IoT Project 2023-2024

Federico Trombetti, Novella Bartolini, Salvatore Pontarelli

1 Introduction

Wireless communications are essential to achieve the expected performance of future generations of mobile communication systems. Despite increased efforts in enabling mmWave regions for 5G NR communications, the spectrum is still scarce, leading to the need of increasing spectral efficiency. Given the practical implementation problems encountered with an increasing number of antennas for a centralized system and the interference problems encountered with cell splitting, the spectral efficiency of 5G systems is not sustainable. Therefore, further innovations in the architecture of cellular networks are required.

In the IoT environment, sensing plays a crucial role. By collecting data taken directly from the field, sensors are able to monitor remote and hazardous environment without the need of human personnel. Due to their nature, sensor devices are able to produce a great amount of data, which has to be delivered and processed by a central unit in order to ensure seamless operations. Sensors are, most of the time, part of a wireless network of devices, used to relay information to central units. For operations to carry out smoothly, this network must remain consistently connected and online. However, the dynamic nature of wireless networks can compromise the availability and reliability of sensing services. Continuous data offloading is necessary to prevent a permanent loss of critical sensing data.

To improve sensing service reliability, we envision an IoT network where sensors' communications are supported by a network formed by drones and balloons, providing caching and network relay services. A central processing server is connected to a base station. Several balloons act as edge servers, and provide both data caching and relay service enabling multihop communications from the sensors or drones to the balloons and then to the base stations. Balloons are mobile devices with storage, processing and locomotion capabilities. They move at low speed and are mainly deployed so as to provide radio coverage to an underlying region. In contrast, drones are employed to create an ad-hoc network with higher mobility devices which can move around the area so as to facilitate message delivery from the sensors to either the edge servers (the balloons) or the base station directly. While drones provide higher speed and more controllable trajectories, their processing and storage capabilities are more limited. In your project you are requested to design the network just described, considering one of three possible scenarios, with different network characteristics in terms of *radio coverage*, *network connectivity* and *sensing data caching*. More specifically, in scenario A) we consider the case of in which we have **static sensors** and **mobile balloons** that cannot provide complete radio coverage of the sensor network, so they should move to obtain data from all of them. In scenario B) we consider a lack of radio coverage of the **static sensors** by a set of **static balloons** in which unlike case A) connectivity is obtained by deploying **mobile drones** around the area to provide dynamic connectivity, either by letting them move as data ferries or as dynamic relay points. Finally, in scenario C) we consider **mobile sensors**, and **static balloons** with limited storage capabilities, so that they can keep track of only K pieces of data obtained by the sensors. Dealing with limited data storage, and having to face demands coming from the base station, requires smart caching capabilities from the balloons, i.e. capability to decide what to cache, and which piece of data to evict based on occurring hits and misses in their interrogation.

2 Project Specifications

In this project, you are requested to implement your first full ROS/Gazebo simulation, modelling the aforementioned problem. Your project will consist of multiple ROS nodes which will be used to model and impart instructions to the devices in the simulation.
In this section, we will give a brief overview of all the devices composing your complex IoT network along with some suggestions on the topics you will need in order to control them.

2.1 Base Station

Represents the main base station where all data needs to be offloaded to enable processing at a central server or cloud environment.
The base station device should subscribe to a topic to listen to data received from other devices in the simulation, i.e., drones or balloons.
The main logic of the base station should consist in a program which looks for sensing data and keeps track of missing entries. Sensing data is obtained by means of a polling activity by means of which the base station requires measurements taken by the sensors deployed over the field of interest. Since the base station is not directly connected to each and every sensor, the base station utilizes the balloons as proxies to obtain data made available by the sensors (directly or via relay drones). The polling activity is carried out at given rates, possibly different, depending on the involved sensor, or in an event based manner, depending on the application.

Topic	Type	Description
`rx_data`	**Subscriber**	A topic dedicated to read incoming data from the UAVs or from the balloons. A callback function should be established to read incoming data and update demand requests depending on the data received.
`odometry`	**Subscriber**	Gazebo inherited topic for *OdometryPublisher*. Used to store the position of the Base Station. Position can also be set to fixed for the base station, but has to be planned accordingly when generating a simulation instance.

