University of Pisa

Department of Computer Science

Master's Degree in

Computer Science

Master Thesis

Properties of Opportunistic Routing in the Argo Underwater Sensor Network

Candidate:

Flaviano Di Rienzo

Supervisors:

Prof. Stefano Chessa

Dr. Michele Girolami

Reviewer:

Prof. Giuseppe Prencipe

Academic Year 2015/2016

Acknowledgements

Table of Contents

Table of Figures

List of Tables

1 Introduction

Monitoring the underwater environment is a key task required to understand the climate changes that are taking place in our planet. In the last two decades, the global state of the underwater environment has become a major issue for the scientific community. In fact, several international research projects have been launched with the goal of monitoring oceans on a global scale.

The Underwater Acoustic Sensor Networks (UASN) represent a novel technology for monitoring the underwater environment. Nodes of a UASN are often equipped with sensors for monitoring the oceans (e.g. salinity or temperature sensors) and with acoustic modems for the communication purpose. The effectiveness of the communication is strongly influenced by many factors such as the temperature, the distance among nodes, the depth and the presence of obstacles between the nodes. A notable example of a research project exploiting underwater sensors is Argo.

Argo is an international project whose goal is to monitor the oceans using an array of free-drifting Lagrangian floats. Nodes of the Argo deployment are not able to communicate with each other directly, but only with a command center by using a satellite communication when they surface.

In this thesis, we study the viability of building a real UASN among nodes following the path of the Argo floats. This network enables a host of new applications, such as distributed fault-tolerance, back up among floats and the possibility for the Argo floats to communicate with the command centers by exploiting other floats as relays. As for example if a float is unable to transmit data to its command center because of malfunctioning of its antenna, it can forward its data to another float that is close to its surfacing phase. In addition, a command center could communicate with a float to reprogram its mission (e.g. prolong its deep-water drifting phase) without waiting for it to surface, by exploiting other floats that are emerged to forward the new directives to the target float.

We firstly analyze some important properties of the underwater network we built. Specifically, we analyze the network degree, the density and some properties of the connected components detected. Moreover, we also analyze two temporal metrics, namely the contact and inter-contact time among floats while varying the transmission range of the nodes of our network. The goal of this preliminary analysis is to study the structure and connectivity of the network.

We then focus on the routing problem in order to study the effectiveness of the underwater opportunistic communications among nodes. To this purpose, we select four well-known opportunistic routing algorithms, namely Epidemic, PROPHET, SprayAndWait and Direct Delivery as representative of different routing strategies. We analyze the message delivery ratio of such algorithms in order to assess the routing performance of the network.

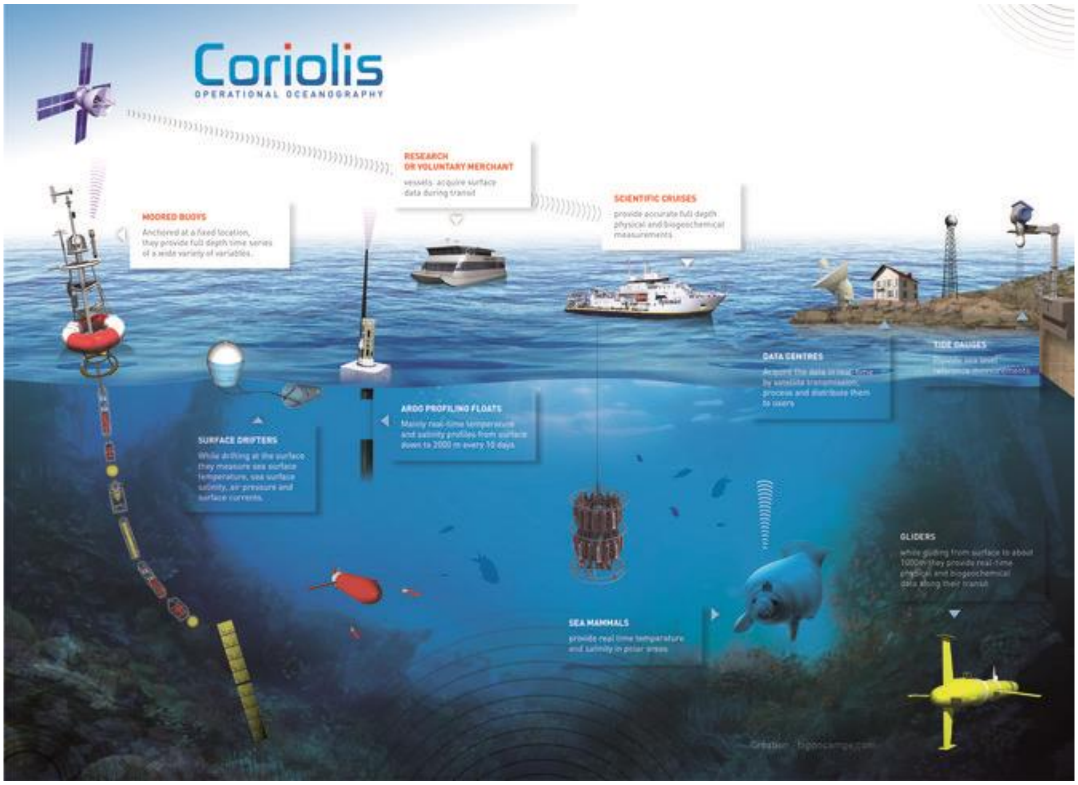
Our work is unique in several aspects when compared to the other works in this research field. Firstly, most of the results obtained with UASN do not rely on real-world mobility traces, rather they exploit synthetic mobility models to simulate the behaviour of UASN nodes. Secondly, most of the works simulate the UASN for short time periods, like few hours or a few days. Differently from the existing works, we strongly rely on the Argo traces for approximately one year of free-drifting. Finally, we select a challenging but realistic scenario for analyzing the performance of the routing algorithms, specifically a scenario in which the UASN is strongly disconnected. Such a scenario, allows us to test the algorithms selected with realistic conditions.

The structure of this thesis is organized as follows. Section 2, introduces the background of our work and discusses some related works. In particular, we introduce the reader to the concepts of Underwater Acoustic Sensor Networks, Delay Tolerant Networks and Opportunistic Routing. We also describe the Argo project. Section 3, discusses how we build a UASN by using the mobility traces of the Argo projects. Afterwards, we analyze the mobility patterns of the drifting floats, theirs density, the average degree and the connected as well as Temporal Out components. Section 4 describes the opportunistic routing protocols we used and the analysis of the performances obtained with the selected opportunistic routing protocols. Section 5 concludes this thesis with some possible future works.

2 Background and Related Works

2.1 Underwater Sensor Networks

Underwater sensor networks represent a new technology that can be used to monitor the depths of the bodies of water on our planet. USNs are composed of nodes immersed in a body of water (being it a lake, river, ocean or sea) equipped with sensors able to measure some physical parameter of the fluid they are immersed in. Nodes that are part of an USN must be able to communicate with each other using a transmission medium. As we will discuss in the next sections, the most used transmission medium for USN is the acoustic channel. Ultimately, data collected by the sensors that are part of an USN must be sent to a sink, so that they can be received by scientific stations and analysed.



Coriolis
OPERATIONAL OCEANOGRAPHY
RESEARCH OR VOLUNTARY MERCHANT
vessels acquire surface data during transit
MOORED BUOYS
Anchored at a fixed location, they provide full depth time series of a wide variety of variables.
SCIENTIFIC CRUISES
provide accurate full depth physical and biogeochemical measurements
DATA CENTRES
Acquire the data in real time by satellite transmission, process and distribute them to users
TIDE GAUGES
SURFACE DRIFTERS
While drifting at the surface they measure sea surface temperature, sea surface salinity, air pressure and surface currents.
ARGO PROFILING FLOATS
Mainly real-time temperature and salinity profiles from surface down to 2000 m every 10 days
GLIDERS
while gliding from surface to about 1000m they provide real-time physical and biogeochemical data along their transit
SEA MAMMALS
provide real time temperature and salinity in polar areas

Figure 1 Tools used for in-situ underwater monitoring.

USN offer various advantages compared to traditional technologies used in the past. Figure1 is taken from the Coriolis[1] website. The picture shows various tools used for monitoring the surface and underwater environments. In the figure are shown traditional tools like CTD[2] bottles, and modern technologies like wireless sensors. Belonging to the last category of underwater sensors are: free drifting sensors (on the surface and below the surface), sensors attached to sea mammals and Autonomous Underwater Vehicles.

2.1.1 UWSN versus Traditional Underwater Monitoring Techniques

The traditional approach used for monitoring the marine environment was to deploy sensing tools in the body of water and, at the end of the measurement, retrieve the sensors to read measured parameters. The profile mission of a traditional sensing device was to measure the physical parameters of interest while it is immersed in water and store them in its local memory. The data stored in the local memory of the device are read and analysed by scientists after the retrieval of the device.

Compared to UWSN, the traditional approach has multiple disadvantages[1], for example:

- *No Real-Time Monitoring:* the data measured by the sensing tools can be read only when the instruments are retrieved. This means that data may be available for analysis only after a consistent amount of time.
- *No online reconfiguration*: if some unusual events occur, it may be useful to reconfigure the sensing instruments to change their mission. Nevertheless, this is not possible with traditional sensing instruments, since there is no interaction between

[1] http://www.coriolis.eu.org/ Coriolis is a component of the French and European operational oceanography that deals with the acquisition, collection and validation of in-situ data for global oceans.

[2] CTD stands for Conductivity Temperature and Depth. A CTD bottle is lowered in water from a ship's deck with a winch and when the bottle reaches the desired depth, it closes itself collecting a sample of the body of water.

the sensors and the onshore control systems. Therefore, the deployed tools cannot be reconfigured during their mission.

- *No failure detection*: if a failure or a misconfiguration occurs during the monitoring, there is no way to notice it, until the sensing tools are retrieved. This can lead to a complete failure of the monitoring mission.
- *Limited storage capacity*: considering that measured data cannot be transferred during the mission, it must be stored inside the sensing tool memory. This limits the amount of data that can be measured to the memory capacity of the sensing tool.

Designing UWSN however leads to critical issues, such as efficient underwater communication.

2.1.2 Underwater Sensor Networks Communication

Wireless communication among Underwater Sensors can be performed by using different types of physical waves, such as acoustic, electromagnetic and optical waves. Each of the aforementioned communication techniques has advantages and limitations[2] [3].

- *Acoustic waves:* these are the most used communication system used in the underwater environments. The main advantage of using acoustic waves in underwater communication is the low attenuation of sound in water. Acoustic waves work well in deep and thermally stable water settings, while in shallow water Acoustic communication systems are hampered by surface ambient noise, reflection and refraction. Compared to EM and optical waves, acoustic waves are much slower. The sound propagation speed underwater is approximately equal to 1500 m/s.
- *Electromagnetic waves:* EM waves in radio frequencies are widely used on terrestrial communication for their fast propagation speed. Unfortunately, EM waves are not well suited for underwater communication, due to the conducting nature of the medium.
- *Free-space optical waves (FSO):* the utilization of FSO waves in underwater communication is limited by the severe attenuation of optical waves in water. This

phenomenon limits the range that can be reached by FSO waves. However, at short ranges (10-100m) optical waves offer high bandwidth communication (10-150 Megabits per second) channels.

Nowadays, for the reasons exposed above, Underwater Acoustic Sensor Networks are the most used typology of UWSN. Therefore, in the following sections we will consider only UWSNs that use acoustic waves for communication.

2.1.3 Underwater Acoustic Sensor Networks

Underwater Acoustic Sensor Networks' nodes may be of different nature:

- *Static two-dimensional UASN for ocean bottom monitoring:* the nodes of these UASN are placed on the ocean floor using deep ocean anchors. This type of UASN is composed of sensing nodes and underwater sinks. The sensors communicate with each other and with the sink using horizontal acoustic waves. The sinks receive the collected data by the sensor and, using vertical acoustic waves they communicate the data to a surface or onshore station.
- *Three-dimensional UASN with fixed sensors:* the nodes composing these kinds of UASN are anchored to a surface buoy or to an anchor moored at the sea floor. To preform vertical sampling of the ocean environment, the sensors must be able to modify their parking depth. To accomplish this, the sensors must be mounted to a floating buoy and must be able to either modify the length of the wire or adjust the internal pressure of the floating buoy. Using this type of UASN it is possible to cover a relatively large 3D area. The collected data can be transferred to an underwater sink or directly to a surface or onshore station.
- *Mobile UASN:* while the previous type of UASN mentioned can monitor only a fixed two or three dimensional area, Mobile UASN are not constrained to a single area. The nodes composing a Mobile UASN may be of two different types:

    - *Underwater Autonomous Vehicles(UAV):* this kind of nodes may be in complete or partial control of their movement (Remotely Operated Vehicles, ROVs).
    - *Lagrangian autonomous devices:* these nodes are passively transported by the ocean currents above or below sea surface. Devices moving only horizontally, are named *Lagrangian drifters,* while the ones moving both horizontally and vertically are named *Lagrangian floats.*

Static or fixed UASN are fairly dense and connected networks, offering a good coverage of the area they are moored in. Fixed UASN cannot be used for large scale monitoring, as to cover large areas, a huge amount of fixed sensors must be installed, thus increasing the costs of setting up the network.

The interest on Mobile UASN is rapidly growing, attracting interests from both academia and industries. UASN have low production cost (especially the Lagrangian devices) and have low deployment costs. Moreover, Mobile UASN can cover large areas. For example, the Lagrangian floats deployed by the Argo project are spread across the oceans and seas on a global scale (an in depth description of the Argo project will be done in Section 2.2).

2.1.4 UASN Application Fields

Underwater Acoustic Sensor Networks are used in a broad range of applications:

- *Environmental monitoring:* by measuring the presence of chemical, biological or radio-active hazardous substances in water, UASN can be used for pollution monitoring of the underwater environment. Water quality analysis has become a very important task, since global pollution is one of the main hazards to the marine ecosystem. This kind of analysis is very important for understanding the impact of the human activities on the underwater environment. Using UASN it is also possible to monitor ocean currents that, in addition to the monitoring of winds can lead to improved weather forecasts. Monitoring ocean currents and measuring the ocean temperature can help understanding the climate changes happening on the planet.

- *Biological Monitoring:* using UASN it is possible to monitor presence and movements of the marine fauna. Tracking fishes or aquatic micro-organism can be useful for fishery and preservation of the marine ecosystem.
- *Undersea Exploration:* UASN can be used to detect natural resources reservoirs such as oilfields or valuable minerals. Also undersea exploration can be useful to determine routes for laying undersea cables or pipes.
- *Disaster prevention:* underwater sensor networks can be used to measure the seabed seismic activity; it may be possible to warn coastal areas of the risk of tsunamis.
- *Assisted Navigation:* UASN can detect possible hazards for ship, such as rocks or shoals in shallow waters or submerged wrecks.
- *Distributed tactical surveillance:* UASN can also be used for military purposes, such as surveillance, reconnaissance, targeting and intrusion detection.

2.1.5 UASN Research Challenges

Even though in the last several years the concept of UASN has caught the attention of many researchers, it still offers major research challenges.

The most challenging topic is implementing efficient communication between underwater sensors, because of the limitations posed by the acoustic channel.

The acoustic channel is affected by multiple hampering factors, such as:

- *Attenuation*: is caused by the absorption of the acoustic signal in water and is affected by distance and frequency of the transmission. Specifically the attenuation increases as distance and frequency increase. This phenomenon limits the distance that can be reached by the underwater acoustic channel.
- *Noise*: the acoustic channel is also hampered by man-made noise (for instance, generated by ships, or by pumps) or ambient noise (can be generated by water movements, such as waves of storms, by seismic activity or by biological phenomena).

- *Propagation speed*: The propagation speed in the acoustic channel is five orders of magnitude lower than in the radio channel. This large propagation delay (0.67 s/km) can reduce the throughput of UASN considerably.

Another challenging research topic is the one concerning energy efficiency. The power needed for acoustic underwater communications is higher than in terrestrial radio communications due to higher distances and to more complex signal processing at the receivers to compensate for the limitations of the channel. Moreover, battery power is limited and usually batteries cannot be recharged, also because solar energy cannot be exploited.

2.2 The Argo Project

The Argo Project is an international project committed to gather data on the ocean underwater environment [4]. To this purpose, information on the ocean state, the Argo project exploits a fleet of free-floating Lagrangian floats able to measure temperature, pressure, salinity and other parameters (e.g. conductivity, chlorophyll, oxygen, and so on...) of the first 2000m of depth from the ocean surface.

The name Argo comes from Greek mythology, and has been chosen for its relation with the satellite JASON-1. In fact, Argo was the name of the vessel used by the Greek hero Jason and the Argonauts to search for the Golden Fleece. JASON-1 used to monitor the shape of the ocean surface. The combined data of the Argo floats and JASON-1 are used to monitor ocean currents, the movement of hot and cold water masses and the sea-level raise.

The Argo fleet is rapidly growing, currently (May 2016) the fleet is composed of approximately 3900 float, that provide nearly 100,000 temperature and salinity profiles per year. Argo is the only source of data on the global scale that provides information about the sub-surface part of the ocean.

2.2.1 Argo History

In 1990-1997 the World Climate Research Programme (WCRP) set out to make unprecedented in-situ and satellite observation of the global ocean. World Ocean Climate Experiment (WOCE) was part of WCRP and had the goal to collect data about the ocean currents at 1000 m depth. For this purpose, an array of Autonomous Lagrangian Circulation Explorer (ALACE) floats was created. ALACE floats used to drift at a fixed depth of 1000 m and to be carried by the currents. Each ALACE float used to resurface at regular intervals of time, so that its position could be computed by the satellite.

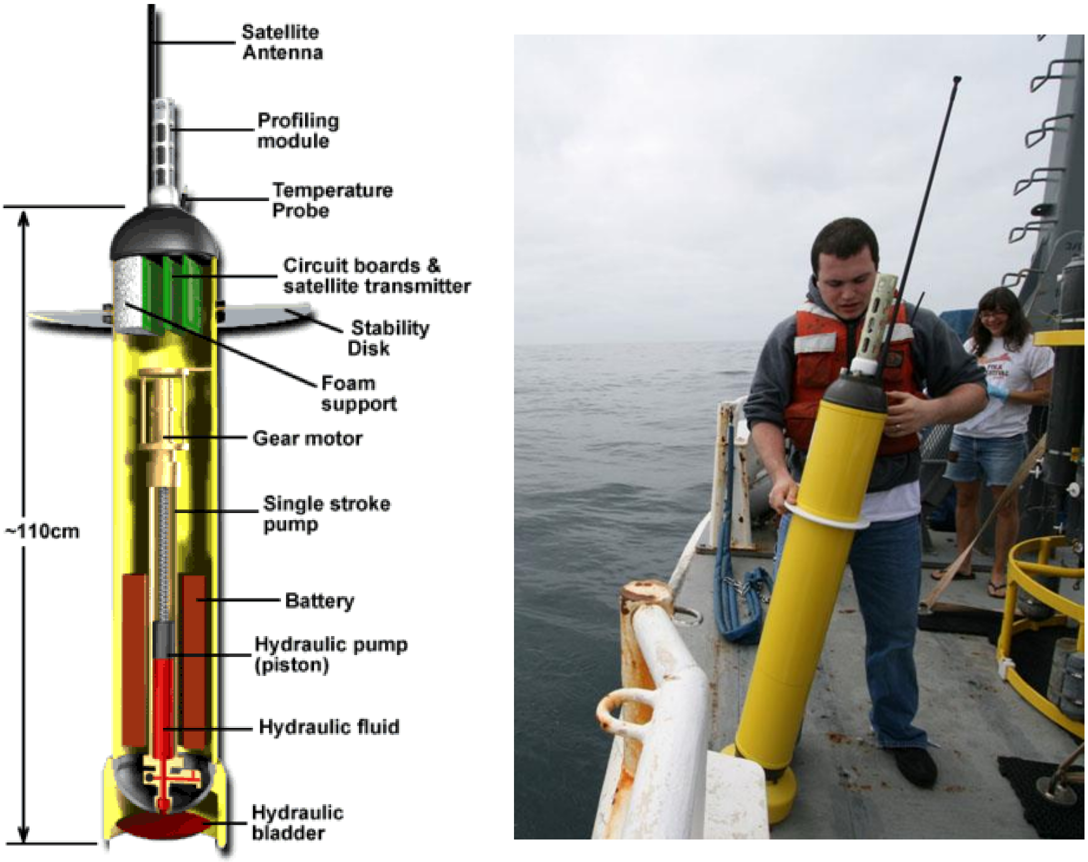
It was later realised that while resurfacing, the floats could also perform temperature and salinity profiles[5].

In 1998 the International Steering Team for the Global Ocean Data Assimilation Experiment (GODAE) planned to develop a global array of profiling floats, thus giving birth to the Argo project[3].

2.2.2 Argo floats

Argo floats appear as yellow aluminium cylinders with to 1.3 m height, a diameter approximately 20 cm, and weight close to 40 kg (Figure 2(b) shows a picture of a real Argo float, so the reader can have a clear idea of an Argo float dimension and shape). On the upper part of the float it is placed the antenna used for satellite communication. The satellite antenna is used both to transfer data from the float to the satellite and to determine the position of the float when it emerges on the surface. Near the antenna are placed the temperature and salinity sensors. On the lower part of float, there is the mechanism used to adjust the internal pressure of the float, to make the float dive or rise in water.

[3] http://www.argo.ucsd.edu/Origins_of_Argo.html



(a)

(b)

Figure 2 (a) Internal structure of an Argo float. (b) Picture of real Argo float.

As shown in Figure 2(a), Argo floats are composed by three sub-systems:

- The hydraulic system that manages the floats buoyancy, thus influencing the vertical movement of the float.
- The microprocessors, that provide control and scheduling.
- The transmission system that manages communication with the satellites.

Argo floats are deployed in water from boats (as can be seen in Figure 2(b)) or planes. As we mentioned in previous paragraphs, the deployment of Lagrangian floats is very cheap compared to the other typologies of UASN nodes. Argo floats deployment do not include setting up moored buoy or anchors attached to the sensors, but they can be simply laid down in water from ships or planes used as vessel of opportunity.

2.2.3 Argo floats life cycle

Argo floats' life cycle follow the "park and profile" schema shown in Figure 3.

