unit as power supply: autonomy and time required to fully recharge it (once it has become empty) were kept into account.
The strategies used for sending the drone are the ones described in the previous section. Regarding the buffer size of the drone sent, we tried with four different values: 0, 2, 10, 30. For the strategy thresholds set in order to send the drone, we made the comparison considering the results related to the best parameters setting for each strategy, optimizing each single strategy.
First of all, the most relevant comparison to do among the different strategies is the total number of packets lost with respect to the total arrivals, hence understanding what is the best strategy that allowed to transmit the higher amount of packets. Considering, for the sent drone, a buffer size equal to 10, over a total of around 25800 packets arrived, we obtained the values of tab. 3.
In fig. 10 it is possible to observe the plots related to these losses and delays. The significant decreases in the losses are related to the time slots in which the drone is sent according to the specific strategy, in which

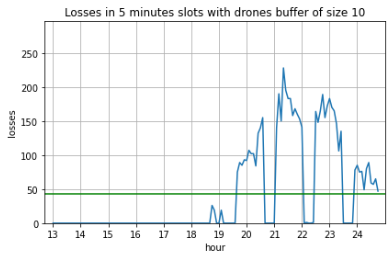
	losses [packets]	**average delay [μs]**
strategy 1	7999	1558
strategy 2	6131	1399
strategy 3	6057	1400

Table 3: Results obtained with drone buffer size = 10.

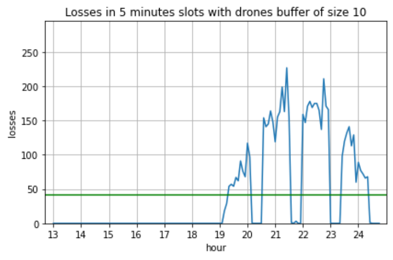
the most performant ones resulted to be the one related to the **buffer** (strategy 3, see fig. 10c) and the one related to the **arrival rate** (strategy 2, see fig. 10b).
It is evident how with these strategies it is possible to better exploit the drone's service, taking into consideration the capacity of the battery (equal to 25 mins) and the needed charging time (equal to 60 mins). Fig. 11 shows the amount of users processed, in which the blue line represents the base station, while the orange line the sent drone. Fig. 11b and 11c confirm the results obtained in fig. 10. The **losses** strategy is the worst one as for sending the drone it is necessary to previously sense a certain amount of losses in the base station, if this value is higher than a fixed threshold (in our case equal to around 110 losses) than the drone is sent. On the other hand, by using the **buffer** strategy we prevent the losses in the base station by sending the drone when its buffer is occupied for more then the 75% of the actual size. In the same way, by following the **arrival rate** strategy, it is not necessary to wait for losses, hence it also performs better.



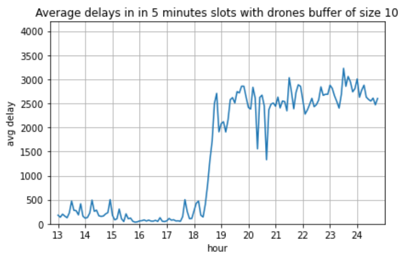
(a) Strategy 1 - losses.



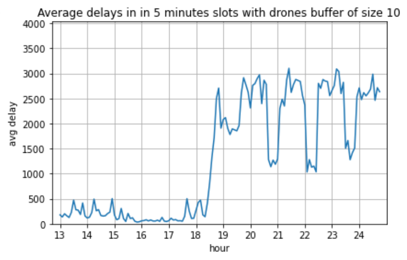
(b) Strategy 2 - losses.



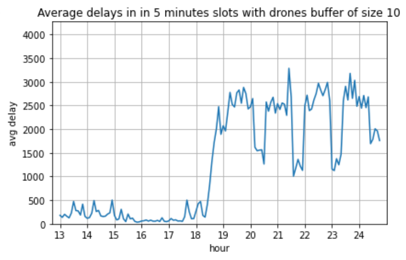
(c) Strategy 3 - losses.



(d) Strategy 1 - delays.