2.2 Sensor Device

Represents a sensor in the simulation which produces data. The sensor should output data periodically, or randomly at a given rate or based on occurring events, and publish it to a dedicated topic. Data that is not offloaded to drones or balloons is either deleted, or stored locally. Such behaviour depends on the choice of the case study of your project, and can work either way. You are free to choose what kind of sensor you want to implement in your simulation, as long as the choice is consistent through the project and well documented.

A full simulation of a sensor device is not requested. The behaviour of the sensing data can be abstracted as a sequence of variables S. We call $s_d(t)$ the measurement taken by the sensing device d at time t. We assume that any measurement taken by a sensor has a timestamp t. Assume that each measurement has a limited validity, due to aging of the measured information. Therefore a measurement expires after Δ_t seconds.

Topic	Type	Description
`tx_data`	**Publisher**	A topic dedicated to publish generated data from the sensor. A sensor should publish data at a pre-defined rate.
`odometry`	**Subscriber**	Gazebo inherited topic for *OdometryPublisher*. Used to store the position of the Base Station. Position can also be set to fixed for the sensor, but has to be planned accordingly when generating a simulation instance.

2.3 Balloon

A balloon is a device which is able to move inside the simulation to provide radio coverage and work on the sensing data locally. A balloon should subscribe to a topic to receive data from a sensor, and publish the data it forwards to another topic. A balloon which is interested in the data measured by a sensor, may be out of radio proximity and require a drone to facilitate the related message delivery, possibly in a multi-hop manner. Balloons also offload their data to the base station. Balloons have the ability to store data locally in case offloading is not possible. The storage capability of the balloons can be considered infinite for the purpose of case study A and B.
A single ROS node can take the role of the controller and coordinate all the balloons together.

Topic	Type	Description
`rx_data`	**Subscriber**	A topic dedicated to receive generated data from sensors.
`tx_data`	**Publisher**	A topic dedicated to publish stored data. The balloon publishes to this topic to communicate with the base station.
`cmd_vel`	**Publisher**	Gazebo inherited topic for moving the Balloon. Balloons can be treated as quadcopter with invisible propellers for simplicity.
`odometry`	**Subscriber**	Gazebo inherited topic for *OdometryPublisher*. Used to store the position of the Balloon.

2.4 Simulation Manager

This is an abstract ROS node which has to be implemented in order for your simulation environment to work correctly. By default, devices in the simulation are not linked with each other and just publish and read on their ad-hoc topics. The manager should link all those topics together and pass messages between topics when the devices are in range.

Topic	Type	Description
`/UAV_[x]/tx_data`	**Subscriber**	Reads data transmitted from **UAVs** and **Sensors** on the field. One for each UAV.
`/UAV_[x]/rx_data`	**Publisher**	Writes the data that the **UAVs** and the **Base Station** receive. One for each UAV.

Note that the Publisher/Subscriber method is inverted for the manager, as it will be the one actually taking data from **`tx`** topics, and sending it to **`rx`** topics.

2.5 Drone

Differently from balloons, they have limited storage capabilities, but are more agile than balloons, namely their trajectories can be finely designed and their speed is considerably higher. They move to keep the network of sensors connected with the base station or the balloons most of the time, and act as relays to transfer data when the balloons' network does not cover the entire set of sensors on the ground. Note that this type of device is required only to carry out case study B, but can be used also for case study A, if desired.
The topics a drone uses should be the same of the ones of a balloon.

Topic	Type	Description
`rx_data`	**Subscriber**	A topic dedicated to receive generated data from sensors.
`tx_data`	**Publisher**	A topic dedicated to publish stored data. A drone should offload data as soon as it in range of a balloon or the base station.
`cmd_vel`	**Publisher**	Gazebo inherited topic for moving the Drone.
`odometry`	**Subscriber**	Gazebo inherited topic for *OdometryPublisher*. Used to store the position of the Drone.

3 Simulation Instance

Your solution should be evaluated under multiple instances. You should be able to parameterize the way a simulation instance is generated, and randomize on top of that.
For example, assuming that in your scenario you use 20 sensors in a square field of 100m x 100m, you can run simulations by randomizing the positions of the 20 sensors over the field. A single instance should spawn a predefined number of sensors and UAVs, and a single base station. Their position should be randomized following the specification of the case study you want to tackle.
Case studies are introduced in the following section.

4 Case studies

We want to study the problem of delivering sensing data to the base station (hereafter shortly defined as offloading) under different constraints. The choice of this constraint is what will differentiate the solution you propose.
You project should cover at least one of the following cases.