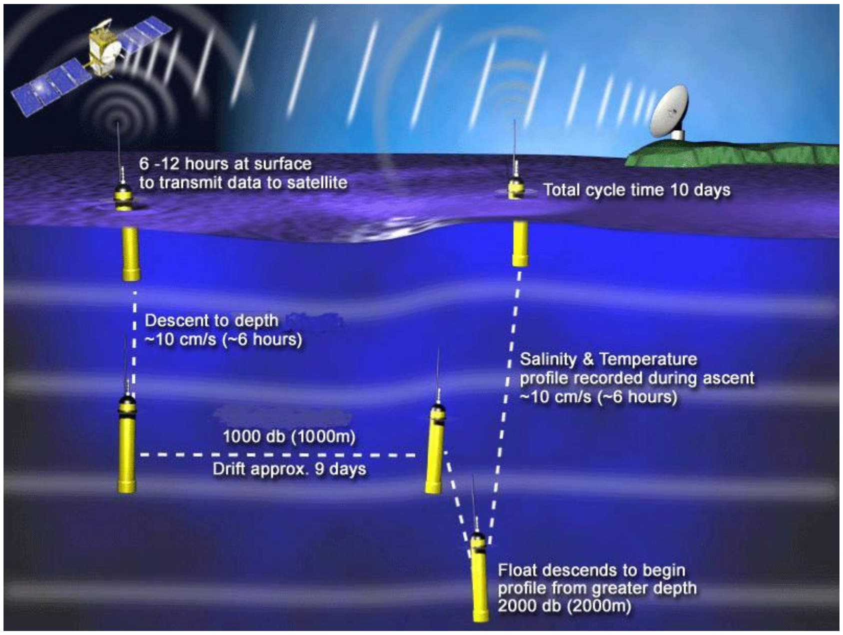


Figure 3 Argo float life cycle

The typical life cycle of an Argo float is composed of the following phases:

- After being deployed in water, the float deflates the hydraulic bladder placed in the lower part of the float's structure, to increase its density. After performing this operation, the float sinks into the water until it reaches 1000 m deep.
- When it reaches the "parking" depth, the float stops sinking and is passively carried by the currents. Every Argo float is designed to assume the same density of the water in the "parking" depth. In this way, the float stays in the same depth for the entire duration of the parking interval. The typical parking time is 9 – 10 days.
- At the end of the parking period, the float sinks below the parking depth, by operating its hydraulic bladder. The float dives until it reaches 2000 m deep.

- After reaching the maximum depth, the float begins its vertical “profile” task. The float inflates the hydraulic bladder and starts raising in water, until it reaches the surface. While emerging, temperature, salinity and pressure profiles are performed (some floats measure also other parameters, such as conductivity, oxygen and chlorophyll presence). As the float rises, a series of about 200 pressure, temperature and salinity profiles are made and stored on board the float.

  Vertical profiling can be performed using two methods:

  - *Discrete sampling:* it returns a sampling for every depth interval
  - *Continuous sampling:* it returns an average sampling for every depth interval

  Considering that profiles performed using these two methods can be different, a field is present in the profile format, “VERTICAL_SAMPING_SCHEME”, pointing out the method used to perform the vertical profile.
- When the float reaches the surface, it communicates its profiles to the satellite. For a certain period of time (approximately 6h – 12 h), the float stays on the surface, and it computes its position multiple times, using the GPS or the Argos System. Every time the float computes its position, it communicates its coordinates to the satellite.
- At the end of the communication with the satellite, the float dives in water and restarts its activity cycle.

The activity cycle described above is followed roughly by every Argo float deployed. However, some float may have a slightly different life cycle, depending on the float model and the area they are deployed in. The differences may be the depth reached and the life cycle duration. For instance, in 2015 80% of the Argo fleet is composed of floats that can perform sampling below 1500 m, while 12 % of the deployed Argo floats can reach at maximum between 1000 m and 1500 m deep. Also as shown in Figure 4, not all the floats have the same life cycle duration.

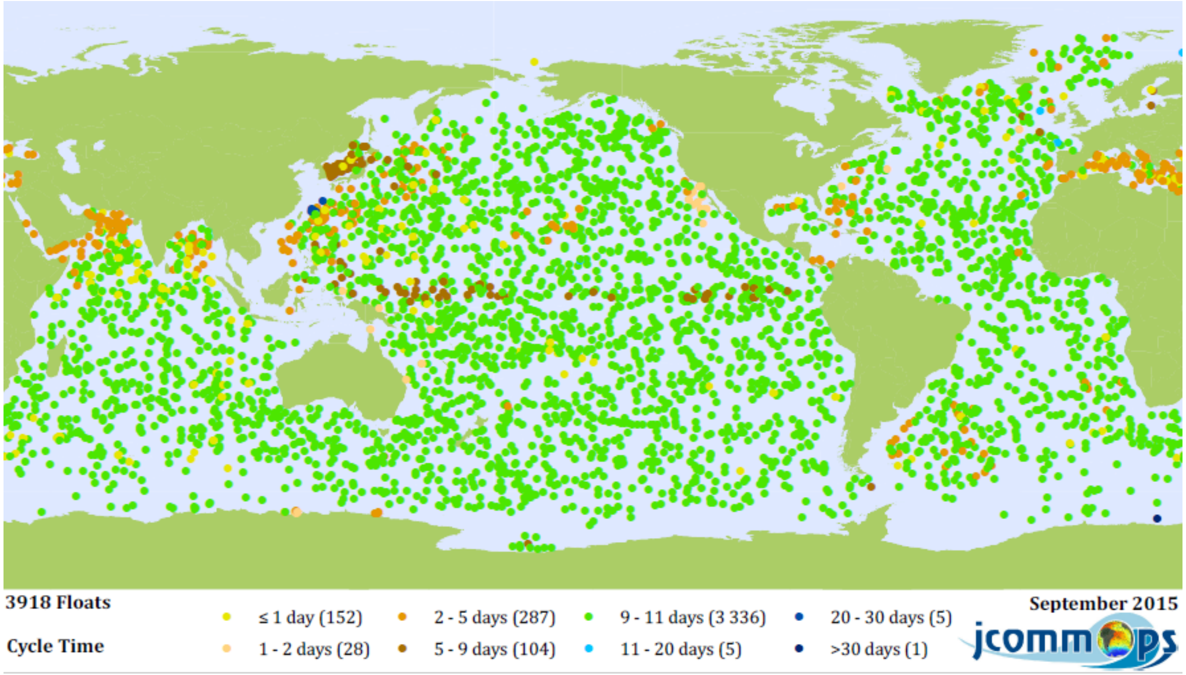


Figure 4 Map of the Argo floats, different colours correspond to different life cycle duration.

Figure 4 shows the life cycle durations for every deployed float in September 2015. As we can see, while the majority of the floats has a life cycle ranging from 9 to 11 days, there are some floats whose life cycle duration is different. In particular, we can observe that in some areas, such as the Mediterranean Sea, the Arabic Sea and the Persis Gulf there is a strong presence of floats whose life cycle is 2 – 5 days long.

Lastly there are some models of float, (such as the new SOLO-II) that profile on descent[4]. These floats execute the "parking" and "profiling" tasks in reverse order. At the beginning of the life cycle of the float, that is when the float leaves the sea surface, the float sinks until it reaches the maximum depth. While the float sinks, it preforms the vertical sampling. When

[4] www.argo.ucsd.edu/Argo_traj_timing_variables.pdf

the float reaches the maximum depth, it ascends to the parking depth, where it remains for the entire parking period duration. After the end of the parking period, the float resurfaces.

Figure 5 shows a more detailed schema of the life cycle of an Argo float, taking into account also the floats with a “reverse” life cycle.

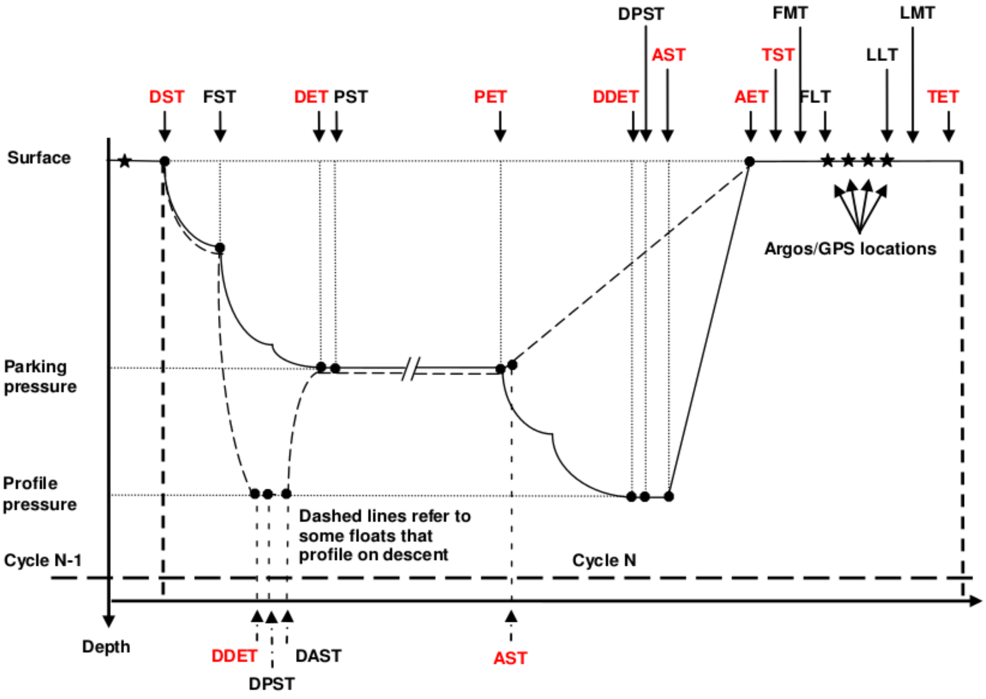


Figure 5 Detailed schema for the Argo floats life cycle.

In Figure 5 are represented both types of the Argo float life cycle. The solid black line represents the life cycle of floats that perform the “profile” task while ascending after the “parking” period. The dashed line represents the life cycle of the other type of floats that perform the vertical “profiling” task while descending and before the “parking” period.

In the figure are also reported the cycle timing variables. The variables highlighted in red are mandatory, while the black ones are typically used for stabilization (a more detailed description of an Argo float life cycle is written in).

2.2.4 Argo floats Communication System

As we mentioned before, when the floats reach the surface, they send the collected data to the satellites. The 60% of the floats of the Argo array uses the *Système Argos* for localization and data transmission. To guarantee the correctness of localization and data transmission, in all weather conditions, the float must spend 6 – 12 hours at the surface. The positions are accurate with an approximation of ~100 m.

However, the newer models of Argo floats do not use Argos as communication system. Instead they use the Global Positioning System (GPS) for localization and the Iridium satellites for data transfer. Using the GPS and Iridium allows more detailed profiles to be transmitted within a shorter period of time, so the floats can spend less time at the surface to communicate effectively with the satellite.

Currently, 40% of the deployed Argo floats communicates with the Iridium satellites. Using the Iridium system allows a two-way communication between the float and the satellite. Using a two-way communication, it is possible to change a float mission. This is very important when extraordinary meteorological events, such as tropical cyclones, occur.

2.2.5 Argo Data System

The data collected by the floats and transferred via the Argos satellites, are collected by the Argo Information Centre (AIC) in France and then received by national Data Assembly Centres (DACs). The data transferred using systems different from Argos may be collected by the owner of the float instead of the AIC before being received by national DACs.

At the DACs the data are subject to a set of real-time quality control tests. Erroneous data are either flagged or corrected before being passed to the two Global DACs. Figure 6(b) shows the distribution of the floats among the national DACs. It can be seen that the majority of the floats is managed by the Atlantic Oceanographic & Meteorological Laboratory (AOML).

Data are made available publicly by the two GDACs, one is managed by Ifremer/Coriolis located in Brest, in France and the other is managed by USGODAE located in Monterrey in California. To make sure data are consistent, they are synchronized among the two GDACs.

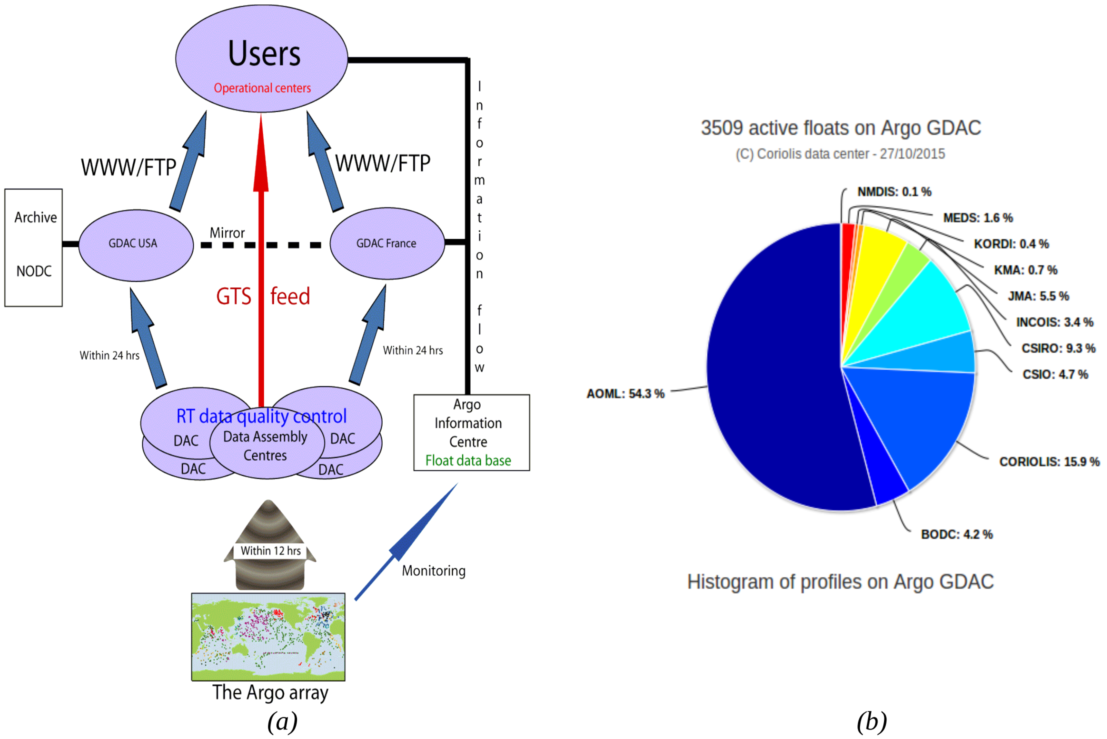


Figure 6 (a) Real-time data flow. (b) Percentage of floats managed by national DACs.

Data are published in two different modes:

- *Real-time mode:* data are made available after a set of automatic quality control tests. Generally, these data are made available after approximately 24 hours.
- *Delayed mode:* data are made available within a year after being scientifically validated.

Figure 6(a) shows the flow of data published with the real-time mode that is the path followed by the data collected by the Argo floats before they become available to the users.

2.3 Delay Tolerant Networks

Mobile Underwater Acoustic Sensor Networks can be classified as a specific case of Mobile Ad hoc Networks (MANETs). More specifically, two nodes part of a MANET can communicate directly if they lie within the transmission range of other nodes. A node that is part of a MANET, can also act as a relay node, in other words a relay node can be used as an intermediate hop between the sender and the receiver of a message. The main difference of MUASN with respect to traditional (static) networks, is that the topology of the network is continuously changing because of the movement of the nodes. Thus, routing in MANET and hence also in MUASN is a non-trivial task, subject of intensive studies [6] [7].

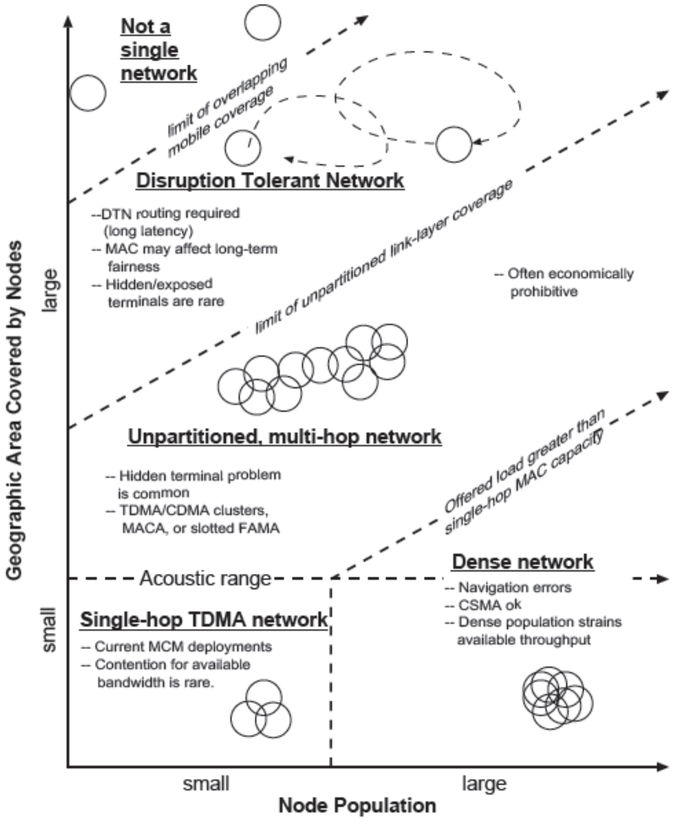

Not a single network
limit of overlapping mobile coverage
Disruption Tolerant Network
--DTN routing required (long latency)
-- MAC may affect long-term fairness
-- Hidden/exposed terminals are rare
limit of unpartitioned link-layer coverage
-- Often economically prohibitive
Unpartitioned, multi-hop network
-- Hidden terminal problem is common
-- TDMA/CDMA clusters, MACA, or slotted FAMA
Offered load greater than single-hop MAC capacity
Acoustic range
Dense network
-- Navigation errors
-- CSMA ok
-- Dense population strains available throughput
Single-hop TDMA network
-- Current MCM deployments
-- Contention for available bandwidth is rare.
Geographic Area Covered by Nodes
large
small
small
large
Node Population

Figure 7MANET: Area Covered by nodes vs Node Population

There are many routing protocols that have been specifically designed for MANETs, such as AODV (ad-hoc On-demand Distance Vector Routing), DSR (Dynamic Source Routing), DSDV (highly dynamic Destination-Sequenced Distance Vector) and OLSR (Optimized Link State Routing)

Most of the aforementioned protocols assume the existence of a contemporaneous end-to-end path between the source and the destination of the message. This assumption is not always true in MANETs, as depending on the nodes movement, the network can become disconnected. In particular, it may be possible that two nodes are never part of the same connected component.

Networks that do not guarantee end-to-end connection between nodes are called Delay (Disruption) Tolerant Networks (DTNs) [8]. To route packets between the nodes of a DTN, new routing protocols must be used. These protocols fall under the name of *Opportunistic Routing Protocols*. The concept of the store-and-forward routing model is extended to a store-carry-forward model (SCF).

In the SCF routing model, a node - being it the sender or a relay - must be able to keep a message to forward in its buffer until the next hop becomes available.

From **Errore. L'origine riferimento non è stata trovata.**, we can see how MANETs can be categorized depending on the ratio between the area covered by nodes, and the node population. We can see that MANETs covering large areas **Errore. L'origine riferimento non è stata trovata.**fall in the DTN category.

The network composed by the Argo floats that we are going to simulate, covers a very large geographic area (it covers the entire Mediterranean Sea area). As we will see, using short transmission ranges (< 10 km) the Argo network is completely disconnected, in other words (as shown in **Errore. L'origine riferimento non è stata trovata.**) it cannot be considered a network. But for longer transmission ranges, the Argo network can be considered a DTN.

That is why we chose to use opportunistic routing protocols to allow communication between the nodes that are part of the Argo network.

2.3.1 Opportunistic Routing

The key problem in opportunistic routing is deciding, while forwarding a message, the policy for the choice of the next-hop. In other words, if a node *n* is carrying a message to forward and the message cannot be delivered directly to its destination, *n* must pass the message to the “best” next hop, which is the node that guarantees the highest delivery rate. If the next hop is not chosen properly, the message may be delivered with a huge delay or not be delivered at all.

Opportunistic Routing protocols can be divided in two categories:

- *Deterministic:* a deterministic protocol assumes that the future movements of the nodes are known (or at least predictable). By knowing the future topologies of the network, it is possible to compute the best route from source to destination of every message. Some examples of deterministic opportunistic routing protocols are: *Space-time routing* [9], *Oracle based routing* [10], *Link state based* [11] and *Tree based* [12].
- *Stochastic routing:* if the future topology of the network is not known, stochastic protocols must be used. If the protocol does not make assumptions on the network topology and it randomly forwards messages, then it is part of the *Epidemic routing protocol* category. Protocols where nodes estimate the delivery probability of their neighbours before forwarding a message are named *Estimation-based* or *History-based routing protocols*. Belonging to this category are the *PROPHET* and *SprayAndWait* protocols. Protocols where nodes use mobility patterns to estimate the forwarding probability, are named *Mobility-based protocols.*

In Section 4.1 we will provide a more detailed description of the routing protocols Epidemic, PROPHET and SprayAndWait.

2.4 Related Works

Different studies have been made to design efficient Mobile UASN. Some studies focused on the analysis of nodes' mobility, while other tried to design new ad-hoc routing protocols for UASN.

One of the most important concepts to understand while dealing with Mobile UASN is the mobility model of the nodes. The mobility model must take into account the fluid nature of the medium in which the nodes move. A widely used mobility model is the *Meandering Current Model (MCM)* described in [13].

Because oceans are stratified rotating fluids, vertical movements are negligible compared to horizontal ones. Events that can cause substantial vertical movements (such as upwelling, down welling or deep water formation) are very rare. For this reason, MCM assumes that the nodes of a Mobile UASN move only by following a horizontal trajectory. MCM uses the two-dimensional streamfunction defined in [14] to model the movements of lagrangian drifters.

In [13] the behaviour of a Mobile UASN following the mobility model MCM has been simulated. During simulations, the movements of the nodes and the connectivity of the network have been observed in an area of 80 km x 8 km for 5 days.

MCM inspired some new three-dimensional mobility models for Lagrangian devices, such as the ones in [15], used to investigate the feasibility of Mobile UASN using pressure routing protocols, and in [16].

The main differences between our study and the one made in [13] are:

- We analysed movement and connectivity of an UASN build using real data instead of synthetic ones.
- We performed our analysis on much larger areas. While in [13], the analysed area spanned 640 km^2, in our study, we observed properties of UASN deployed in basins

spanning million of km^2 (for instance one of the analysed basins was the Mediterranean sea, spanning 2,500,000 km^2).

- We observed movements and connectivity of our Mobile UASN for a longer period. In [13] the observation period lasted a few days, while our observation period lasted one year.