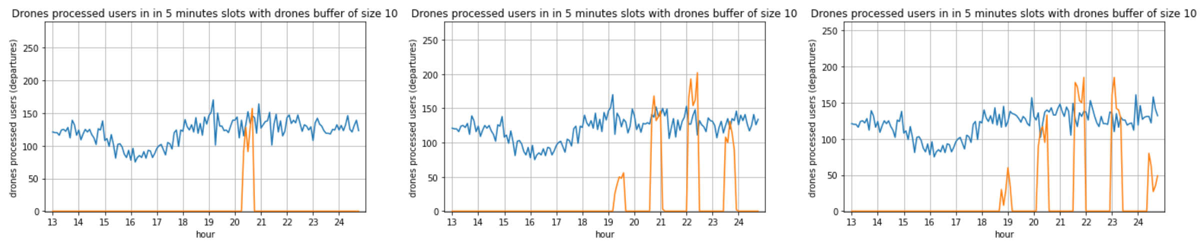
(e) Strategy 2 - delays.



(f) Strategy 3 - delays.

Figure 10: *Losses and delay with respect to the time slots for each strategy.*

Moreover, the losses distribution fluctuates more with respect to the arrival rate one, in fact with the strategy related to the second mentioned, the drone manages to cover more uniformly the congested time slots. Trying to increase the buffer size, the system showed less losses until a certain a point, then the number started increasing as, when a drone is turned off, the clients in the queue are lost. To avoid this drawback it should be useful to develop a smart strategy to make drones not loss those packets, for example avoiding to be available for receiving packets in the last instants. In addition, an increase in the buffer size obviously showed an increase in the average delay.

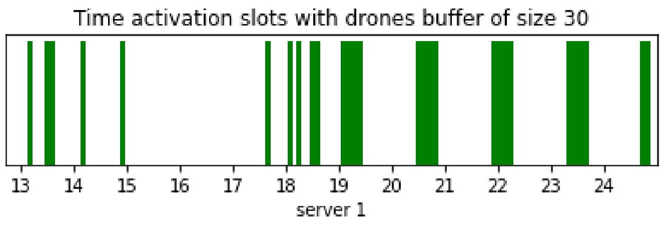


(a) Losses. (b) Arrival rate. (c) Buffer.

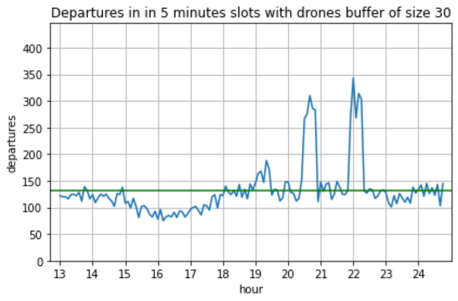
Figure 11: *Processed users with respect to the time slots (blue=base station, orange=sent drone).*