A: Offloading with Coverage Constraint

In this scenario you can assume unlimited storage capability for the **moving** balloons and guaranteed connectivity between the balloons and the base station.

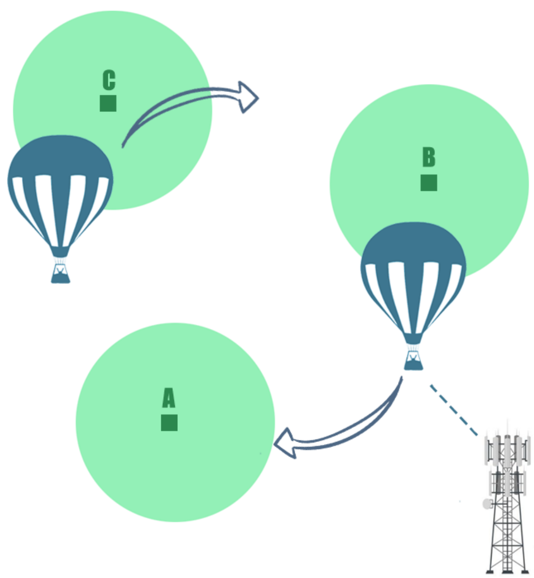

Figure 1: Scenario A

However, due to the limited power available, the sensors' communication range is rather limited and the balloons cannot cover the whole set of sensor devices on the field from a unique position. Your solution should account for this and move balloons accordingly to store data from the sensors when needed, by moving through multiple sensor devices with trajectories that allow them to collectively obtain data from the entire sensor network, and then relay it to the base station.

Optimizations

- Consider the effect of the caching policy on the balloon behavior. Allow smart trajectory planning for the balloons to gather data from the only sensors for which they do not have any data which was collected recently enough (within a time window Δ_t of aging).
- Drones can be used to have a better coverage, but they can only act as a relay between other devices.

B: Offloading with Distance Constraint

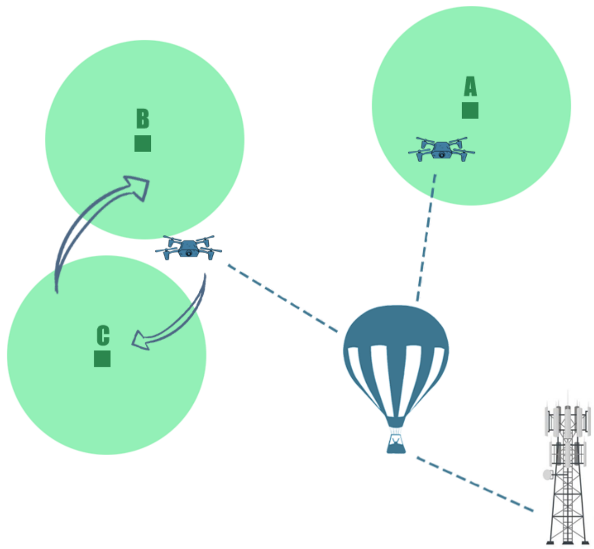

Figure 2: Scenario B

In this last scenario, there is a number of almost **static** balloons deployed in key points of the area, from which they can easily communicate with the base station. The sensors' radio range is not sufficient for all of them to reach the closest balloon. Therefore the balloon network cannot guarantee coverage of the entire set of sensors. To cope with the lack of connectivity (permanent or intermittent), we deploy a set of drones that act as data ferries facilitating data delivery from the sensors to the balloons. Upon receipt of new data from either a drone or a sensor, the balloon sends related messages to the base stations.

Optimizations

- Consider the two cases in which the balloon network cannot cover the sensors either because the sensors are too far or because they can move and get disconnected from the balloon network. Design strategies for the drones to ensure minimum loss of data facilitating message delivery through a smart trajectory design (random or TSP or others).