Multiple studies have been made on the design and analysis of Opportunistic Routing protocols on MANET [17] [18] [19]. However, many of these studies have been performed on terrestrial MANET, rather than underwater ones. A broad survey of opportunistic networks' protocols, main projects and applications has been discussed in [20]. Opportunistic Routing has been analysed in various contexts, such as Space Communication, Wildlife Monitoring, and Social Applications, just to cite a few examples.

A similar approach to the one we adopted in our work, has been used in [21]. Chaintreau et al. used real data collected by devices carried by humans, to perform their experiments. In particular, they collected data on the contacts occurred between the devices. Then, they measured the distribution of the Intercontact Time as well as the performances of some opportunistic routing algorithms in the network build exploiting the collected data.

To the best of our knowledge, only few studies have been made on opportunistic routing in the UASN domain. One of them is [22], where a new Opportunistic Routing protocol for UASNs has been proposed, the *GEographic and opportunistic routing with Depth Adjustment-based topology control for communication Recovery over void regions (GEDAR)* routing protocol.

GEDAR is an opportunistic, geographic and depth based protocol. Geographic routing algorithms exploits the nodes position awareness to route packets. Depth based routing algorithms are used to greedily route packets from nodes floating in deep waters to sinks drifting on the sea surface.

GEDAR has been designed for dense underwater networks, and it exploits opportunistic routing to improve network performances.

Conversely, in our work we are dealing with a very sparse network and we study opportunistic routing not as an optimization layer, but as the only possibility to diffuse packets in a strongly disconnected network.

Also, we do not plan to design new opportunistic routing for UASN, but we exploit well known opportunistic protocols to establish communication between the nodes.

3 The Argo Dataset

For our study on Mobile Acoustic Underwater Sensor Networks, we decided to exploit real-world mobility data. We used data produced by the Argo Lagrangian floats. These data can be selected via the web interface available in the Argo Data Management [5] website. The Coriolis GDAC then delivers the selected data to the user.

We built a fictitious Mobile wireless UASN whose nodes are represented by the Argo floats. By knowing the positions of the floats for a substantial interval of time, we can simulate their movements. We assumed that the nodes of the network are equipped with an acoustic modem for underwater communications. Then we analysed movement, connectivity and message delivery of the network.

3.1 Overview of the Dataset

The Argo dataset collects all data gathered by the Argo floats, such as measured parameters (such as temperature, salinity, conductivity, dissolved oxygen, chlorophyll fluorescence and many others) and followed trajectories. The two Global Data Assembly Centres that we introduced in Section 2.2.5, Coriolis and USGODAE, manage the dataset. The data stored in the Argo dataset contains information about the physical parameters collected by the floats, such as the temperature, the salinity and the pressure, and moreover the dataset also provides the coordinates of the floats as WGS84 coordinates.

The dataset can be obtained in two different formats:

[5] http://www.argodatamgt.org/Access-to-data/Argo-data-selection

- *Argo profiles:* these files contain information about the physical parameters measured by the floats, as well as the Argo floats identifiers and the coordinates in space and time of each performed measurement.
- *Argo trajectories:* these files contain only information about the recorded positions of the floats.

For the purpose of our analysis, we needed only data about the locations of the floats and the time instants when these were collected. For this reason, we ignored the Argo profile files and we download only the Argo trajectory files.

The format of data contained in the trajectory files includes the following informations[6]:

- *PLATFORM*: contains the platform code. If it exists, it is the WMO (World Meteorological Organization), otherwise it is an internal code. It acts as the unique identifier for the platform.
- *ARGOS_ID*: it is the beacon code of the platform. This field is optional.
- *DATE (yyyy-mm-ddThh:mi:ssZ)*: The date/time of the observation. The date and times are given in UTC time, using the ISO 8601 international standard representation.
- *LATITUDE (degree_north)*: latitude of the observation
- *LONGITUDE (degree_east)*: longitude of the observation
- *QC*: it stands for Quality Control, and it concatenates the quality control flags of each of the previous fields. Table 1 lists the meanings of the quality control flags.

[6] A detailed description of the Argo profiles and trajectory data format is available at http://www.coriolis.eu.org/content/download/22362/154150/file/csv-oco.pdf

Quality flag	**Meaning**
0	No Quality Control performed
1	Good Data
2	Probably good data
3	Probably bad data
4	Bad data
5	Value changed
6	Not used
7	Not used
8	Interpolated value
9	Missing value

Table 1 Quality control flags

3.2 Data Selection and Cleaning: The Mediterranean Case-Study

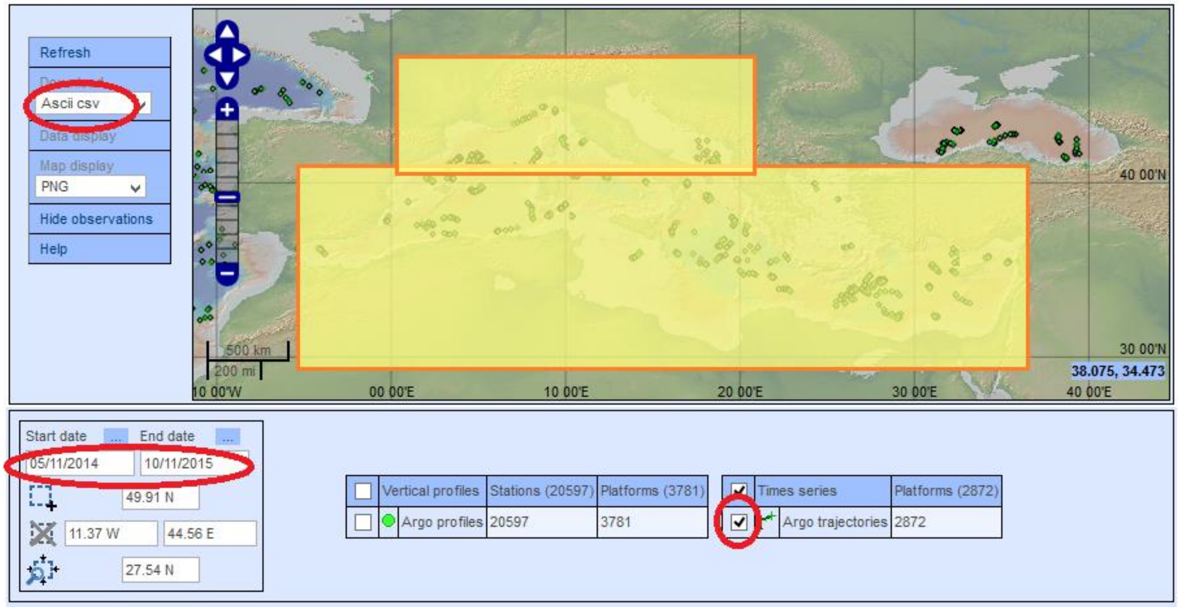
The number and the positions of the Argo floats changes over the time. At the time we started working with Argo (November 2015), the dataset counted 3500 free-drifting floats. Considering that the oceans' surface covers about$3.6 * 10^8 km^2$, the density of the floats is one every 10^8 km^2.

Therefore, our first challenge has been selecting the portion of the hydrosphere to observe so that to consider a sufficiently dense area. We considered the following constraints for the selection of the area of interest:

1) select an area of interest with an *acceptable* number of floats (as we pointed out in **Errore. L'origine riferimento non è stata trovata.**, Section 2.3)

2) Restrict the observation to “closed” subset of the Argo floats, in other words, we did not want to deal with a network where nodes were moving inside and outside from the observation domain. To obtain a “closed” set of nodes, the best choice was to select a basin enclosed by the landmasses.

Our choice was to consider the Mediterranean Sea as our observation domain. The Mediterranean Sea is one of the densest basins of the world containing an average of 25 active Argo floats at the same time in an area covering 2,500,000 km^2. This means that in the Mediterranean Sea there are approximately one float every 100,000 km^2, and thus its density is much higher than that of the global one (nearly one thousand times higher). In addition, the Mediterranean Sea is the only basin containing salt water that is almost entirely enclosed by landmasses and having a significant Argo float population.



Ascii csv

PNG

Start date: 05/11/2014 End date: 10/11/2015

49.91 N

11.37 W 44.56 E

27.54 N

□	Vertical profiles	Stations (20597)	Platforms (3781)
□	● Argo profiles	20597	3781

☑	Times series	Platforms (2872)
☑	Argo trajectories	2872

Figure 8 Mediterranean Data Selection

Figure 8 shows the Argo Data Management data selection interface as well as the selection criteria set for retrieving the data. In particular, we selected a considerable observation period of 370 days, from 5th November 2014 to 10th November 2015. In addition, we downloaded only the Argo trajectory files, since we are interested only in the coordinates of the floats.

We made two selections to capture all the floats drifting in the Mediterranean Sea and exclude the ones present in nearby basins like the Atlantic Ocean and the Black Sea.

The downloaded datasets are merged file by file, that is for each float appearing in both datasets the data referring to the shared float are concatenated and sorted by date to rebuild the complete trajectory followed by the device. The resulting dataset is composed by the data referring to the 51 Argo float that travelled in the Mediterranean Sea during the selected observation period.

It is worth to notice that data collected by the Argo project and stored in the GDACs are updated periodically[7]. Therefore, if a user performs a selection and after some time (e.g. few months), he performs another selection using the same parameters (i.e. same time interval and same geographical area), than the user can receive slightly different data. Even data published in Delayed Mode are subject to changes over time.

3.2.1 Data Quality Analysis

The Quality Control field gives us information about the quality of the downloaded data. Concerning the trajectory files, the QC field is made of 5 digits, one for each of the fields in the trajectory file (Platform ID, Argos ID, Date, Latitude and Longitude).

By analysing the QC codes of the trajectory files we downloaded, we can see that the first two digits are always 0, this means that (as reported in Table 1) the Assembly Centres performed no quality control checks on the Platform and Argos identifiers.

The last three digits correspond to the fields Date, Latitude and Longitude. Concerning these three parameters, we decided to use for our analysis only data flagged good by the quality controls. As reported in Table 1, data flagged “good” by the quality checks correspond to a

[7] http://www.argo.ucsd.edu/Data_FAQ.html#reD

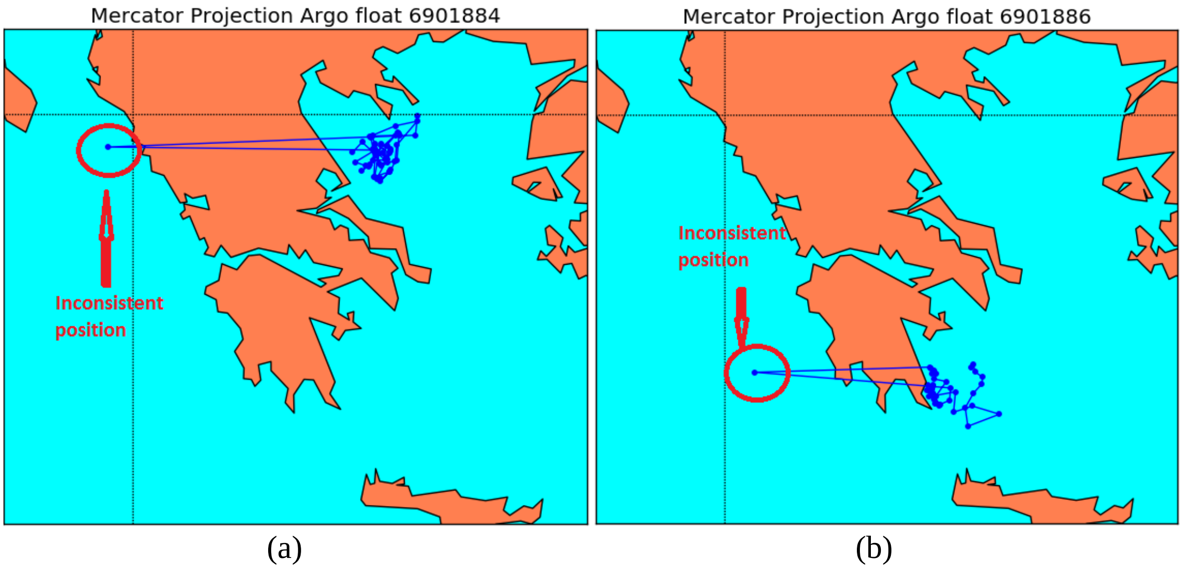
1 in the QC filed. For this reason, we decided to remove every row in the trajectory files whose QC field differs from 00111.

Furthermore, to ascertain consistency of the data flagged good by the quality controls, and to have a clear idea of the floats movements, we generated a map for every Argo trajectory file representing the paths followed by the devices. We used the python[8] library Basemap[9] to generate maps.

We found out that there are two trajectory files corresponding to the Argo floats 6901884 and 6901886, containing inconsistent data. The two maps in Figure 9 are generated using the positions of the Argo floats 6901884 and 6901886 contained in their respective trajectory files. The points in Figure 9 correspond to the geographical coordinates contained in the trajectory files of the floats. Being $t_0, t_1, \ldots, t_n$the timestamps of the recorded positions of the floats, the lines in the figure link the points corresponding to positions with timestamps t_i and t_{i+1}.

[8] Python Programming Language https://www.python.org/

[9] http://matplotlib.org/basemap/


Mercator Projection Argo float 6901884
Inconsistent position
Mercator Projection Argo float 6901886
Inconsistent position

(a) (b)

Figure 9 Inconsistent Argo trajectories

As we can see, in the both the trajectory files, there is a recorded position that is far away from the others. In particular, nearly all the positions are gathered in an area east of Greece, but in both trajectory files, there is a position corresponding to a point west of the Hellenic peninsula. Obviously these positions are not consistent, even if they are flagged correct by the quality controls. We remove these positions from our selection as we did with the positions flagged incorrect by the quality controls.

3.2.2 Argo floats timelines

In this Section, we analyse the timestamps of the Argo floats drifting in the Mediterranean Sea. We are interested in understanding the frequency with which the positions are communicated to the satellites and if the floats communicate their positions for the entire duration of the observation period or only for a fraction of it.

In this Section and in the following ones, we will consider an Argo float "active" in a certain time instant t if t is between the timestamps of the first and the last known positions of the float in our observation period. Furthermore, we consider a float F "inactive" in a time window $[t_k, t_{k+1}]$ if t_k and t_{k+1} are two subsequent timestamps associated to two positions of

F and the length of the time interval $[t_k, t_{k+1}]$ is longer than the median of the length of F's activity cycle plus one day.

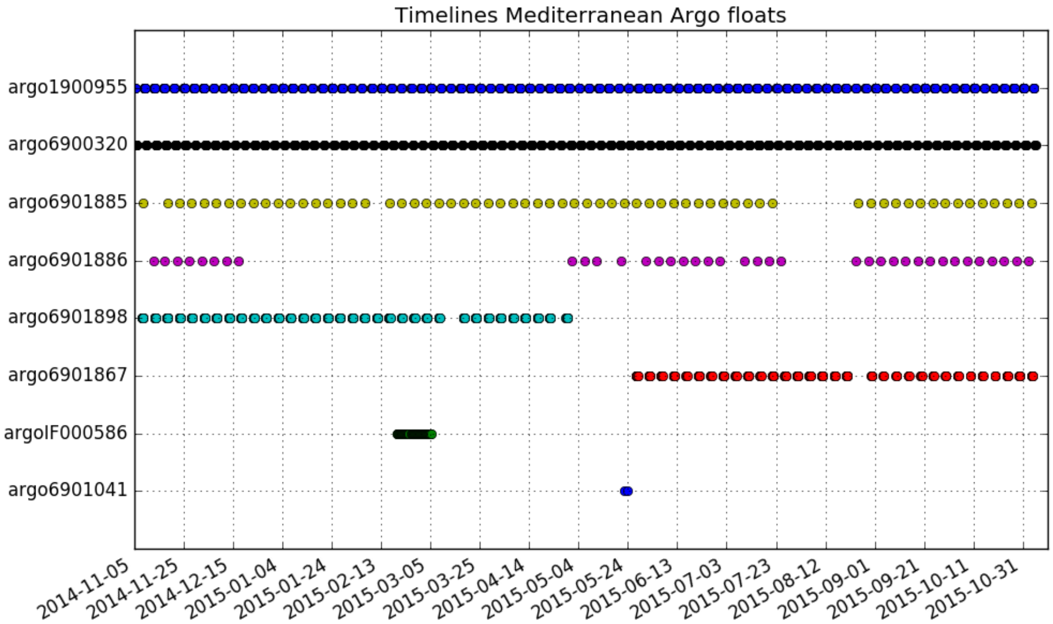


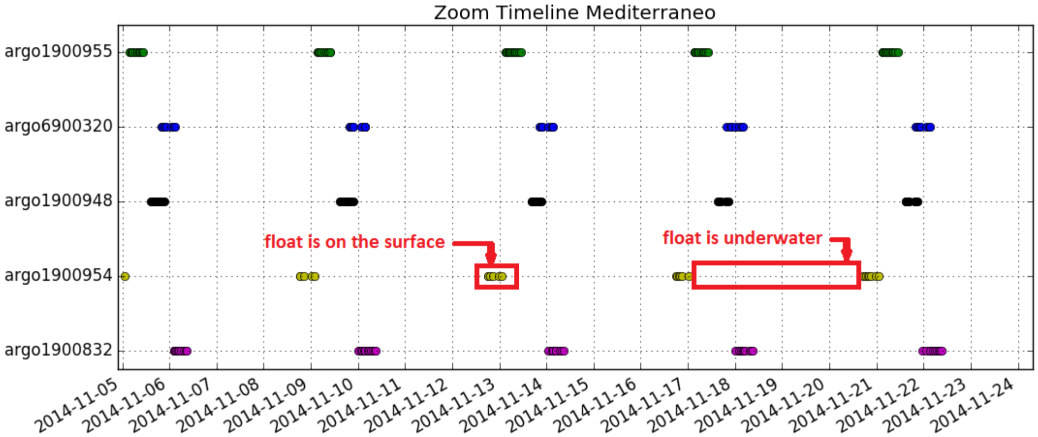
Figure 10 Timelines Argo floats

Figure 10 shows a time series related to 8 Argo floats that are part of our dataset. On the y-axis of the plot, we report the identifier of the float, while on the x-axis we report the time instants composing the observation period. The plotted points correspond to the time instants in which a float reports a position. As we can see from Figure 10, only a few floats (e.g. floats with identifier `1900955` and `6900320`) report their positions for nearly the entire duration of the observation period. Differently, the majority of the Argo floats in our dataset are active only for a fraction of the observation period. Such a scarcity of reported positions is caused by the several factors:

- Some floats are deployed in the Mediterranean Sea after the start of our observation period.

- Some floats cease to work (possibly because of battery depletion) before the end of the observation period.
- Some floats stop sending data to the satellites for relatively long periods (possible reasons may be malfunctioning, bad weather conditions or damages).

Moreover, floats IF000586 and 6901041 reported in Figure 10 report a very limited number of positions. We consider that such floats are not relevant for our analysis therefore we introduce a filtering procedure to drop out non-useful floats We decided to ignore those floats reporting their position for less than τ % of the observation period; in this work we set τ = 6%. Therefore, after filtering the floats, we exclude 5 floats reducing the number of floats in our dataset from 51 to 46 floats.


Zoom Timeline Mediterraneo

float is on the surface

float is underwater

Figure 11 Zoom on the Mediterranean Argo floats Timelines

By taking a closer look to the timelines of the Argo floats drifting in the Mediterranean Sea (Figure 11), we can observe that there are timespans of few hours in which a float has many associated positions, and differently there are timespans of few days in which floats have no associated positions at all. This phenomenon is caused by the behaviour of the Argo floats we discussed in Section 2.2.3. Every Argo float, when it emerges, communicates multiple times with the satellite (computing and transmitting its position). When a float dives, it stays in water for an amount of time equal to the sum of the descend, park and profile times. When the float is underwater it cannot communicate with the satellite, thus we have no means to know its exact geographical position.

Considering that an Argo float spends most of its time underwater, in the following we will approximate their activity cycle length with the time it spends under the sea surface.

3.3 Building the Argo Underwater Sensor Network

Given the positions of the Argo floats, we can build and simulate the behaviour of the Argo Mobile Underwater Sensor Network. Our network can be seen as a Temporal Network [23], in which the nodes are the floats and an edge between two nodes reflects the possibility for those nodes to interact directly.

While building the Argo Underwater Sensor Network, we make the following simplification: considering that the Argo floats spend the vast majority of their time drifting at approximately the same depth, we do not consider depth while computing proximity among the floats. In other words, we will use only the geographic coordinates of the floats to compute distances among them.

3.3.1 The Temporal Network model

To model the Argo Underwater Sensor Network (AUSN), we use the Time-varying graph metric used to represent Temporal Networks.

A time-varying is an ordered sequence of M graphs $G_{[0,M]} = \{G_1, G_2, \ldots, G_M\}$ obtained during the observation period $[0, M]$. Each graph $G_m = (V,E_m)$ represents the snapshot of the network during the time window $[t_m, t_m + \Delta t_m]$, where V is the set of nodes and E_m are the edges among the nodes. It is worth to notice that the edges in E_m are only valid for the time window $[t_m, t_m + \Delta t_m]$

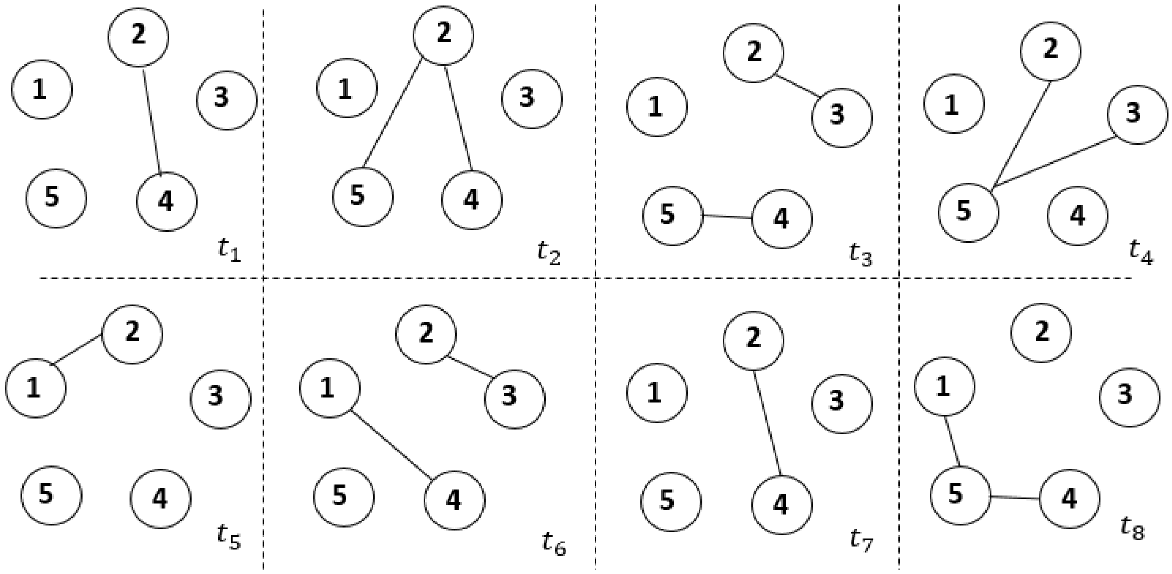


Figure 12 Time-varying graph

Figure 12 shows an example of time-varying graph.