2.2.2 Task 2

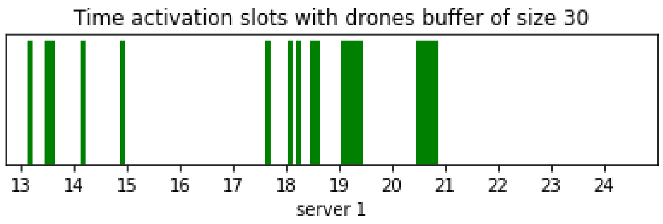
The second task consisted in introducing a constraint on the maximum number of charging/discharging battery cycles the UAV can undergo during a day, in order to limit the battery degradation.
For this task, the comparison will be made with a buffer size equal to 30.
The strategy with which the analysis was performed is the **buffer** one (strategy 3), as it resulted to be the most performant.
Regarding the strategies implemented in the previous task, the optimal setting of the thresholds made the sent drone to perform its support to the Base Station being able to perform a maximum of 4 charging/decharging cycles, the optimal ones (sending the drone more frequently does not allow the drone to cover the most critical time slots).
Knowing this, we decided to try to decrease the thresholds in order to trigger the sending of the drone more frequently, but in time slots that do not necessary are the optimal ones. Considering in particular the **buffer** strategy, we decreased the buffer threshold (defining the percentage of the drone's buffer that must be filled in order to trigger the sending of the drone) to 0.1 and set the maximum number of charging/decharging cycles to 8. In fig. 12a and fig. 12b it is possible to observe the time slots in which the drone activates and the overall departures (from base station and drone). The total amount of losses was equal to 6788 packets. Secondly, we tried to decrease the number of charging/decharging cycles to 4, by maintaining the same threshold value for the buffer (0.1). As expected, the number of losses increased significantly to 7430 packets as the drone is able to cover a less amount of time slots, hence having less departures with respect to the previous case (fig. 12c and fig. 12d). In fact, it is important to highlight that when the number of charging/decharging cycles is reduced, it is better to set a threshold value that allows the drone to support the base station only in most critical situations. With four maximum charging/decharging cycles, by increasing the buffer threshold to 0.75 (which is the optimal setting defined in task 1), the drone exploits in a better way the time slots, reducing the amount of losses.
Also considering the QoS, the 0.75 threshold resulted to be better. As it can be observed in fig. 13, in fact, it allows smaller delays (average of 1416 μs versus 1508 μs obtained using the threshold 0.10 with the max recharging cycles value equal to 4). Obviously, this behavior causes an higher variability in the average delay. Hence, again, the choice between smaller average delay or smaller variance of it, also considering the losses number, has to be performed.
For what concerns the remaining scheduling strategies, by following the same rational, the results were more or less the same.



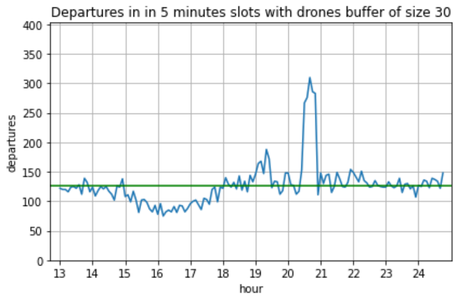
(a) Time slots when drone is active (max recharging/decharging cycles=8, buffer threshold=0.1).



(b) Departures (max recharging/decharging cycles=8, buffer threshold=0.1).

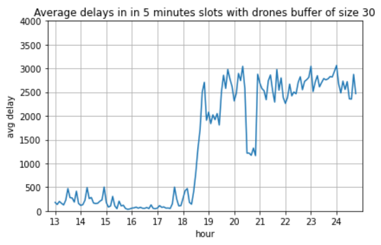


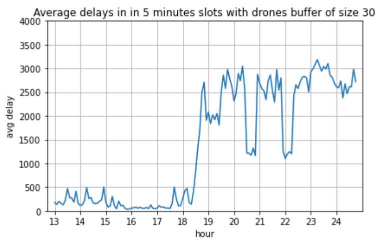
(c) Time slots when drone is active (max recharging/decharging cycles=4, buffer threshold=0.1).

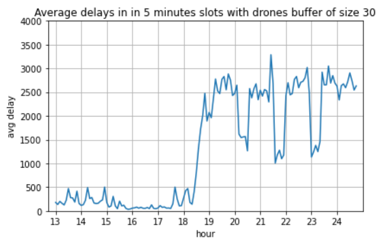


(d) Departures (max recharging/decharging cycles=4, buffer threshold=0.1).

Figure 12: *Drone's activation time slots and overall departures.*







(a) Threshold=0.1, max cycles=4. (b) Threshold=0.1, max cycles=8. (c) Threshold=0.75, max cycles=4.

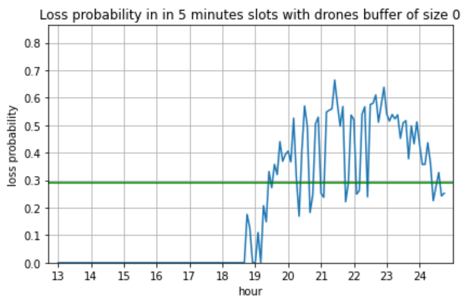
Figure 13: *Average delay with different threshold for strategy 3 and different maximum recharging value.*