C: Offloading with Storage Constraint

For this scenario you can assume that the balloons are always connected to the base station. The balloons are deployed so that they can collectively guarantee radio coverage to the entire set of **mobile sensors**. However, the balloons have limited storage capability that they use to provide a proxy caching service to the base station. In this scenario, the base station makes application specific requests on the data produced by the sensors. Balloons have a finite amount of memory storage and cannot store all the data the sensors produce. The balloons should smartly select which part of the sensing data to store in order to meet the requests of the base station. You are required to implement a cache replacement policy (at least the FIFO). Remember to always

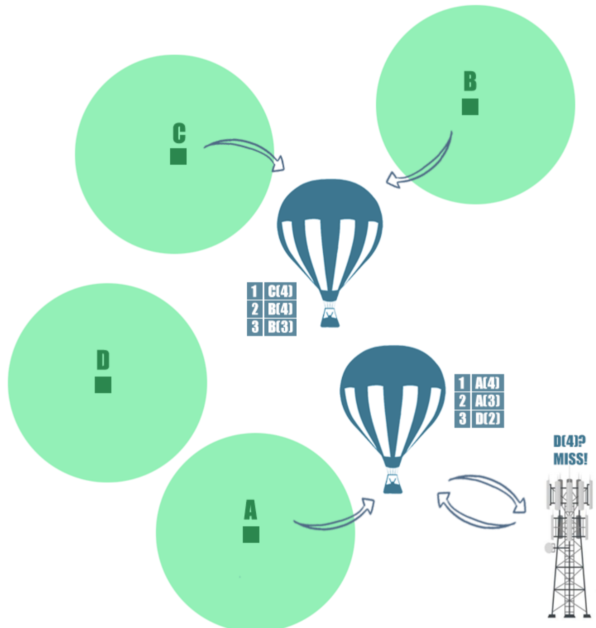

Figure 3: Scenario C

evict expired cache entries items older than Δ_t. Notice that due to sensors' mobility, anytime the base station requests data related to a sensor, the request is sent to all the balloons, and generates a cache miss for the balloons' caches that do not have the required data. To interrupt unnecessary misses' management (for the balloons that do not cover the required sensor), once the base station obtains the required information, it sends a notification message that notifies the other balloons and cancels the previous request.

Optimizations

- Consider the effect of the caching policy on the balloon behavior by considering a base station that randomly polls the sensors through the balloons acting as caches. Each sensor is polled according to a different time distribution of requests (e.g. Poissonian interarrivals at different rates). Consider sensors also sending their data at different rates. Let the balloons adopt a smart cache replacement policy, for instance the random, the LRU or the LFU. Make performance comparisons.
- Consider the case in which a sensor s provides data with device-specific expiration thresholds Δ_t^s. Consider the effect of this setting on the caching policy.

5 Performance Evaluation

The performance of your solution should be evaluated through multiple iterations, focusing on key metrics to understand its efficiency and reliability.
You have to consider the strength and weaknesses of your solution and propose metrics that reflect your considerations. Examples of evaluation criteria include the following (though not all of them are reasonable in all the provided scenarios):

- **Packet Delay**:
  Measures the time sensing data takes to be delivered to the base station. Since the delivery of a packet of data can require the movement of a device, this delay incorporates two components, a movement delay and a communication delay.
- **Lost data**:

Counts how much sensing data got lost and could not be delivered, either due to storage constraints, or because offloading to UAVs failed when transmission occurred.

- **Travel Distance**:
  Measures the distance travelled by packets stored on balloons.
- **Number of Hops**:
  Counts the number of hops the packets take before getting processed. This metric may be of particular interest for case study B, as solutions with smaller hop number may yield better results.

6 Project rules and exam evaluation

- You can work in teams of up to three students.
- The project challenges are presented in increasing order of difficulty, each with optional optimizations.
  - Project A requires passing also the written test to pass the exam. The final grade will be the average between the project grade and the written test.
  - For project B, if the delivered project does not meet exceptional standards, you will be required to pass the written test. Therefore, prepare for the written test if you choose this option. You can expect a better grade than with Project A for the project part of the exam, provided that you demonstrate a reasonable degree of novelty.
  - Project C allows you to pass the exam **without the written test**. If you are dissatisfied with the grade, you can try to improve it by either interacting with the professors to enhance your project solution or by taking the written test to increase the average of the project and written test scores.
- Both the grade of the written test and of the project are scored in the range between 18 and 30+.
- You are asked to provide a) your solution's code, b) a text report and c) a 3-minute video presentation of the project.
- No introduction of the problem is necessary in either the report or the video presentation; simply state the letter of the chosen project and proceed to present your solution. Each team member should present a part of the project in the video, ensuring that the parts presented by each member are equal in length and technical content.
- To ensure that each team member has actively contributed to the project, we reserve the right to request an in-person presentation or ask oral questions about the project if there are any doubts.