By using the notation introduced in [23], we can deduce that $(t_M + \Delta t_M) - t_1$ represents the temporal length of the observation period.

3.3.2 Interpolating the Positions of the Argo floats

In order to represent the AUSN as a time-varying graph, we split the observation period in a set of time windows $\{[t_1, t_1 + \Delta t_1], [t_2, t_2 + \Delta t_2], \dots, [t_M, t_M + \Delta t_M]\}$.

We set the initial observation time $t_1 = 00:00:00\ 5th\ November\ 2014$, the length of every snapshot to $\Delta t_i = 1$ day and the final observation time $t_M = 00:00:00\ 10th\ November\ 2015$. As a result, we obtain M=370 different snapshots of the network.

Two nodes are connected by an edge if they can exchange a message. To model the transmission capability of a node, we set γ as the transmission range of the nodes that is the maximum distance at which a node can transmit a radio signal. Therefore, we create an indirect edge between the node pair $n_i, n_j \in V$ if the distance d between n_i, n_j at time t_m is lower than γ:

$$d(n_1, n_2) \leq \gamma \rightarrow e_{i,j} \in E_m$$

We adopt the Great-circle distance [24] and we initially set γ to an infinite value so that to build the complete graph of the AUSN. In this way, every active float can theoretically communicate with every other active float.

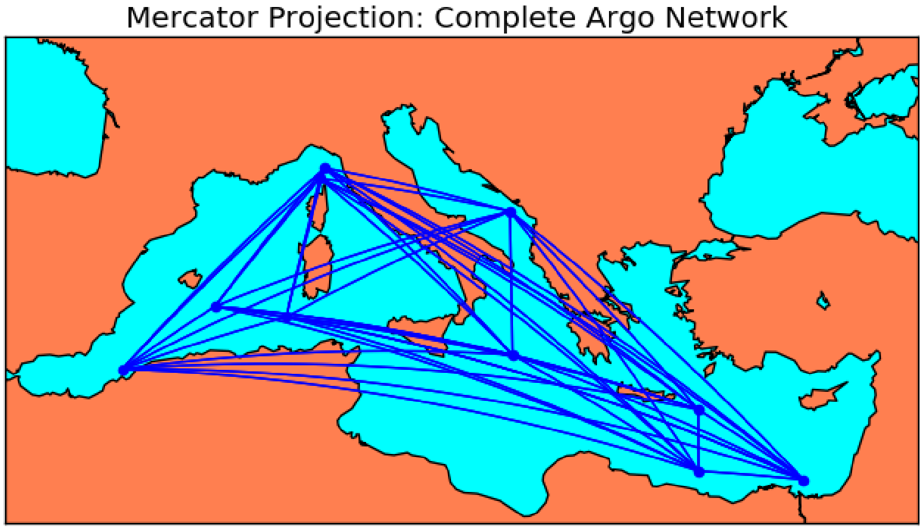


Figure 13 The Argo Underwater Sensor Network as a complete graph

Figure 13 shows the Argo Underwater Sensor Network represented as a complete graph. Every point representing an active float is connected to every other float in the network with a direct link.

Please note that in Section 3.3.4**,** we describe how we set the transmission range γ so that to reproduce a realistic scenario.

As observed in Figure 10, most of the floats do not report their position at regular intervals. Therefore, in order to cope with the partial absence of float's positions, we apply an interpolation strategy so that to re-build the trajectories of the floats along the time. We interpolate the position of the Argo float *F* at time windows t_i, only if *F* is active at the immediately previous and immediately after time t_i. More specifically, we interpolate the position of F at time $t_i \in \{t_1, t_2, \dots t_{370}\}$ only if our dataset provides the position of F at time t_j, $t_j \leq t_i$ and if the dataset provides the position of F at time t_k, $t_k \geq t_i$. Figure 14 shows the interpolation process in a graphical way.

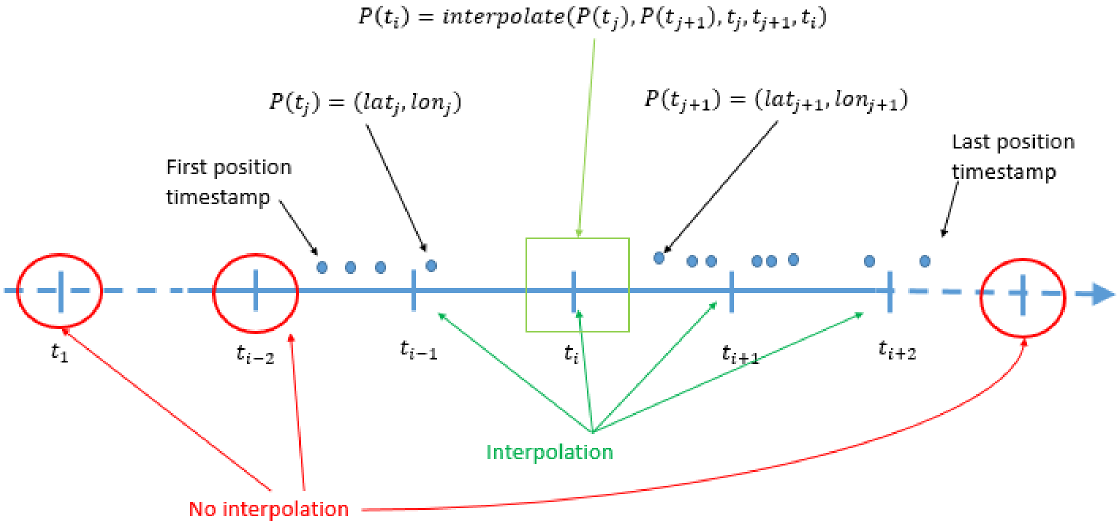

$P(t_i) = interpolate(P(t_j), P(t_{j+1}), t_j, t_{j+1}, t_i)$
$P(t_j) = (lat_j, lon_j)$
$P(t_{j+1}) = (lat_{j+1}, lon_{j+1})$
First position timestamp
Last position timestamp
t_1
t_{i-2}
t_{i-1}
t_i
t_{i+1}
t_{i+2}
Interpolation
No interpolation

Figure 14 Interpolation Schema

In the beginning of Section 3.3 we stated that inactive floats are also the ones such that: being t_j and t_{j+1} two timestamps associated to positions of F in our dataset, such that t_j and t_{j+1} are the closest timestamps to t_i and $t_j \leq t_i \leq t_{j+1}$:

$$\left(t_{j+1} - t_j\right) \leq median(lifecycle_{duration}) + 1\ day$$

We made this choice because if an Argo float does not communicate its position for a long period, the interpolation process could not approximate accurately the float's movements.

To compute the median life cycle duration of an Argo float F, we computed the timespans between every subsequent pair of timestamps (t_j, t_{j+1}). Then we excluded from the median calculation the timespans whose length was lower than 1 day, because we assumed that such timespans are associated to two subsequent communications while F is on the surface.

Figure 14 shows the behaviour of the interpolation process while dealing with a period characterized by an absence of recorded position for an Argo float.

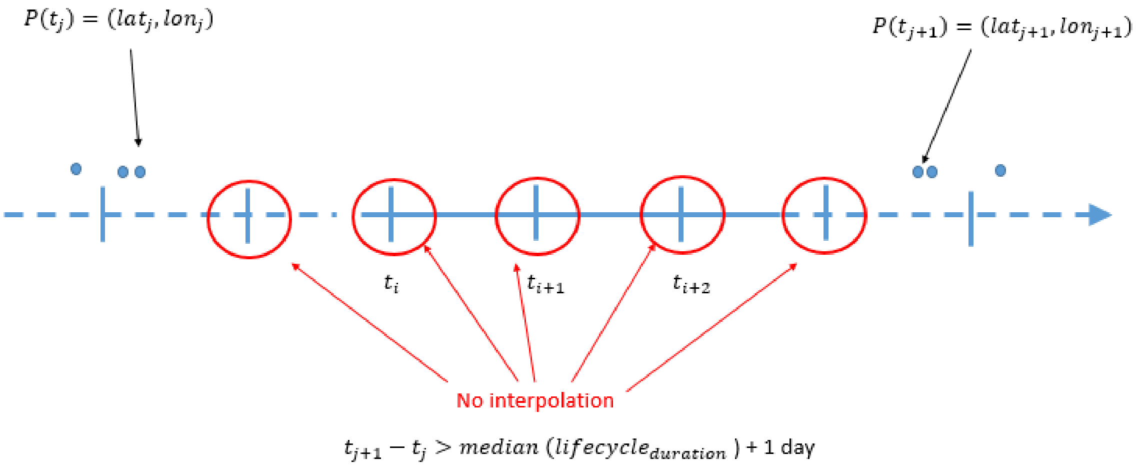

$P(t_j) = (lat_j, lon_j)$
$P(t_{j+1}) = (lat_{j+1}, lon_{j+1})$
t_i
t_{i+1}
t_{i+2}
No interpolation
$t_{j+1} - t_j > median\ (lifecycle_{duration}\ ) + 1$ day

Figure 15 Interpolation Schema 2

To interpolate the positions of the active Argo floats in every time instant $t_1, t_2, \dots, t_{370}$, we use the formula for computing the *Intermediate points on great circle*[10]. This formula, takes as parameters two points P_1 and P_2 represented by their geographical coordinates, latitude and longitude (P_1 = (lat1, lon1) and P_2 = (lat2, lon2)), and a fraction *f* of the distance *d* between them. Than it computes the coordinates of the point P_f that sits at distance *f*d* from P_1 on the great circle that connects P_1 with P_2. For instance, if *f*=0 than P_f is equal to P_1, if *f*=1 than P_f is equal to P_2, else if *f=0*.5 than P_f is equal to the point in the middle of the great circle that connects P_1 and P_2. The formula works only if P_1 and P_2 are not antipodal, because in that case, the route is undefined.

[10] http://williams.best.vwh.net/avform.htm#Intermediate

To compute the interpolation of the position of an Argo float *F* in a time instant *t_i*, we need to know the geographical coordinates of the two positions $P_{\mathrm{F}}(t_j)$ and $P_{\mathrm{F}}(t_{j+1})$ of *F* whose timestamps t_j and t_{j+1} are closest to t_i and $t_j \leq t_i \leq t_{j+1}$.

Being the geographical coordinates latitude and longitude of $P_F(t_j)$ and $P_F(t_{j+1})$ respectively $(lat1, lon1)$ and $(lat2, lon2)$, the great-circle distance between $P_F(t_j)$ and $P_F(t_{j+1})$ is:

$$d = 2 * \operatorname{asin}\sqrt{(\ \sin\left(\frac{lat1 -\ lat2}{2}\right))^{\ 2} + \cos(lat1) * \cos(lat2) *\ \left(\sin\left(\frac{lon1 - lon2}{2}\right)\right)^2}$$

With *d* representing the great circle distance in radians between $P_F(t_j)$ and $P_F(t_{j+1})$

$$f = \frac{t_i - t_j}{t_{j+1} - t_j}$$

f represents the fraction of the distance between $P_F(t_j)$ and $P_F(t_{j+1})$ of the point $P_F(t_i)$

$$A = \sin\frac{(1 - f) * d}{\sin(d)}$$

$$B = \frac{\sin(f * d)}{\sin(d)}$$

$$x = A * \cos(lat1) * \cos(lon1) + B * \cos(lat2) * \cos(lon2)$$

$$y = A * \cos(lat1) * \sin(lon1) + B * cos(lat2) * \cos(lon2)$$

$$z = A * \sin(lat1)$$

$$lat = atan2\ \left(z, \sqrt{x^2 + y^2}\ \right)$$

$$lon = atan2\ (y, x)$$

The coordinates we obtain are respectively the latitude and longitude of the point $P_F(t_i)$, that is the location of the Argo float *F* in the time instant t_i.

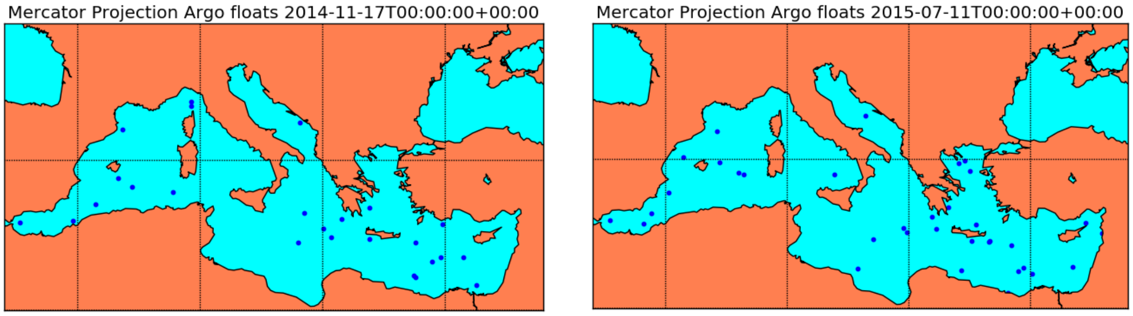

Mercator Projection Argo floats 2014-11-17T00:00:00+00:00
Mercator Projection Argo floats 2015-07-11T00:00:00+00:00

Figure 16 Mercator projections of two snapshots

Figure 16 shows the positions of the Argo floats as blue circles with a Mercator Projection. The maps are obtained with the positions of November 17th 2014 and July 11th 2015 respectively.

Given the positions of the floats in every time window, we can compute the weight of the edges of the time-varying graphs by measuring the distance *d* between nodes along the time.

Because of the simplification we made at the beginning of Section 3.3, we leave out the depth parameter while computing the distance between two Argo floats, and we use only the geographical coordinates to compute it. In particular we use the Great Circle distance[24] to represent the distance between two Argo floats. To compute the great circle distance we use the python library *geopy*[11].

Given the complete graph build with the interpolation strategy, we analyse some metrics of the network such as the active floats, the float's trajectories (Section 543.3.3).

Active Floats

Figure 17 Number of active floats for each snapshotFigure 17 shows the number of active floats for each of the 370 snapshots created.

[11] https://geopy.readthedocs.io/en/1.11.0/

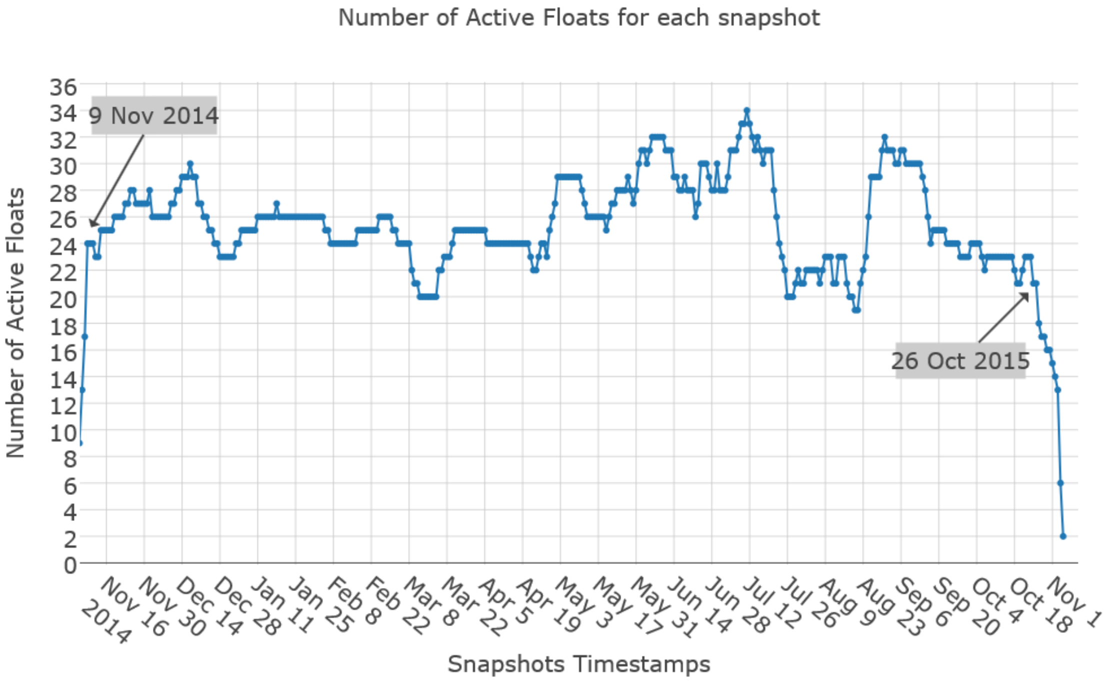


Figure 17 Number of active floats for each snapshot

On the x-axis of Figure 17 we report the timestamps of the snapshots, while on the y-axis the number of active floats. As we can observe, before November 9th 2014 and after October 26th 2015, the number of active floats drastically decreases. This behaviour is caused by the lack of the positions needed by the interpolation strategy before the beginning and after the end of the observation period.

In the following steps of our analysis, we shrink the observation period from 370 to 352 days, so that to exclude from our analysis the snapshots whose timestamps are prior to November 9th 2014 or after October 26th 2015.

After the restriction of the observation period, we can observe that the number of floats that are active at the same time ranges from 19 to 34, with an average of 26.

3.3.3 Drifting Analysis

We also investigate the mobility patterns followed by the floats while drifting in the Mediterranean Sea. In particular, we want to understand if the Argo floats travel long distances or if they tend to wander in a small area, i.e. when they are trapped in vortexes. To analyse the float's trajectories, we introduce the following notation: $D\left(P_{F1}(t_i), P_{F2}(t_j)\right)$ is the great – circle distance between the positions $P_{F1}(t_i)$ and $P_{F2}(t_j)$. We computed two values for each float *F*:

- *Sum of distances* $d_{\Sigma}(F)$: represents the sum of the distances covered by *F* between a snapshot t_i and the following t_{i+1}:

$$d_{\Sigma}(F) = \sum_{i=0}^{n-1} D\big(P_F(t_i), P_F(t_{i+1})\big)$$

With t_0 and t_n corresponding respectively to the first interpolated position of *F* in our snapshots and t_n to the last interpolated position associated to *F* in our snapshots.

- *Overall distance* $d_{overall}(F)$: represents the distance between the positions of *F* in t_0 and t_n.

$$d_{overall(F)} = D\big(P_F(t_0), P_F(t_n)\big)$$

In Section 3.2.2, we noticed that some floats do not communicate their positions to the satellite for long periods, thus we decided to do not interpolate their positions during the periods where they have no associated position.

Considering that the computation of the sum of distances and the overall distance of these floats may lead to a poor approximation of the distances they actually traveled, we excluded them from the analysis in this paragraph. We compute $d_{\Sigma}(F)$ and $d_{\text{overall}}(F)$ only for the Argo floats that have no associated positions only for at most 10 snapshots.

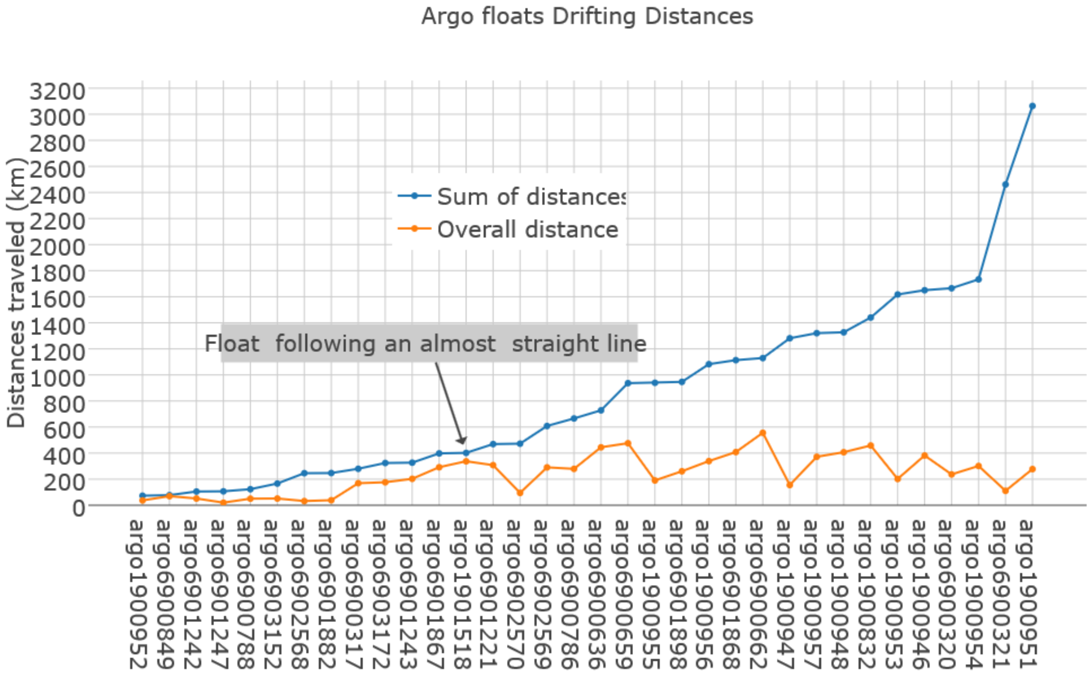
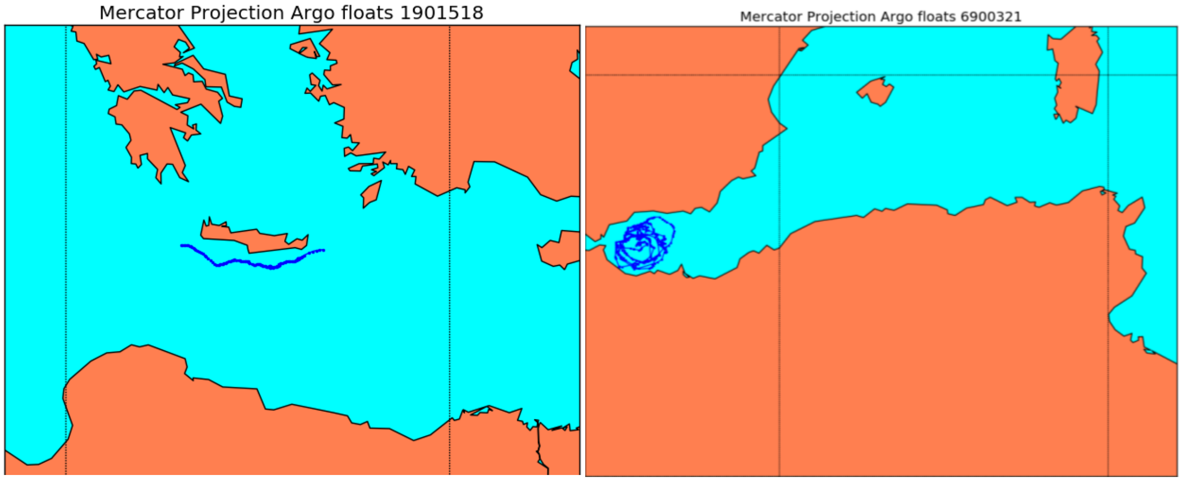


Figure 18Argo floats distance covered

In Figure 18, we show the sum of distances and the overall distance of the 28 floats which are active for almost the entire time period corresponding to their presence in the network.