2.2.3 Task 3

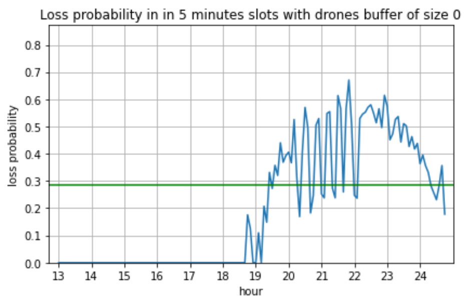
For this task, we tried to implement the usage of drones with solar panels during the sun hours but this did not lead to any change as the arrival rate increases over significant thresholds during late afternoon. In any case, to make an analysis based on different capabilities of UAV in sense of autonomy, we considered four types of drones:

- type A: 25 minutes of autonomy;
- type B: 30 minutes of autonomy;
- type C: 35 minutes of autonomy;
- type D: 40 minutes of autonomy.

Scheduling strategy 1 (losses).
Considering the scheduling strategy related to losses, it is observable that an increase in the availability of the drones leads to a lower loss probability.



(a) Type A UAV.



(b) Type D UAV.

Figure 14: *Loss probability using different types of drones.*

As observable in fig. 14, the longer availability of the drone allows to reduce the spreading of the time periods related to higher loss probability (as, for example, immediately after 21), allowing the drone to be resent for more slots before recharging.

Scheduling strategy 2 (arrivals).
Also in this case we observed an overall improvement, for example the type A drone allows to have in average almost 1000 less packet losses per day. Moreover, through a grid search, we observed that that the optimal parameter for the threshold used to activate the drone slightly changes using drones equipped with different autonomies, as observable in table 4. This is related to the fact that, to optimize the scheduling, the value of the threshold resulted such that the drone activation is more or less centered around the major peaks of arrivals and the possibility to increase the drone usage time makes the active time slots larger and so centered on bigger or more numerous possible slots. In this table, it is also explicitated that the the optimal thresholds resulted to be the same both minimizing the losses and the delay, being both parameters strongly related to the buffer usage.

Drone type	**Best threshold w.r.t. losses**	**Best threshold w.r.t. delay**
Type A	0.050	0.050
Type B	0.052	0.052
Type C	0.053	0.053
Type D	0.051	0.051

Table 4: Best threshold for scheduling strategy 2.

Scheduling strategy 3 (buffer).
In this case we found an interesting behavior: different types of drones are related to different optimal buffer sizes if considering the ones ensuring the less total amount of losses during the day.
These values, as previously discussed, cannot be taken as absolute choice-driver parameters as the delay must be considered for QoS provisioning, but also in this case (as shown in tab. 5) the best buffer size values resulted the same considering both metrics. Moreover, this analysis confirms that it is useful to have a buffer but increasing too much its dimension can worsen the performance.

Drone type	**Best buffer size w.r.t. losses**	**Best buffer size w.r.t. delay**
Type A	10	10
Type B	2	2
Type C	2	2
Type D	10	10

Table 5: Best buffer size for scheduling strategy 3.

2.2.4 Task 4

The fourth task consisted in considering a scenario in which there are N available drones (in our case N=2) that can be of different types. Keeping the value of N fixed, we considered different combinations regarding the type of the two drones sent and comparing their performances, reflecting on which is the configuration that allows the better trade off between the volume of traffic handled by the available drones, the number of battery charging/decharging cycles and the QoS in terms of delay.
For this task, the service time of the Base Station was increased (service time=2.5μ) so that the drones would both have been able to activate in a sufficient amount of time slots, as the probability for the first drone to need help by the second one to help the Base Station is increased.
Moreover, as in task 3, we assumed that the drone can have different battery autonomies.
The **losses** strategy (strategy 1) is the one we used in order to perform this task.

First combination
Regarding the first combination, we decided to configure two types of drones, a low performance drone and a high performance drone, with the following settings:

- low performance drone: type A battery autonomy, number of antennas=1, service rate=1/15 (μ), buffer size=10;
- high performance drone: type C battery autonomy, number of antennas=2, service rate=1/15 (μ), buffer size=10.