On the x-axis are listed the identifiers of the Argo floats we used. The blue plot represents the Sum of the distances covered by every float, while the orange plot represents the overall distance. We sorted the Argo float on the x-axis by the sum of distances (blue plot) to make the chart more readable. The distances on the y-axis are expressed in km.

By comparing the two line plots shown in Figure 18, we can better understand the floats' trajectory. If the overall distance and the sum of distances for a float F_1 are very close, then we consider that probably, F_1 moved almost along a straight line. Conversely, if the sum of distances is much larger than the overall distance for the float F_2, then probably F_2 moved in circle or moved “back and forth”.


Mercator Projection Argo floats 1901518
Mercator Projection Argo floats 6900321

(a) (b)

Figure 19 Drifting trajectories (a) straight-line trajectory (b) looping trajectory

As for example, consider the floats 1901518 and 6900321 whose trajectories are reported in Figure 18(a) and Figure 18(b). For what concerns, the float 1901518, the sum of distances and the overall distance are very similar, specifically 402 km and 337 km respectively. Moreover, as shown in Figure 19 (a), the float 1901518 follows almost a straight-line trajectory. Differently, the Argo float 6900321 shows a remarkable difference between the sum of distances and the overall distance, specifically 2461 km and 110 km respectively. Figure 19(b) shows the looping trajectory of float 6900321, wandering in the same small area for its entire active time.

We can see from Figure 18 that even if there are some floats boasting a high sum of distances, the overall distance covered by them is at most 556 km, in the timespan of a year. In addition, the average overall distance is 237 km, highlighting the fact that the Argo floats drifting in the Mediterranean Sea, tend to wander always in the same area.

Concerning the sum of distances, we can observe that some floats have very high sum of distances. For instance, the Argo float 1900951 boast the maximum sum of distances reaching 3064 km, but travels only 300 km as overall distance. The average sum of distances is also high comparing to the overall distance, being it close to 935 km.

3.3.4 Minimum Spanning Tree

The transmission range γ strongly affects the connectivity of the time-varying graph shown in Figure 13. To this end, we analyse the Minimum Spanning Trees (MST) of the time-varying graph built in Section 3.3.2 in order to define some admissible ranges of the γ value.

Please note that we exclude the inactive floats from our analysis. This means that the graphs G_i that are part of our time-varying graph, are composed only by the floats that are active at time t_i.

The spanning tree of a connected undirected graph *G* is a tree that is a subgraph of *G* and contains all the vertices of *G*. The Minimum Spanning Tree (MST) of a connected weighted graph *G* is a spanning tree whose sum of the weights is minimum among all the possible spanning trees of *G*.

We compute the MST for each snapshots of the time-varying graph. The MST is obtained with the python network library NetworkX[12], which is based on the Kruskal algorithm.

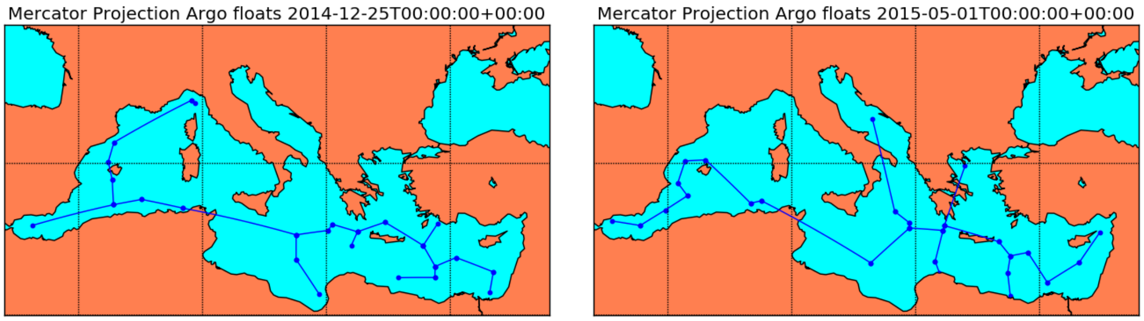

Mercator Projection Argo floats 2014-12-25T00:00:00+00:00
Mercator Projection Argo floats 2015-05-01T00:00:00+00:00

Figure 20 Mercator projections of the MST build on two snapshots

[12] https://networkx.github.io/documentation/networkx-.10/reference/generated/networkx.algorithms.mst.minimum_spanning_tree.html#networkx.algorithms.mst.minimum_spanning_tree

Figure 19 shows two MSTs at different snapshots (with γ= ∞). Note that we are not considering the presence of landmasses between the floats. Therefore, there are cases in which we connect floats even if among them there exist islands or landmasses.

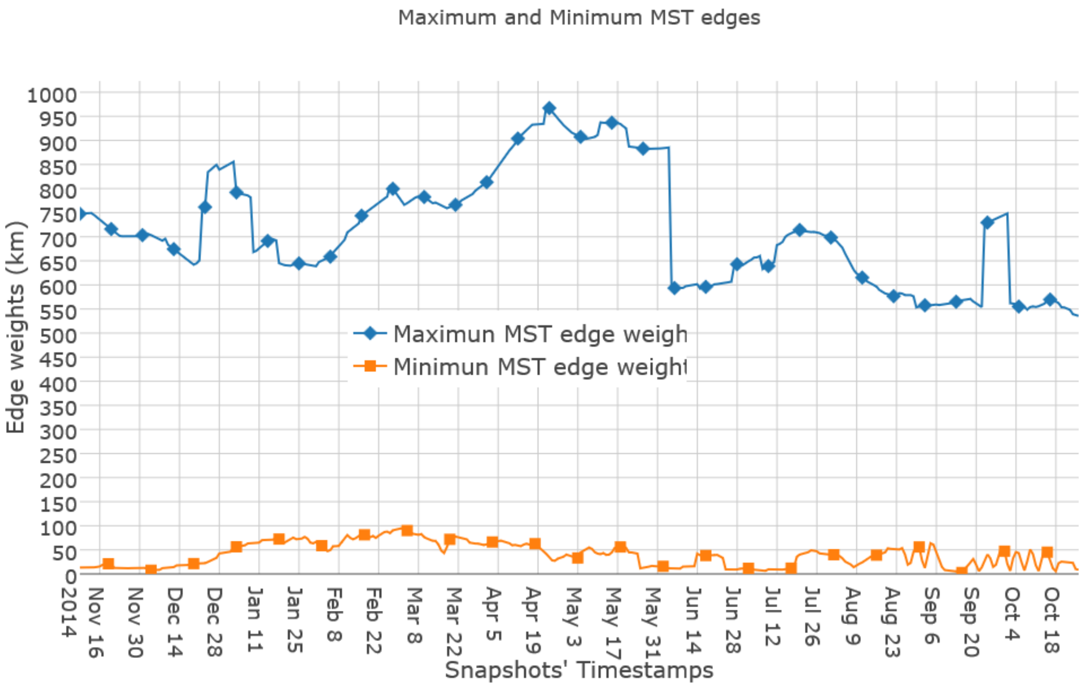


Figure 21 Maximum and Minimum MST edges' weights variation

By analysing the weights of the edges of the MSTs, we can investigate admissible values of γ. In particular, we focus on maximum and minimum values of the weights of the edges of the MSTs obtained. Figure 21 shows the variation of such values. The blue line shows the weights of the maximum edge of every MST, while the orange line shows the weights of the minimum edge of every MST. The distances on the y-axis is in km. On the x-axis, we show the timestamps of the snapshots.

The value of the *minimum edge* varies:

- From a minimum of 3.02 km
- To a maximum of 94.7 km
- With an average of 39.8 km

- And a standard deviation of 24.3 km

The value of the *maximum edge* varies:

- From a minimum of 535.6 km
- To a maximum of 972 km
- With an average of 715.5 km
- And a standard deviation of 117.9 km

We can see that the weights of the minimum and maximum MST edges are highly variable throughout the course of the observation period.

If we set γ ≤3.02 km, than the resulting network is always completely disconnected, as no floats are able to communicate with any other float. If we choose γ ≥972 km then the resulting network is always completely connected. This means that we must choose admissible values for γ ranging between 3.02 km and 972 km. In particular, we decide to use the *Average MST Percentiles* as possible values for γ. We computed the 1st, 5th, 10th, 15th 20th …, 95th percentiles of every MST. Then we computed the average of these values of the percentiles, to obtain the averages values reported in Table 2.

1st 46.0013596144	**5th** 68.6516109409	**10th** 90.9610965281	**15th** 112.586317809
20th 128.512319728	**25th** 144.197619873	**30th** 160.471817985	**35th** 175.236956497
40th 189.635270626	**45th** 201.584109562	**50th** 213.588458034	**55th** 225.369952526
60th 240.310052578	**65th** 260.403514916	**70th** 284.590331258	**75th** 312.604947651
80th 346.922661761	**85th** 390.835494302	**90th** 472.002134206	**95th** 593.752577104

Table 2 Average MST edges' weights percentiles expressed in kilometres

By using values of γ reported in Table 2, we can determine, on average, the percentage of the floats in the snapshots that are connected. As for example, by setting γ =46km, the 1% of the floats are connected, while with γ = 390.8 km, the 85% of the floats are connected.

It is worth to notice that most of the transmission ranges reported in Table 2 can be considered only from a theoretical point of view. In fact, hardware and software constraints of the acoustic medium for underwater communication (Section 2.1.5) reduce the communication distance among floats.

In the rest of this chapter, we consider the time-varying graph obtained by selecting different values of γ. We refer to $G_i(\gamma)$ as the snapshot of the graph at time t_i with the transmission range γ.

3.3.5 Average Degree Analysis

We first compute the average degree for each of values of γ in Table 2. The average degree of the graph *G = (N, E)*, is the number of edges linking *n* with any other nodes:

$$degree(n) = |\,\{(n,n')|(n,n') \in E\}\,|$$

$$degree_{avg}(G) = \sum_{n \in N} \frac{degree(n)}{|N|}$$

With | *S* | corresponding to the cardinality of set *S*.

We computed the average degree for each transmission range γ in Table 2 and for each $G_i(\gamma)$.

Figure 22 shows the average degrees obtained with a Box Plot graph. On the x-axis, we report the percentile as well as the resulting γ value (in km) and on the y-axis, we repot the average degree.

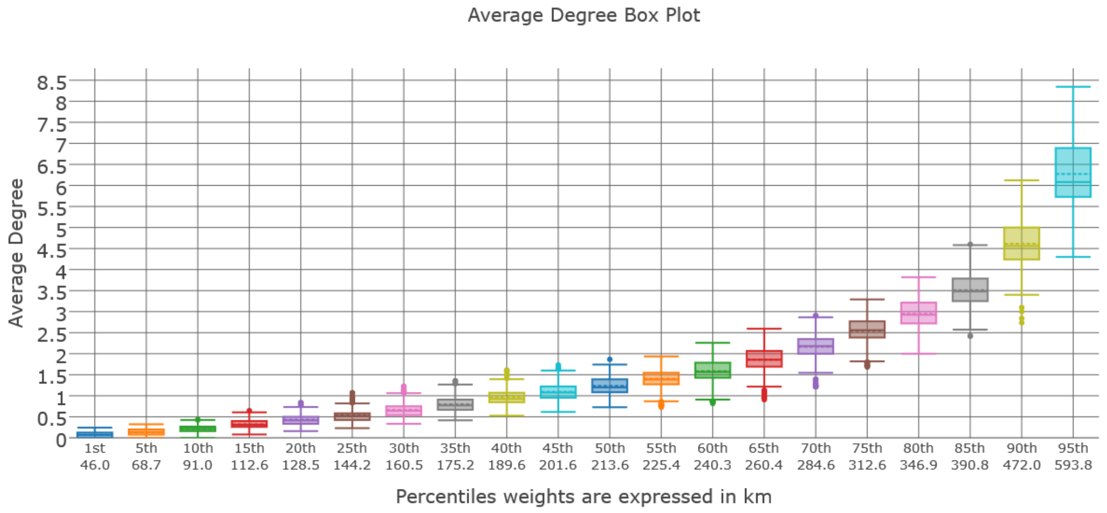


Figure 22 Average Degree Box Plot

For each box the whiskers represent the maximum and the minimum without considering the outliers, the points above or below the whiskers represent the outliers, the top and the bottom of the box represent respectively the third and first quartile; the full line at the centre of the box represents the median, while the dashed line represents the average.

As we can see in Figure 22, the average degree is very low even for long transmission ranges. In particular, the mean and median average degree of every $G_i(\gamma)$ are higher than 1 only with a transmission range $\gamma \geq 201.6$ km (45th percentile). Also the mean and median average degree of $G_i(\gamma)$ are higher than 2 only with a transmission range $\gamma \geq 284.6$ km (70th percentile) and are higher than 3 only with $\gamma \geq 390.8$ km (85th percentile).

Lastly, Figure 22 shows that even when using a huge transmission range, such as $\gamma = 593.8$ km, corresponding to the 95th percentile, the average degree is rather low. In particular, the mean and median average degree are for each $G_i(593.8\text{ km})$ are between 6 and 6.5 and the maximum average degree is slightly less than 8.5.

3.3.6 Density Analysis

We then analyse the Density of the graphs $G_i(\gamma)$, so that to have a more in-depth understanding of the structure and connectivity of the underwater network. The density represents the ratio between the number of edges in *G* and the number of edges that *G* would contain if it was a complete graph.

The density of an undirected simple graph *G = (N, E)* is defined as:

$$density(G) = \frac{2 * |E|}{|N| * (|N| - 1)}$$

With |*N*| equal to the number of nodes in *G* and |*E*| equal to the number of edges in *G*.

For each simple graph *G*, the density value is between 0 and 1. If *G* is complete, than the density is equal to 1, if *G* is composed only by isolated nodes (i.e. |*E*| = ∅), than the density is equal to 0.

We computed the density of our Temporal Networks, for each transmission range γ reported in Table 2 and for each $G_i(\gamma)$.

As we did in the previous Section, we grouped the computed values by the transmission range γ, and created a box for each group.

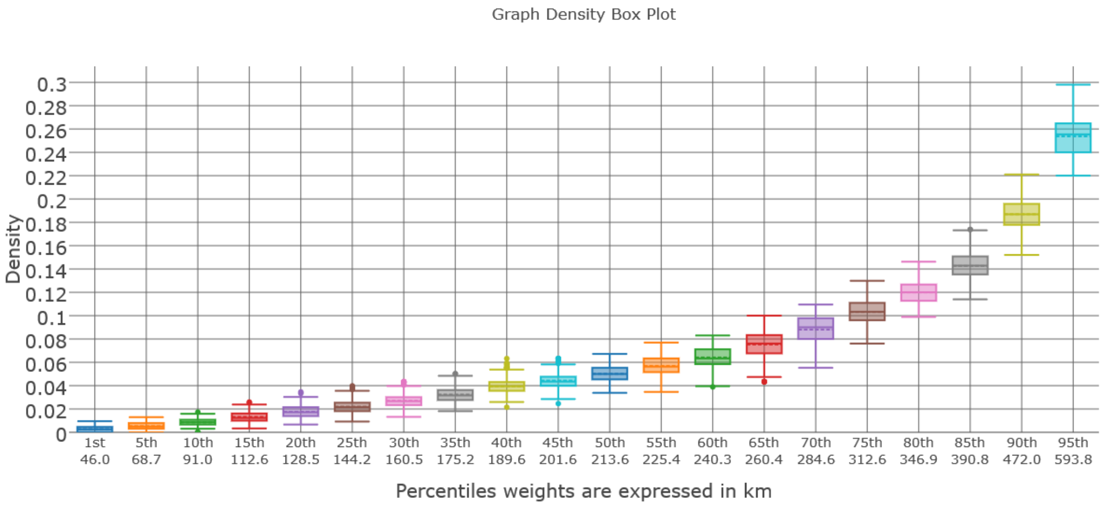


Figure 23 Graph Density Box Plot

The Box Plot contained in Figure 23, shows a similar behaviour to the one in Figure 22. By observing both plots, we can deduce that the Argo Underwater Sensor Network is a strongly disconnected network, even for sizable transmission ranges like 200 km. In particular, using the values of the 1st and 5th percentiles as transmission ranges, we can see that the density is close to zero. This means that choosing a transmission range < 70 km, the Argo Network is almost completely disconnected.

Furthermore, we can observe that density is below 0.1 for almost every Temporal Network $G_i(\gamma)$ having transmission range $\gamma < 260.4$ km, equal to the value of the 65th percentile. Only using a transmission range equal or greater than 312.6 km (75th percentile), we can record a mean and median density for $G_i(\gamma)$ slightly larger than 0.1. This means that even for large transmission ranges (near 300 km), the density of the network is very low.

Using the value of the 95th percentile as transmission range (593.8 km), the maximum density recorded is close to 0.3, but the mean and median of the density for $G_i(593.8\ km)$ is close to 0.25.

3.3.7 Connected Components Analysis

A connected component in an undirected graph *G*, is a subgraph *G'* in which every pair of vertices is connected by a path, and the vertices of *G'* have no links with the other vertices of the G. The analysis of the connected components, along with the analysis of the density and average degree, reveal more details on the structure of the Argo Underwater Sensor Network.

We first we compute the average connected component size for every $G_i(\gamma)$ and for every transmission range γ contained in Table 2, then we analyse the variation of the number of connected components in $G_{[0,352]}(\gamma)$ and lastly we compute the variation of the dimension of the three largest connected components.

To compute the average connected components for each $G_i(\gamma)$ we exploited the python library NetworkX. For each transmission range γ reported in Table 2, we computed the average cardinality of the connected components in every $G_i(\gamma)$.

As we did for the average degree and density, we generated a Box Plot for the average connected components size.

By observing Figure 24, we can see that for using the value of the percentiles 1st and 5th , the average connected components size is very close to 1. This confirms what we already noticed observing Figure 22 and Figure 23. Using a transmission range γ < 70 km the Argo Underwater Sensor Network is composed almost entirely by isolated nodes.

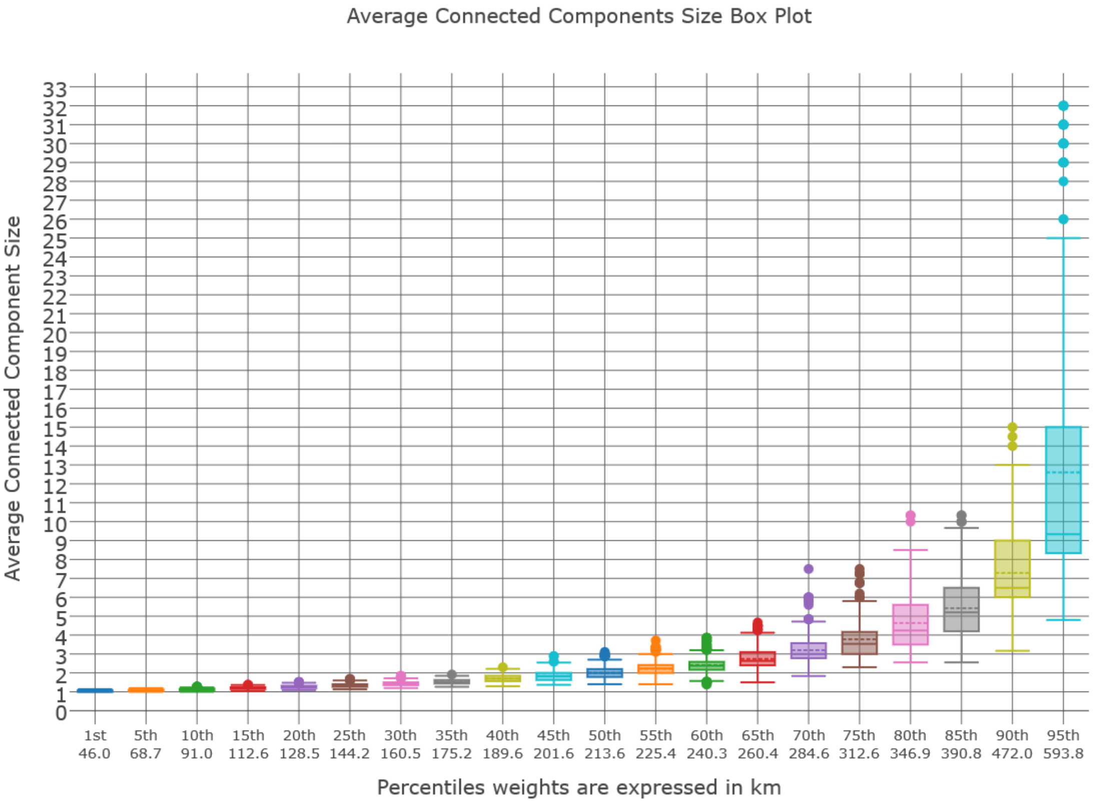


Figure 24 Average Connected Component Size Box Plot

The mean and median average connected components size for $G_i(\gamma)$ are close to 2 only for transmission ranges $\gamma \geq 201.6\ km$ equal to the value of the 45th percentile and are above 3 only using γ equal or greater than the value of the 70th percentile (284.6 km).

By observing the box plots corresponding to the percentiles corresponding to high transmission ranges, we can see a high variance in the average connected components sizes. In particular observing the box plot corresponding to the 95th percentile, we can see that the average connected components size varies from a minimum of 5 to a maximum of 25. Furthermore, we can observe that there are some outliers reaching 30 as average connected components size, which is close to the maximum number of floats that are active at the same time, which we discussed in Section 3.3.2.

We restrict our analysis to values γ corresponding to 10th, 55th, 80th, 85th and 95th percentiles the following reasons:

10th	45th	70th	85th	95th
90.96 km	201.58 km	284.59 km	390.84 km	593.75 km

Table 3 Chosen percentiles

- the 10th percentile: as we found out in the previous sections, by using a transmit range γ less than or equal the value of the 10th percentile, the connectivity between the nodes in the Argo network is almost non-existent;
- the 45th percentile: using a transmit range equal or greater than its value, the mean and median average degree of the graphs $G_i(\gamma)$ is above 1;
- the 70th percentile: using a transmit range equal or greater than its value, the mean and median average degree of the graphs $G_i(\gamma)$ is above 2;
- the 85th: using a transmit range equal or greater than its value, the mean and median average degree of the graphs $G_i(\gamma)$ is above 3;
- the 95th percentile as an upper bound to the transmit range.

Variation of the Number of Connected Components

In this Section, we analyse how varies the number of connected components during our observation period.

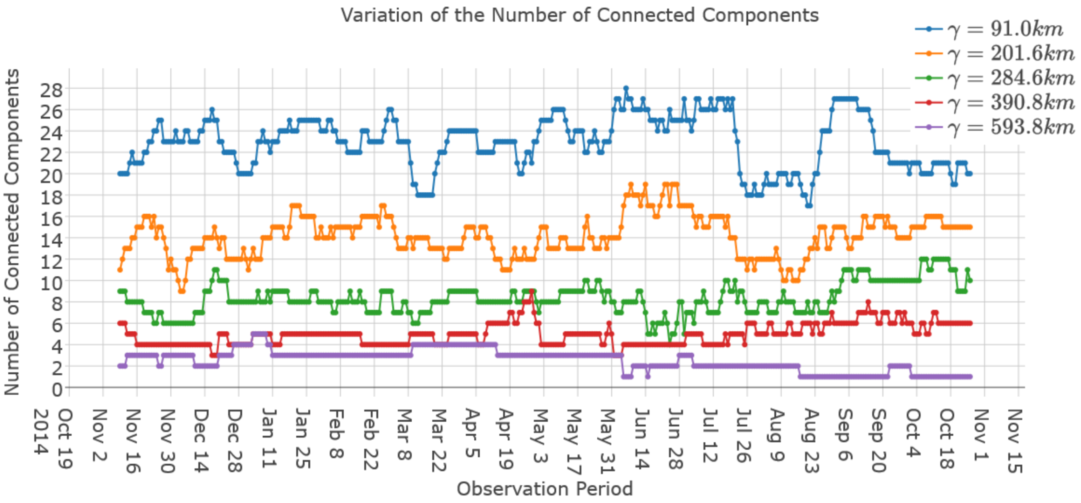


Figure 25 Variation of the Number of Connected Components

Figure 25 shows a line plot for every transmission range used. On the x-axis are listed the time instants $t_i \in \{t_1, t_2, \dots, t_{352}\}$ that are part of our observation period and represent the snapshots of our Temporal Networks. On the y-axis are listed the number of connected components. The legend on the right side of the chart shows the percentiles and their values, expressed in kilometres.

By observing the line plot corresponding to the 10th percentile, we can see that the number of connected components is very close to the size of the Argo Sensor Network. In particular, by comparing the line corresponding to the number of connected components using the 10th percentile as transmission range to the plot representing the variation of the number of active floats during the observation period, we see that both plots have a similar shape (Figure 26). This means that for every instant of time in our observation period the number of connected components is very similar to the number of active floats, or equivalently the network is composed mainly by isolated floats.

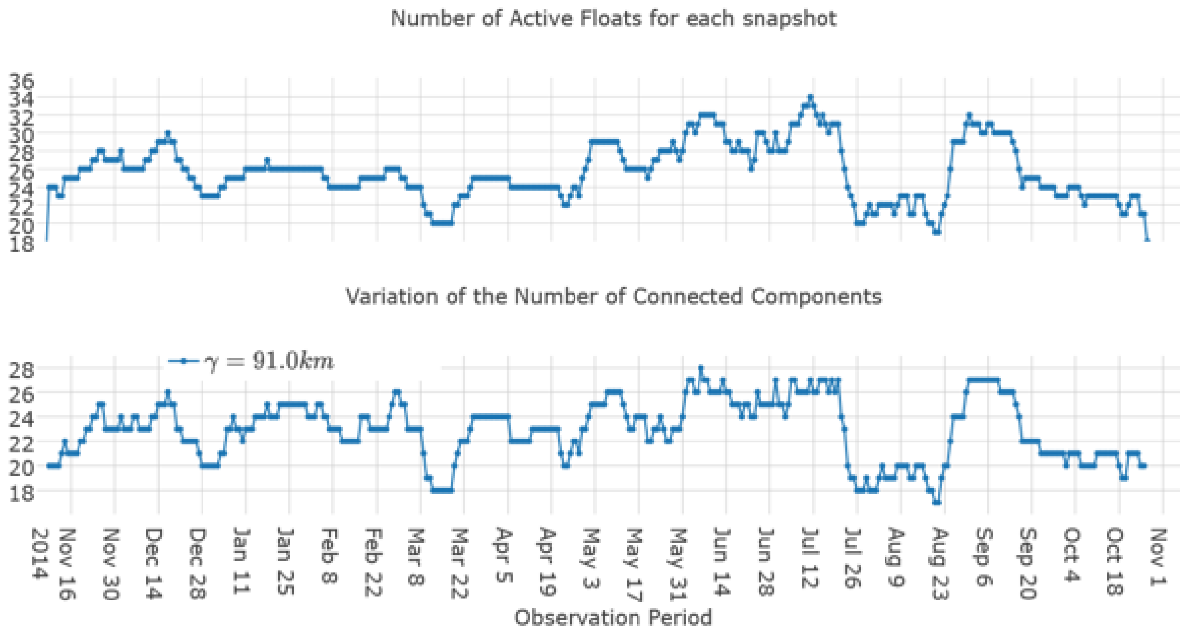


Figure 26 Comparison between the number of connected components using the value of the 10th percentile as transmission range and the number of active floats

Concerning the 45th percentile: the number of connected components for a transmission range γ = 201.6 km, is highly variable, ranging from 9 to 19. The average number of connected components is 14.

Focusing on the line plots corresponding to the 70th and 85th percentiles, we can observe that they are less variable than the one obtained using the 45th percentile as transmission range. In the first case, the number of connected components for $G_i(284.6\ km)$ varies from 4 to 12, but the number of connected components for most of the time is between 6 and 10, with an average of 8. The number of connected components for $G_i(390.8\ km)$ varies from 3 to 9, with an average of 5. By using the value of the 95th percentile as transmit range, we observe that for some snapshots the number of connected components is equal to 1. This means that for some snapshots the network is almost connected.

Largest Connected Components

We then analyse the variation of the size of the three biggest connected components (CC) of each graph $G_i(\gamma)$ with values of γ shown in table Table 3.

Figure 27 shows that the size of the three biggest CCs with γ = 91.0 km ranges from 1 to 3. Moreover, the cardinality of the CCs is always greater than 1. This means that the network is never fully disconnected.

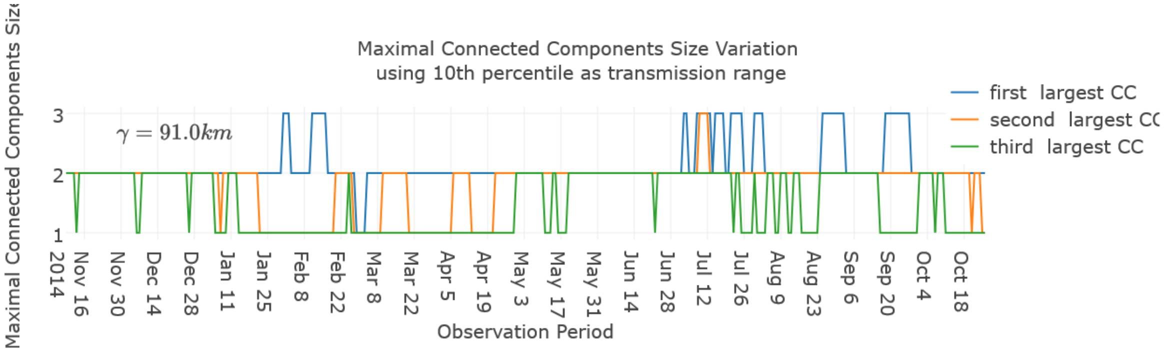


Figure 27 Largest Connected Components size variation using γ = 91.0 km

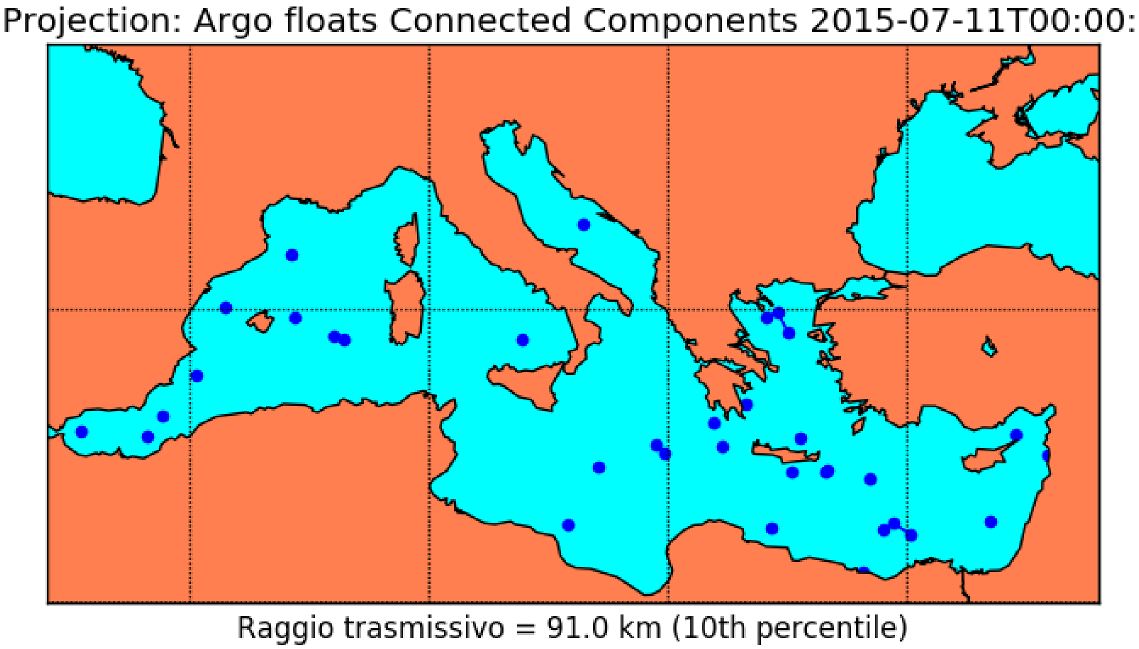

Projection: Argo floats Connected Components 2015-07-11T00:00:
Raggio trasmissivo = 91.0 km (10th percentile)

Figure 28 Argo Underwater sensor Network using 10th percentile as transmission range.

Figure 28 shows a maps representing a snapshots of the Argo Underwater Sensor Network with γ = 91.0 km. The blue points represent the active Argo floats, while each blue line links two nodes whose distance is smaller than the transmit range γ. We can see that most of the floats are isolated, as they cannot communicate with other floats, but there are a few small group of 2 or 3 nodes that are linked.

Figure 29 shows the size of the three biggest CCs with γ = 201.6 km. From the figure, we can see that, differently from the previous case, the cardinality of the three biggest connected components has always size greater than 1. Furthermore, the size of the biggest connected component ranges from 4 to 12. The average size of the biggest connected component is close to 6, that is nearly ¼ of the average number of active floats at the same time.

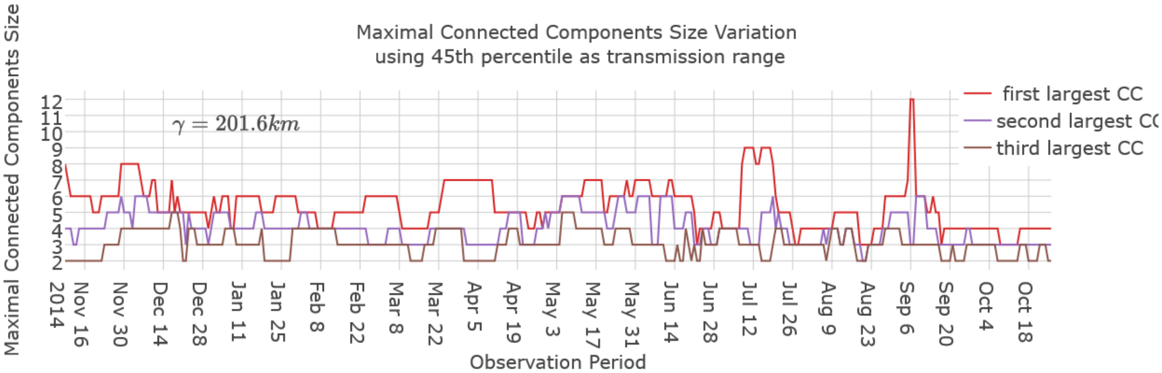


Figure 29 Largest Connected Components size variation using γ = 201.6 km

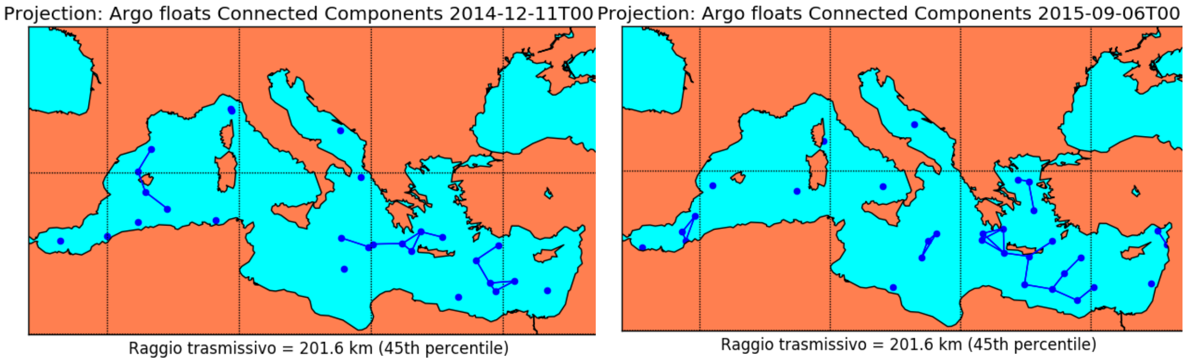

Projection: Argo floats Connected Components 2014-12-11T00
Projection: Argo floats Connected Components 2015-09-06T00
Raggio trasmissivo = 201.6 km (45th percentile)
Raggio trasmissivo = 201.6 km (45th percentile)

Figure 30 Argo Underwater sensor Network using 45th percentile as transmission range.

In Figure 30, we can observe two maps showing the structure of the Argo Underwater Sensor Network with the hypothesis that the floats are equipped with an acoustic modem with transmission range equal to 201.6 km. The map corresponding to the snapshot with $t_i = 00:00:00\ 11th\ December\ 2014$ shows many isolated floats, but few connected components of with size ranging from 4 to 7. The other map has the same timestamp $t_i = 00:00:00\ 6th\ September\ 2015$ of the peak recorded for the largest connected component size in Figure 29. At time $t_i = 00:00:00\ 6th\ September\ 2015$ the maximal connected component reaches size equal to 12. Figure 30 shows that the floats composing the maximal connected component drifting south of Crete.

Figure 31 shows that the size of the three biggest CCs with γ = 284.6 km. The size of the maximal connected component is much larger than the other two largest connected component for most of the observation period. Although, the size of the maximal connected component is highly variable, ranging from 6 to 21. For most of the time, the maximal connected component has size that is close to half the maximum number of active floats at the same time.

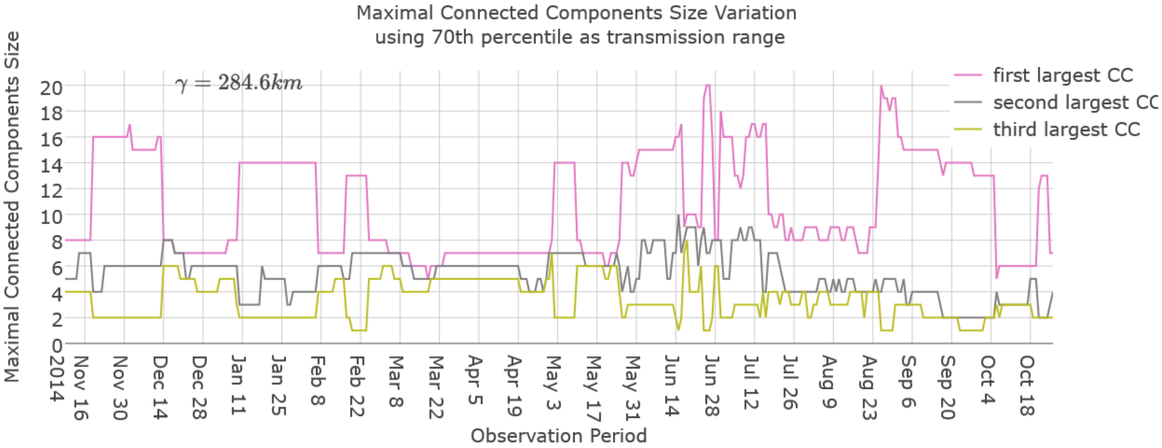


Figure 31 Largest Connected Components size variation using γ = 284.6 km

In Figure 31, we can see that setting γ = 284.6 km as transmission range for the Argo Underwater Sensor Network, the size of the maximal connected component is much larger than the other two largest Connected component for most of the observation period. Although, the size of the maximal connected component is highly variable, ranging from 6 to 21. For most of the time, the maximal connected component has size that is close to half the maximum number of active floats at the same time.

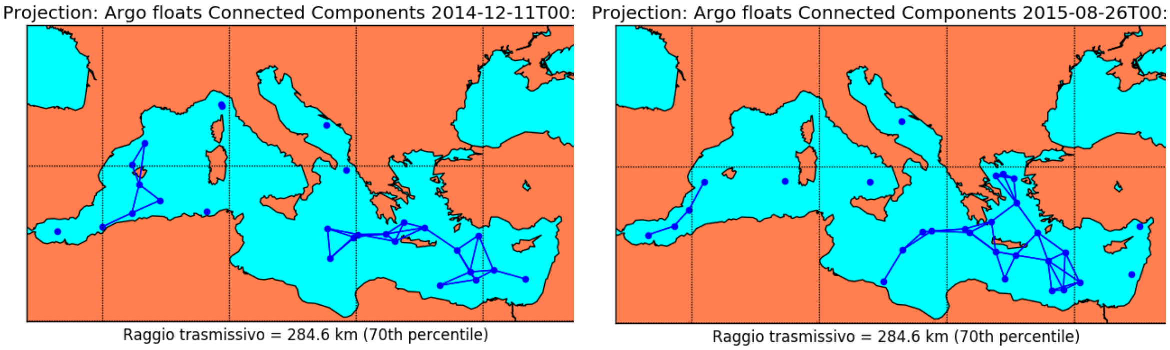

Projection: Argo floats Connected Components 2014-12-11T00:
Raggio trasmissivo = 284.6 km (70th percentile)
Projection: Argo floats Connected Components 2015-08-26T00:
Raggio trasmissivo = 284.6 km (70th percentile)

Figure 32 Argo Underwater sensor Network using 70th percentile as transmission range.

Figure 32 shows two maps of our network using the 70th percentile as transmission range. In both maps, there are few isolated nodes and a massive connected component that include most of the floats in the network.

In Figure 33, we can observe that using the value of the 85th percentile as transmission range, the largest connected components of the Argo Underwater Sensor Networks have a similar behaviour to the one shown in the previous sub-paragraph. Moreover, the maximal connected component contains the majority of the Argo floats drifting in the Mediterranean Sea for almost the entire observation period.

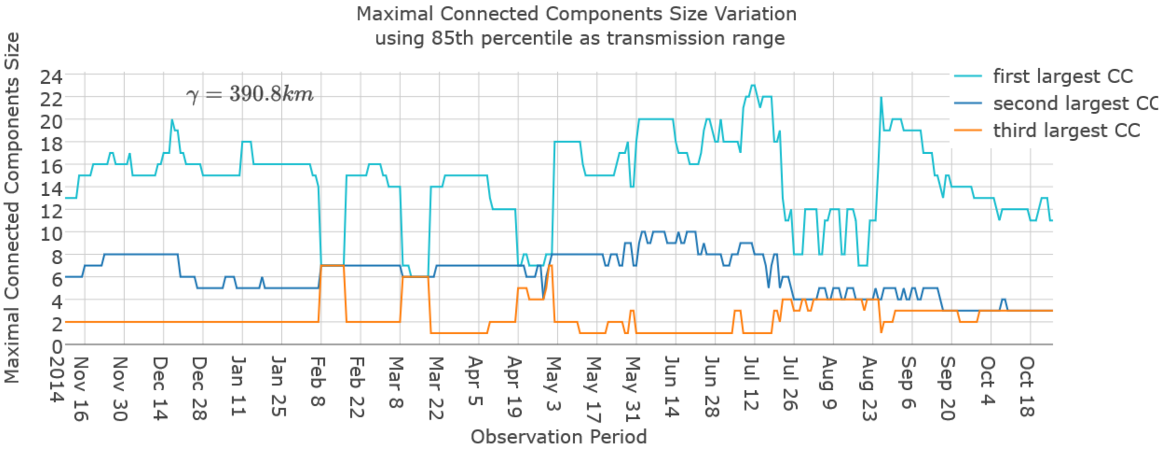


Figure 33Largest Connected Components size variation using γ = 390.8 km

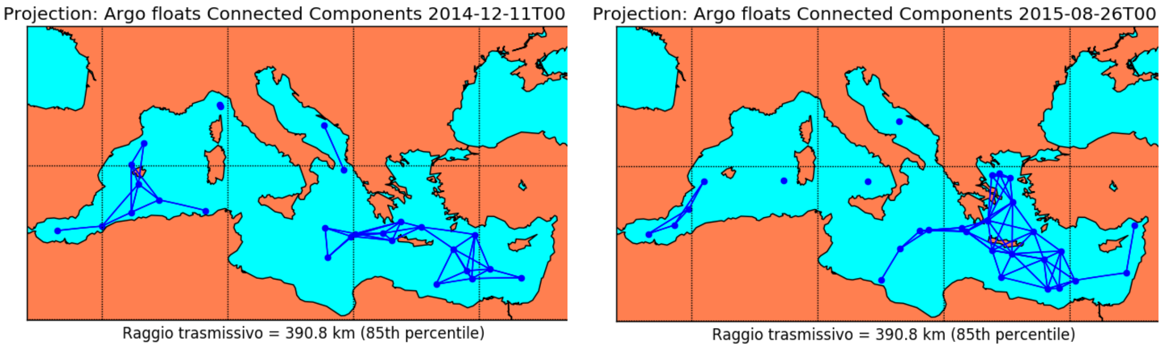

Projection: Argo floats Connected Components 2014-12-11T00
Raggio trasmissivo = 390.8 km (85th percentile)
Projection: Argo floats Connected Components 2015-08-26T00
Raggio trasmissivo = 390.8 km (85th percentile)

Figure 34 Argo Underwater sensor Network using 85th percentile as transmission range.