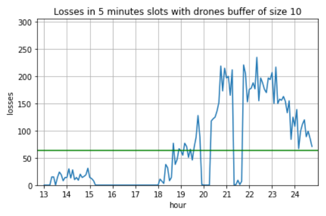
The loss threshold was set equal to 110 and the maximum number of charging/decharging cycles to 4.
Our aim was to make a comparison between a situation in which two low performance drones are sent and another one with a high performance and a low performance drone.
Regarding the first of the two configurations, we obtained the following results in terms of losses and delay:

- losses: 9251 packets;
- delay: 2399 μs.

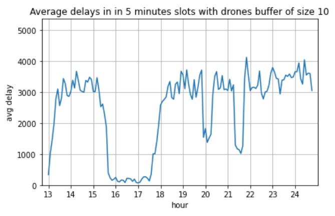
In fig. 15a and fig. 15b, it is possible to observe the graphs related to both metrics.
The significant decrease in both losses and delay from around 4pm to 6pm is due to the buffer of the Base Station that is almost empty due to the lower arrival rate. This implies that there are no losses and that the delay is significantly reduced. In fig. 15c it is possible to see the average buffer occupance of the Base Station, highlighting the discussed behaviour.
Regarding the configuration with one high performance drone and a low performance one, we obtained the following results:

- losses: 7347 packets;
- delay: 2143 μs.

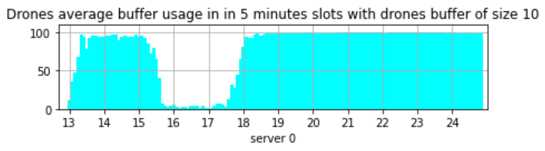
In fig. 16a and fig. 16b, it is possible to see the graphs related to the performance metrics.
By analyzing these results, it is evident that sending the more efficient drone improves the performances in terms of losses and quality of service from the point of view of the user (delay) and better supports the Base Station, but it is also true that the investments in deploying it would be higher. On the other hand, sending a low performance drone means investing less but providing a worse service to the user (as well as having more losses). An option for dealing with this problem could be to increase the maximum number of charging/decharging cycles of the low performance drones so that they can be active for a higher amount of time and improve the performance, but this would mean to exploit more the drones' batteries and ruin them with a faster pace, hence investing money in order to replace them. However, the users would experience more or less the same delay as we would send the low performance drones but only for a higher amount of time.
It is a trade-off between improving the user experience (QoS, i.e. delay), investment in terms of money and average battery usage and later replacement due to the defined maximum number of charging/decharging cycles.



(a) Losses.

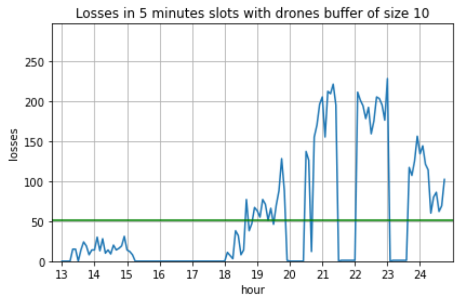


(b) Delay.

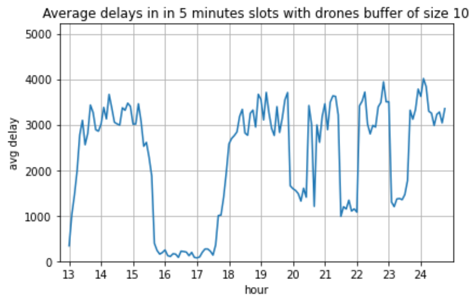


(c) Base Station buffer usage.

Figure 15: *Two low performance drones configuration.*



(a) Losses.



(b) Delay.