Figure 34 shows two snapshots of the Argo Underwater Sensor Network using γ = 390.8 km as transmission range. In both maps, we can see that almost all the floats are contained in the two largest connected components. This can also be deduced observing the plots in Figure

33, where the sum of the sizes of the two largest connected components is very close to the number of active floats at same time.

Figure 35 shows that using the value of the 95th percentile as transmission range there are some periods where the Argo Underwater Sensor Network is completely connected. This property was also evident by observing Figure 25. Figure 35 shows also that the third largest connected component size ranges from 0 to 2. This means that almost all the floats are contained in the two largest connected components and during a month (from 19th August 2015 to 24th September 2015) the two largest components are linked, forming a unique connected component containing all the floats.

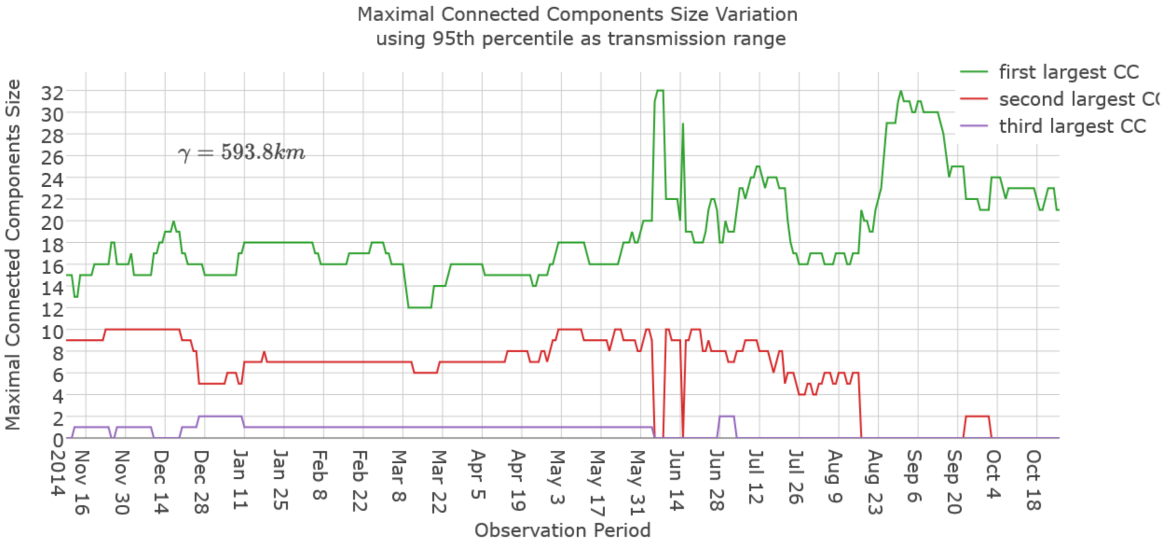


Figure 35 Largest Connected Components size variation using γ = 593.8 km

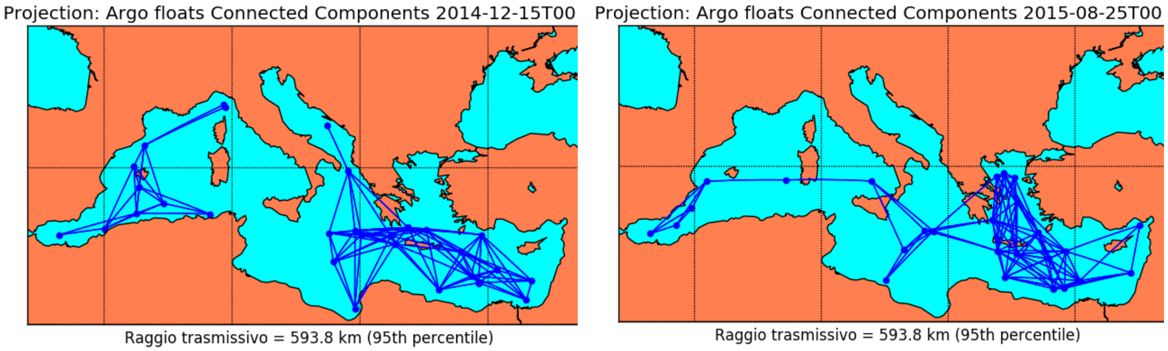

Projection: Argo floats Connected Components 2014-12-15T00
Raggio trasmissivo = 593.8 km (95th percentile)
Projection: Argo floats Connected Components 2015-08-25T00
Raggio trasmissivo = 593.8 km (95th percentile)

Figure 36 Argo Underwater sensor Network using 95th percentile as transmission range.

The first map in Figure 36 show the Argo Underwater Sensor Network in a time instant where all the floats of the network are contained in the maximal connected component or in the second largest component. The second map shows a time instant that is part of the month where the Argo floats form a unique connected component.

3.3.8 Temporal Out-Component Analysis

The Temporal Out-Component of a node *n*, in a time-varying graph, is denoted as $OUT_T(n)$, and it is the set of nodes that can receive a message from *n* during a certain time interval *T*.

In our scenario, the temporal out-component of the float *F*, $OUT_{[t_i,t_j]}(F)$ in the Temporal Graphs $G_i(\gamma)$ with $t_i, t_j \in \{t_1, t_2, \dots, t_{352}\} \wedge t_j > t_i$, is computed using the following procedure (for the sake of simplicity, in the remainder of the chapter we will use the slightly different notation $OUT_i(F)$, to indicate the temporal out-component of a float *F* in the time interval $[t_0, t_i]$):

- $OUT_0(F)$ is composed by *F* and every float that lies within the transmission range of *F* at time t_0
- $OUT_i(F)$ is composed by every float $F' \in OUT_{i-1}(F)$ and every float that lies within the transmission range of *F'* at time t_i

The Temporal Out-Component $OUT_{352}(F)$ can be seen as the number of floats that can be reached by a message sent in broadcast by the Argo float *F* at time *t*$_0$. During the broadcast, every node that receives the message can act as a relay and forward it to every other node it comes in contact during the observation period. This metrics provides a first indicator of the connectivity of the network. Also in this case, we consider the values of γ reported in Table 3.

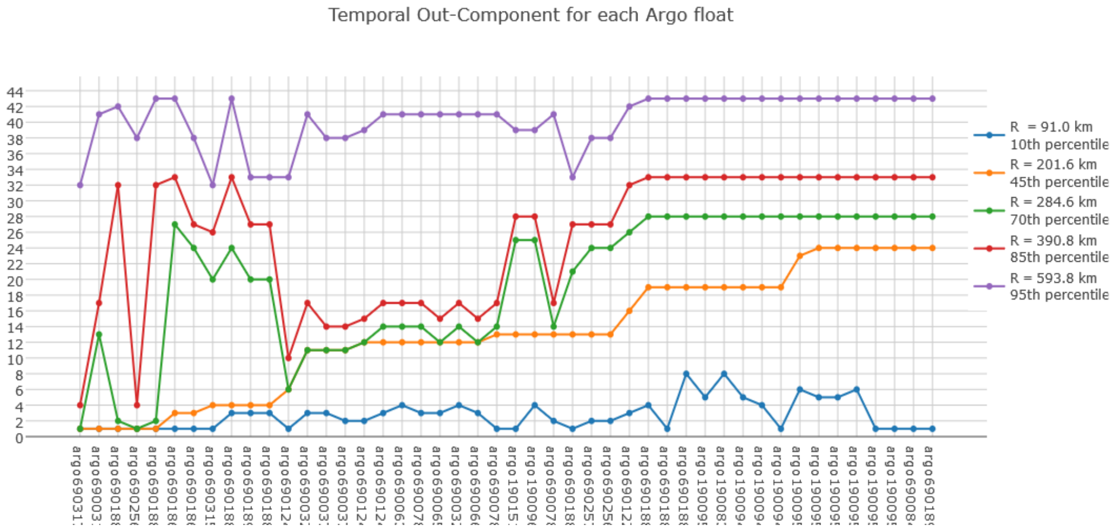


Figure 37 Temporal Out-Components for each Argo float

Figure 37 shows different lines for the different values of **γ**. On the x-axis we show the Argo floats, while on the y-axis we show the number of floats that are part of the temporal out-component $OUT_{352}(F)$.

With γ = 91.0 km, every float can reach only few other floats. In particular, there are 8 floats that are not able to send a message to any other float. Furthermore, the temporal out-components with the maximal dimension are composed only by 8 floats.

With γ = 201.6 km (45th percentile) there are still some floats that are not able to forward any message, but there are some floats whose temporal out-components have size equal to 24, that is close to the average network size (i.e. the average number of active floats at the same time). However, the mean temporal out-component size for our time varying network with transmission range equal to the value of the 45th percentile is close ½ of the average network size and close to ¼ of the total number of floats.

With values corresponding to the 70th and 85th percentiles, the results are very similar. In both of the cases there are some floats whose temporal out-component is very small, but the mean temporal out-component size is above 50% of the maximum network size network size (i.e. the maximum number of active floats at the same time), and close to 1/3 of the total number of floats. Furthermore, using the value of the 85th percentile as transmission range, there is a significant number of floats whose temporal out-component size is 33, which is equal to the maximum network size, and 2/3 of the total number of floats.

Finally, with transmission range equal to the value of the 95th percentile; the minimum temporal out-component size is 32, that is equal to the maximum network size. Furthermore, most of the floats temporal out-component is above 40. This means that most of the floats if equipped with an acoustic modem boasting transmission range $\gamma = 593.8$ km, could reach almost every float contained that is part of the Argo Underwater Sensor Network.

4 Percolation Strategies with the Argo Dataset

In this chapter, we study the performances of some percolation strategies on the Argo Underwater Sensor Network. We exploit the ONE Simulator[25], [26] to simulate message exchange between floats of the AUSN. We first simulate the message generation among the nodes of the AUSN and then we analyse the performance of some well-known opportunistic routing protocols. We chose four opportunistic routing protocols for our simulation, all representative of different strategies. Namely, the Direct Delivery[26], Epidemic [27], SprayAndWait [28] and PROPHET [29] routing protocols. Direct Delivery represent a single-copy approach, the Epidemic routing protocol represent and unlimited-copy epidemic approach, SprayAndWait implements an n-copy routing algorithm and finally, PROPHET adopts a probabilistic approach. Along with the simulations, we analyse some statistics that are useful to understand the performance of opportunistic routing in the AUSN. Specifically, we consider the contact time among floats (i.e. the duration of the period in which two floats are in range of each other), the inter-contact time (i.e. the time elapsed between two consecutive contacts between two floats), and the message delivery ratio obtained by using the aforementioned opportunistic routing protocols.

4.1 The Opportunistic Routing Protocols

In this Section, we provide a description of the opportunistic protocols we chose for our simulations.

4.1.1 Direct Delivery Routing Protocol

The Direct Delivery Routing protocol is the simplest opportunistic routing protocol. When the sender *s* must send the message *m* to the recipient *r, s* does not forward the message *m*, until it encounters *r*. By using the Direct Delivery routing protocol, no copies of the message *m* are created, the node *s* keeps the message in its buffer until it meets *r*. Only nodes *s* and *r* participate to the message forwarding process. Considering the simplicity of Direct Delivery Routing, we expect poor performance in term of the Message Delivery metric when compared to other sophisticated protocols. More specifically, with Direct Delivery if two nodes of the network will never meet each other, then they will be not able to interact at all.

4.1.2 Epidemic Routing Protocol

The Epidemic [27], is a very efficient routing protocol in terms of message diffusion. Its strategies relies on flooding the network with multiple copies of the same message. The Epidemic Routing does not make any assumptions about the topology of the network, rather every node *n* carries the messages it has generated as well as messages generated by other nodes. In the version of the Epidemic protocol implemented in The ONE[26], when a pair of nodes *n* and *n'* come into communication range of one another, *n* forwards all the messages stored in its buffer to *n'* and vice-versa. A message in the network is not forwarded anymore only when it is delivered to the recipient.

It is worth to notice that in a simulation scenario in which buffer space of nodes and the available bandwidth are unlimited, Epidemic Routing gives the optimal solution to the problem of routing in an intermittently connected network with regard to message delivery ratio and latency of the message forwarding.

However, with real-world application scenarios, the Epidemic Routing protocol causes high resource consumption. By flooding the network with message copies, the Epidemic Routing needs a huge amount of memory to store messages in the nodes' buffers and massive energy consumption needed to transfer the messages. For these reasons, many implementations of the Epidemic Routing protocol try to reduce the number of copies flowing through the network using some heuristics, such as bounding the buffer size of each node or setting a time-to-live for the messages. When a node *n* receives a message to forward and its buffer is full, *n* must discard a stored message. When a node *n* receives a message *m* having an exhausted time-to-live, *m* is discarded.

The ONE implementation for the Epidemic Routing uses the drop-oldest buffer policy: when a node's buffer is full, it discards the oldest stored message.

By setting low thresholds for the buffer size and hop count the resource consumption decreases, but the performances in terms of message delivery and latency decreases as well.

Along with our simulations, we set the buffer size parameter greater than the total number of messages generated during and we set the time-to-live parameter greater than the entire simulation time. We made this choice so that to simulate an ideal scenario for our network where nodes have buffers of infinite size and messages have an infinite time-to-live. By simulating message exchange using the Epidemic Routing protocol with infinite buffer sizes and infinite time-to-live, no message is dropped during the simulation, and thus we can have a clear idea of what is the optimal message delivery ratio of the Argo Underwater Sensor Network using opportunistic routing.

4.1.3 Spray and Wait Routing Protocol

SprayAndWait (SnW) [28] is an n-copy routing protocol that limits the number of message copies created to a configurable maximum. SprayAndWait represents a compromise between the simplicity of Direct Delivery and the aggressiveness of the Epidemic routing protocol.

SprayAndWait works by replicating the messages that must be forwarded, but unlike Epidemic, SnW limits the total number of copies and transmissions per message.

SnW compared to Epidemic has the advantage to perform significantly fewer transmissions, leading to a better usage of the resources such as memory needed to store messages in the buffers and energy used to transmission messages.

SnW is topology oblivious, as it does not exploit the knowledge of the network structure while forwarding messages.

Spray and Wait consist of two phases:

- *Spray phase*: whenever a node *n* produces a message to forward, *L* copies of the message are created. The *L* copies of the message are “sprayed” to *L* different relays. In other words, the source node *n* distributes one or more of the *L* copies of the message to the nodes it encounters. The relay nodes holding more than one copy of the message can forward one or more copies to other nodes. As we will discuss below, different spraying strategies can be used.

- *Wait phase*: this phase starts when the *L* copies of the message have been distributed among *L* different nodes and the recipient has not received the message. Each of the *L* nodes carrying a copy of the message switch to direct delivery mode, in other words, they will forward the message only to the recipient.

As we mentioned before, SnW supports different spraying strategies. The work in [28] describes two ways to distribute messages during the spraying phase:

- *Source SprayAndWait:* it represents the simplest spraying strategy. The source node forward a copy of the message to the first *L* nodes it encounters.
- *Binary SprayAndWait:* the source of the message *m* generates *L* copies of *m*, and carries them until it meets another node. Any node *n*, being it source or relay, that carries $k > 1$ copies of the *m* and encounters another node *n'* with no copies of the message, *n* forwards $\lfloor k/2 \rfloor$ copies of *m* to *n'*, while keeping $\lceil k/2 \rceil$ for itself.

In [28] has been demonstrated that when the node movement is independent identically distributed, Binary SprayAndWait has the minimum expected delay among all SnW routing algorithms.

The ONE implements both the aforementioned versions of SprayAndWait, and provides the user with the possibility to choose between them and fix the maximum number of copies *L* that can be generated for each message.

4.1.4 PROPHET Routing Protocol

PROPHET (Probabilistic Routing Protocol using History of Encounters and Transitivity)[29] is an opportunistic routing protocol that exploits the non-randomness of real-world mobility to improve routing performances. PROPHET make use of the *delivery predictability* metric $P_{(a,b)}$, that is the probability for a node *a* to deliver a message to node *b*.

The delivery predictability $P_{(a,b)} \in [0,1]$ is computed in three steps:

- When a node *n* meets a node a, the delivery predictability is computed as:

  $$P_{(a,b)} = P_{(a,b)_{old}} + \left(1 - P_{(a,b)_{old}}\right) * P_{init}$$

  Where $P_{init} \in (0, 1]$ is an initialization constant. Using this formula to compute $P_{(a,b)}$, nodes that are encountered often have a high delivery predictability.
- The delivery predictability is also affected by an *aging* factor:

  $$P_{(a,b)} = P_{(a,b)_{old}} * \gamma^k$$

  With $\gamma \in (0, 1)$ is the aging constant and k is the number of time units that have lapsed since the last time the metric was aged. The aging process is used to decrease the delivery predictability of nodes that have not met for a long time.
- Finally, delivery predictability has also a *transitive* property. When two nodes meet each other, they exchange delivery predictability vectors, containing the delivery predictability the delivery predictability information for destinations known by the nodes. The impact of the transitive property on the delivery predictability is computed as follows:

  $$P_{(a,c)} = P_{(a,c)_{old}} + \left(1 - P_{(a,c)_{old}}\right) * P_{(a,b)} * P_{(b,c)} * \beta$$

  Where β is a scaling constant that decides how large must be the impact of the transitivity property on the delivery predictability. If a encounters often b and b encounters frequently *c*, than *a* has a good probability to delivery messages destined to *c*.

When a node *n* is carrying a message destined to node *d* and *n* meets a node *n'*, *n* forward the message to *n'* only if, according to the delivery predictability metric, *n'* has more chances then *n* to deliver a message to *d.*

Compared to Epidemic, PROPHET generates lower number of message copies and thus it reduces resource consumption (e.g. energy consumption, memory allocation and transmission bandwidth).

Compared to Spray and Wait and Direct Delivery, PROPHET does not set a maximum number of message copies that can be created.

ONE implements PROPHET setting as default parameters $P_{init} = 0.75$, $\gamma = 0.98$ and $\beta = 0.25$, as these are commonly used parameters for PROPHET [29]. Differently, the magnitude of the time unit that is used while computing the aging process must be set by the user.

4.2 The ONE Simulator

The ONE Simulator (Opportunistic Network Environment Simulator)[25] [26] is a Java-based framework designed to evaluate DTN routing and application protocols. The ONE is used to simulate scenarios based on synthetic movement models as well as on real-world traces. Furthermore, the architecture of The ONE allows to easily extend the simulator so that to implement new networking algorithms such as routing or data dissemination.

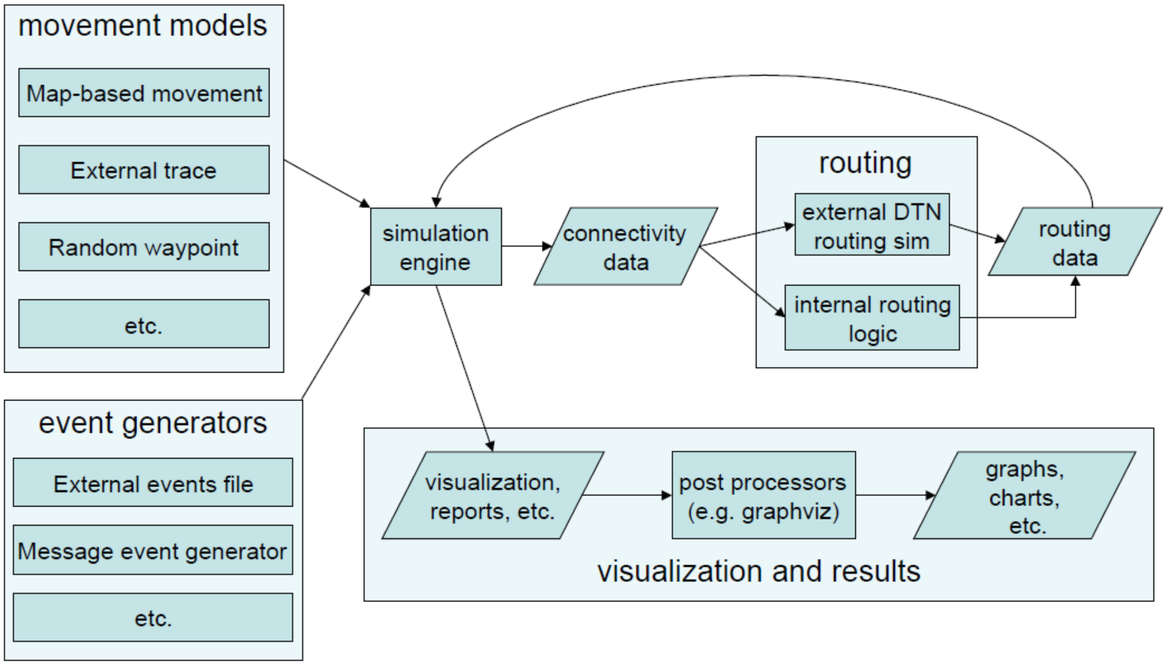


Figure 38 Overview of the ONE Simulator

Figure 38 shows the infrastructure of The ONE.

The ONE includes functionalities to model:

- nodes capabilities
- movement models
- energy consumption
- routing and application protocols.

The ONE implements a set of capabilities for the nodes, such as a radio interface, a buffer, movement and message routing. Considering that the focus of The ONE is on the routing level, the radio interface is abstracted only to a communication range and bit-rate. The energy consumption model is based on an energy budget approach. Every time a node performs an action such as the transmission of a message, it spends part of the energy budget. The energy budget can be refilled at certain locations. Concerning routing, application and movement capabilities they are implemented through the interaction with the movement and routing modules.