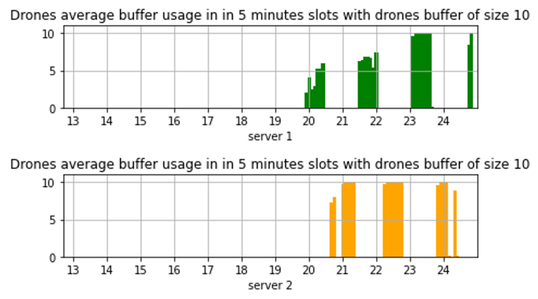
Figure 16: *One high and one low performance drones configuration.*

Second combination
The second combination was studied to analyze the difference between the usage of drones with service rate 2μ and ones with 2 antennas with service rate μ each. To perform this comparison, we considered 4 different combination:

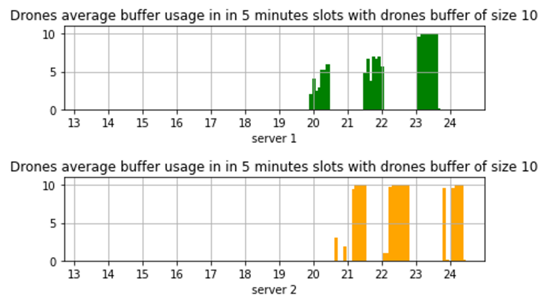
- 2μ - 2μ;
- 2μ - 2 antennas;
- 2 antennas - 2μ;

- 2 antennas - 2 antennas.

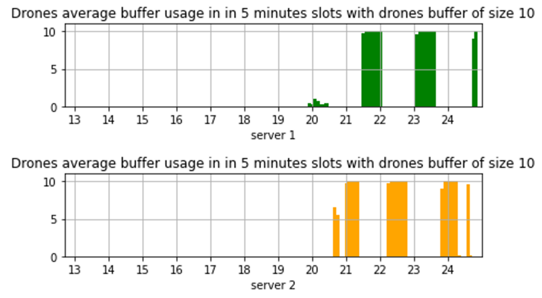
The second and third combinations must be considered separately, as our code is thought to give priority to the first server in assigning the incoming packets. The first two combinations showed almost the same result. This can be better understood by observing fig. 17a and fig. 17b. This situation can be explained analyzing the behavior inside the different systems, described as different queues.



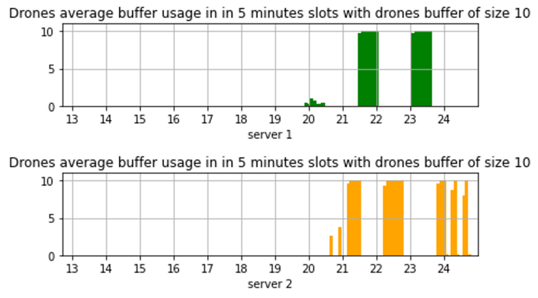
(a) 2μ - 2μ.



(b) 2μ - 2 antennas.



(c) 2 antennas - 2μ.



(d) 2 antennas - 2 antennas.

Figure 17: *Processed users with respect to the time slots (blue=base station, orange=sent drone).*

When the traffic is low (when the buffer is not completely full), the systems with 2 servers (2 antennas) with rate μ is expected to work worst than a single server system with a service rate μ: it can be understood considering the extreme case of the buffer always empty: the total delay will be only composed by the service rate and one system will have a delay equal to the half of the other. We computed the overall average delay of the analyzed 4 systems that range from $2320\mu s$ to $2480\mu s$. We considered interesting to observe the differences in two different situations: between 8pm and 10pm, when the buffers are not always full (as it can be seen in fig. 16 and between 11pm and 12am, when buffers are always almost full.
We obtained the delays shown in tab. 6. As supposed, the main differences can be appreciated in between 8pm and 10 pm, when buffers are not always full: the usage of two servers with 2 antennas shows an increase of 22.8% in terms of delay w.r.t. the case with two servers with a single antenna and service rate 2μ, while between 11pm and 12am the difference was lower than 10%. This can be a really useful parameter to be considered while choosing between different types of UAVs.

	Daily average	**8pm-10pm**	**11pm-12am**
2μ - 2μ	2332 μs	2205 μs	3454 μs
2μ - 2 antennas	2320 μs	2148 μs	3475 μs
2 antennas - 2μ	2480 μs	2745 μs	3424 μs
2 antennas - 2 antennas	2452 μs	2708 μs	3463 μs

Table 6: Delay obtained with different configurations.