The algorithms included in the simulator can generate mobility models that implements movement capabilities of the nodes or external mobility data can be loaded exploiting specific interfaces. Examples of synthetic models included in ONE are: random movement, map-constrained random movement and human behaviour based movement. Furthermore, The ONE includes a framework for implement and integrate user defined movement models in the simulator.

The routing module is implemented similarly to the movement module. The ONE already provides a set of known state-of-the-art routing protocols for Opportunistic Networks, such as Direct Delivery, PROPHET, Epidemic, SprayAndWait. Moreover, The ONE allows to implement new routing protocols.

The Application Support provides two ways to generate application messages: message generators (e.g. single recipient and broadcast) or from external event files. The source and destination of the generated messages can be fixed or chosen randomly.

The ONE provides also the possibility to reproduce real-world scenarios where, instead of using a mobility model, the nodes movements are reproduced by exploiting real-world mobility traces (e.g. GPS coordinated) or by exploiting the co-location traces. A co-location trace contains information about the connections between the nodes of a network (see Section 4.2.1).

Furthermore, The ONE provides a Graphical User Interface (GUI) for visualizing the network behaviour during the simulation. In particular, the GUI shows nodes movement on a map and reports, in real-time on a log, important events such as message delivered or relayed.

Lastly, the ONE generates reports at the end of the simulation containing information about contacts (e.g. Contact time, Inter-contact time or Encounters vs Unique Encounters) or message statistics (e.g. Message Delivery Probability or Message Delay).

4.2.1 Generation of Co-location traces

We configure our simulations so that to exploit the co-location traces. Such traces are obtained by analyzing the mobility of the Argo floats that are part of the network defined in Section 3.3. The co-location traces have usually the following format: <time_unit CONN ID1 ID2 up/down>

```
360 CONN 1 7 up

408 CONN 1 7 down
```

The previous snippet shows that at simulated time 360, a connection between the nodes with identifiers 1 and 7 is established, while at simulated time 408, the connection between nodes 1 and 7 is disrupted.

We exploited our knowledge of the Temporal Graphs $G_i(\gamma)$, representing the AUSN, to generate the co-location traces. In particular, we generated five co-location traces, corresponding to each transmission range that we chose in Section 3.3.7 (Table 3), so that we could run different simulations for each chosen transmission range. We used the following approach to generate one trace for each γ:

- If at a certain time instant *T*, two floats *F* and *F‘* are in the communication range of one another and they are not already connected, than a connection between *F* and *F’* is established and we write in the co-location trace:

$$T \texttt{ CONN } F\ F' \texttt{ up}$$

- If at a certain time instant *T*, two floats *F* and *F’* are connected and either they are not anymore in the communication range of one another, or one of the two floats becomes inactive, than the connection between *F* and *F’* is interrupted and we write in the co-location trace:

$$T \texttt{ CONN } F\ F' \texttt{ down}$$

4.3 Temporal Analysis of the Argo Floats

In this Section, we analyse the reports generated by The ONE. Specifically, we simulated the behaviour of the Argo Underwater Sensor Network without any message exchange and we set the simulator to generate statistics about the nodes contacts, namely Contact-Times, Inter-Contact Times and Encounter vs Unique Encounters[13].

[13] Please consider that along with our simulations, we set hours as the time unit of the network simulator. Therefore, with an observation time of 352 days, our simulations last for 8448 time units.

4.3.1 Analysis of the Contact Times

The Contact Time between the nodes *n* and *n'* is the time elapsed from the connection and the subsequent disconnection between them.

We computed the Complementary Cumulative Distribution Function (CCDF) for the Contact Times. The CCDF of a real-valued random variable *X*, evaluated at *x* represents the probability that *X* will take a value that is higher than *x*. The CCDF is defined as:

$$\bar{F}(x) = P(X > x)$$

CCDF is a widely used metric for analysing Contact and Inter-Contact times among nodes of mobile networks [21][18].

Since, the time interval between two consecutive snapshots is 24 hours, every contact time recorded during the simulations is multiple of 24.

We exploited the co-location traces that we defined in Section 4.2.1 to model the contacts between the nodes of the network.

As we can see from Figure 39, the behaviour of the CCDF of the Contact Times is very similar for all the chosen transmission ranges. In all the cases, the majority of the contacts lasted less than 400 hours (17 days). In particular, with $\gamma = 91.0$ km, none of the contacts lasted more than 2280 hours (95 days), only the 40% of the contacts lasted more than 200 hours (8 days).

The CCDF of the contact times using $\gamma = 201.6$ km has a similar behaviour to the one obtained with $\gamma = 91.0$ km. In particular, only the 10% of the contacts lasted more than 1200 hours (50 days).

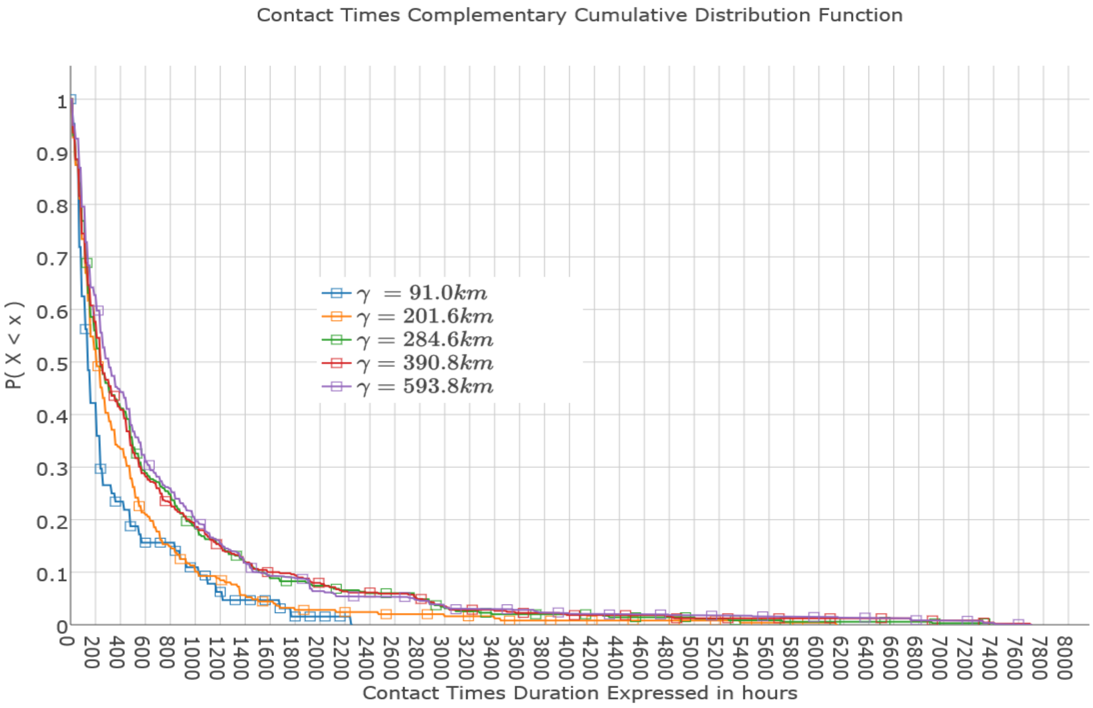


Figure 39 CCDF of the Contact Times among nodes

The three line plots representing the CCDF of the contact times of the AUSN using γ = 284.6 km, γ = 390.8 km or γ = 593.8 km show a very similar shape, only the 20% of the contacts lasted more than 1200 hours, but there are some contacts whose duration is very close to the duration of the entire observation period.

Higher values of the contact times are very likely due to floats that are trapped in the same vortex and for this reason, they remain close (and thus in contact) to each other for a long time.

4.3.2 Analysis of the Inter-Contact Times

The inter-contact time between two nodes *n* and *n'* represents the time elapsed from a disconnection and the next connection between *n* and *n'*.

As we did for the Contact Times, we computed the CCDF for the Inter-Contact Times.

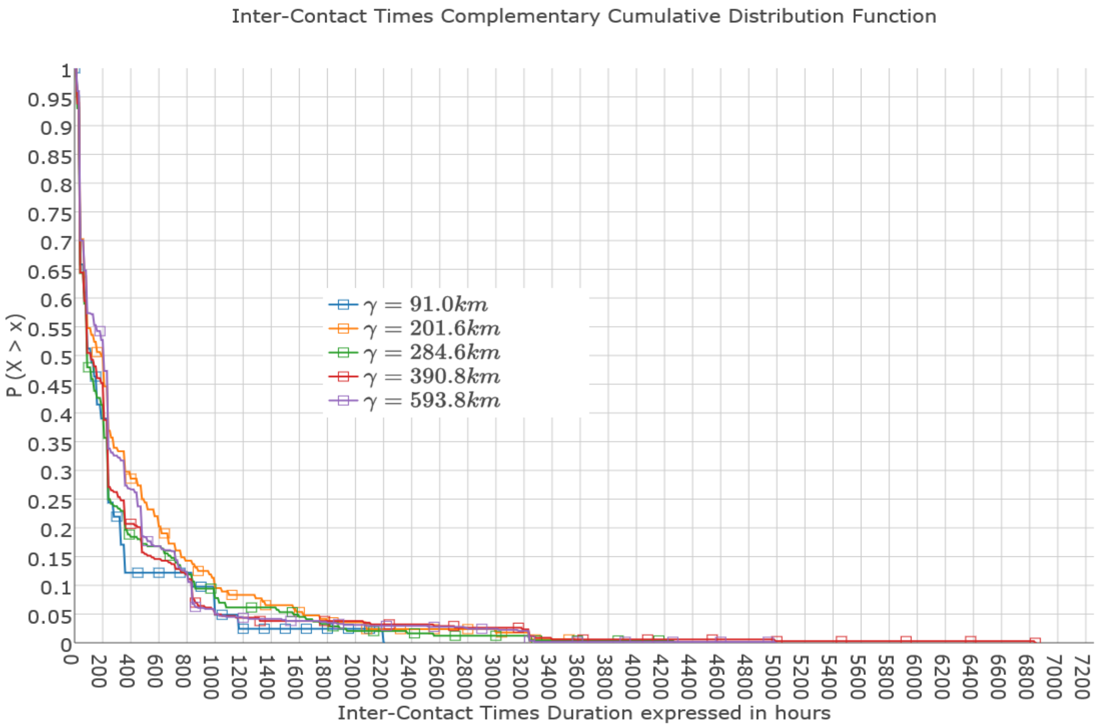


Figure 40 CCDF of the Inter-Contact Times among nodes

In Figure 40, we can observe that the CCDF of the Inter-Contact Times for all the chosen transmission ranges. All the plots show a very similar trend, in particular in all the cases the 40% of the inter-contacts lasted less than 120 hours (5 days). Floats that are close and meet frequently cause this behaviour. Conversely, there is a 5% of the inter-contacts lasted more than 1800 hours (75 days).

By plotting the CCDF of the Contact and Inter-Contact times in a log-log scale (Figure 41), we can observe that both functions decay exponentially.

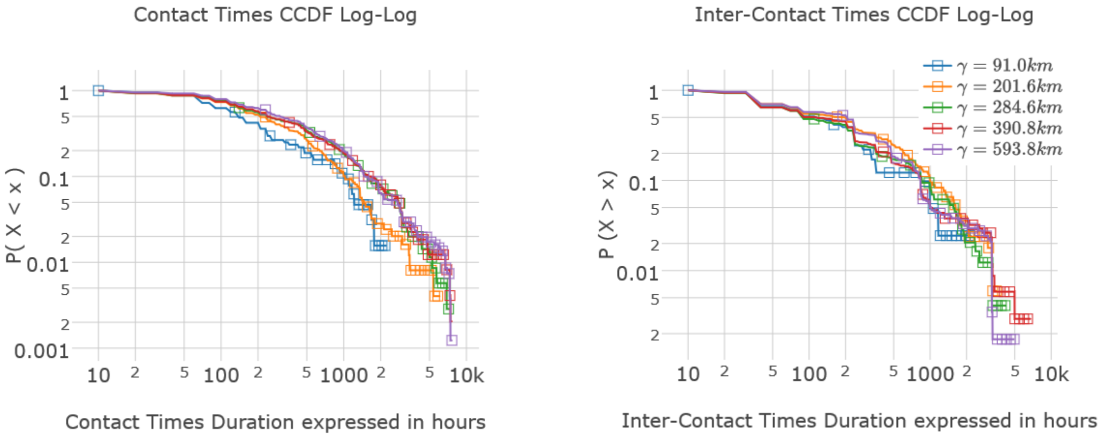
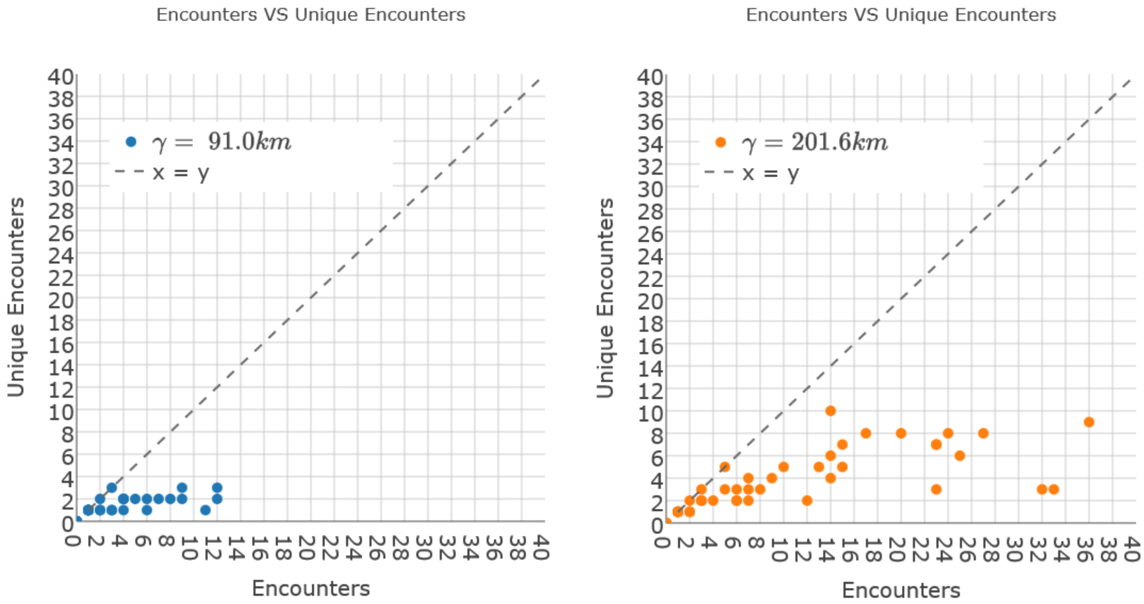


Figure 41 CCDF of Contact and Inter-Contact Times in log-log scale

4.3.3 Analysis of the Encounters and Unique Encounters

In this paragraph, for each float *F*, we analyse the relationship between the number of encounters (how many floats *F* encounter during the simulation) and the unique encounters (how many different floats *F* encounters during the simulation). This analysis reveals very interesting behaviour of the floats. In particular, if a float *F* reports a high number of encounters, but a low number of unique encounters, than *F* has encountered frequently a same small group of floats. Differently, if *F* reports a number of encounters that is equal to the number of unique encounters, than *F* has met a different float in each encounter. The floats reporting a high number of unique encounters and an even higher number of encounters are the most interesting to us, because in an opportunistic environment, they can be used as "couriers". Considering that couriers nodes meet many different other nodes, they can be effectively used as relays.


Encounters VS Unique Encounters

Encounters VS Unique Encounters

Figure 42 Encounters VS Unique Encounters for transmission ranges equal to the values of the 10th and 45th percentiles

In Figure 42, we can observe that with the value of the 10th percentile as transmission range (91.0 km), the Argo floats report very few unique encounters. In particular, the maximum number of different floats that can be encountered during the simulation is 4. The number of encounters for each float is low as well, it is a maximum of 12. With such a setting, we expect low performance of the opportunistic routing algorithms we analyse.

We set γ = 201.6 km and we observe an higher value both for the number of encounters and for the number of unique encounters with respect to the previous scenario. The maximum number of unique encounters reported is 10, while the maximum number of encounters is 37. From Figure 42, we can observe a cluster of floats that is close to the origin of the axes. These floats are the ones registering a low number of encounters and local encounters. We can also observe that there are many floats reporting a significant number of encounters, even if the number of unique encounters ranges from 2 to 9.

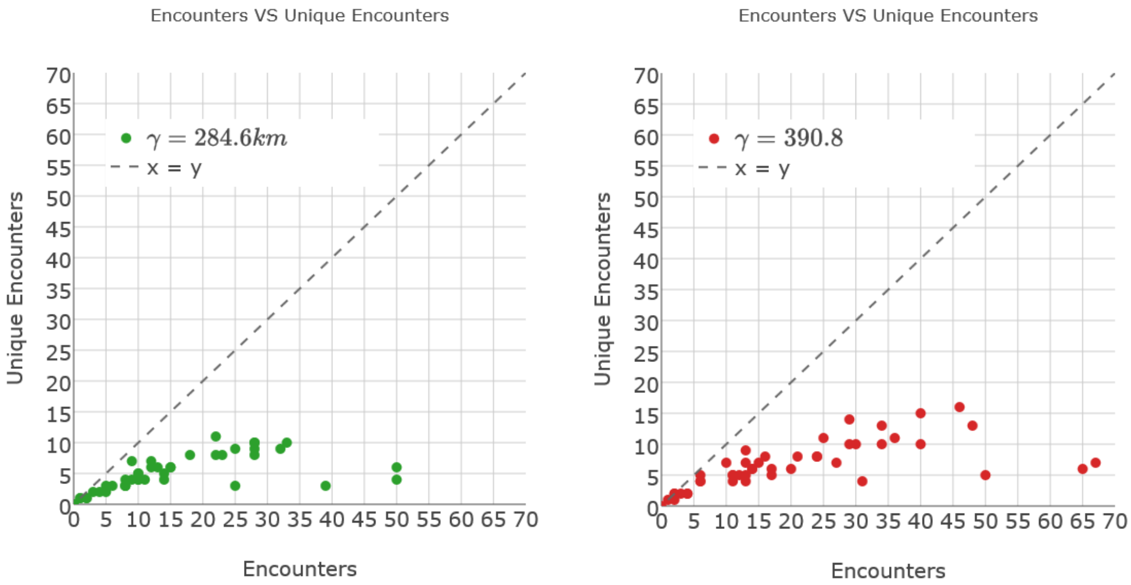


Figure 43 Encounters VS Unique Encounters for transmission ranges equal to the values of the 70th and 85th percentiles

Using the value of the 70th and 85th percentiles as transmission ranges (γ=284.6 km and γ=390.8 km respectively), we can observe that as γ increases, the number of floats that are close to the origin of the axes decrease. Also, Figure 43 shows that as the value of γ increases the number of floats reporting an high number of encounters increases significantly. However, the number of unique encounters for each float does not grows as much as the number of encounters.

Observing Figure 44 we can see that compared to the previous cases, using γ = 593.8 km, the majority of the floats report a number of unique encounters that is above 10. Furthermore, we can observe a group of floats reporting number of encounters ranging from 70 to 85 and a number of unique encounters ranging from 15 to 25. These floats can be effectively used as “couriers”.

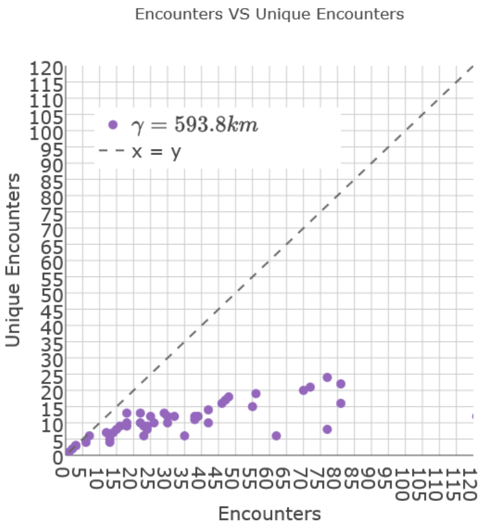


Figure 44 Encounters VS Unique Encounters for transmission ranges equal to the value 95th percentile

4.3.4 Message Delivery Distribution Analysis

We investigate the Message Delivery metrics of the selected algorithm. We configure the simulations according to the following criteria:

- the simulation period is set to 8448 hours, corresponding to the 352 days composing our observation period of the AUSN.
- As in previous sections, we abstracted from the movement model of the nodes by adopting a stationary movement model for the nodes, and we exploited only data from the co-location traces to simulate the interaction between nodes. We assume the existence of a network interface foe every float with a high transmit speed interface boasting 10MBps to simulate instant delivery of the messages. The nodes' buffer size and energy budget are set to infinite.
- We generate one message per hour with a random source and a random destination. The source and destination of each message are selected among the 46 nodes that are part of the Argo Underwater Sensor Network (see Section 3.3). For each message, the size is set to 1 bit and the time to live is set to infinite.

Using these configurations, no message is dropped during the simulations and every message is delivered instantly. In such ideal scenario, all the routing protocols we chose can achieve the best performances possible.

The parameters of the routing protocols PROPHET and SprayAndWait must be set as well. Specifically, for SprayAndWait we set the maximum number of copies that could be generated to 10 (that is close to ¼ of the network size), and we selected the binary mode for the Spraying strategy, because, as mentioned in Section 4.1.3, it achieves better performance compared to the source Spraying strategy. Concerning PROPHET, we set the value of the time unit for the aging process k to 30 hours.

We exploited the co-location traces that we defined in Section 4.2.1 to model the contacts between the nodes of the network. We run multiple simulations for each co-location trace, so that we could analyse the variation of the message delivery ratio with different transmission ranges.

We generate reports for the Message Delivery Probability Distribution. More in detail, The ONE reports hourly the message delivery ratio, computed as follows:

$$message\ delivery\ ratio = \frac{delivered\ messages}{created\ messages} \qquad (1)$$

Using the value of the 10th percentile as transmission range, we can see from Figure 45 that the performances of every routing protocol are very poor. As expected, with a low transmission range the connectivity of the network is very poor and thus the achieved message delivery ratio is very low (the statistics we analysed in previous sections highlighted the poor connectivity of the network for γ = 91.0 km).

All the graphs are characterized by spikes and periods with decreasing and stable values. The spikes are caused by the successful delivery of one or more messages (the numerator of (1) increases), while a decrease in the message delivery probability is due to the creation of new messages in the network (the denominator of (1) increases). Note that the distances among spikes can be used visually as a crude indication of the average delay of packet delivery. As time progresses, the message delivery probability increases. This behaviour is due to the fact that no message is dropped during the simulation (we set the time to live of the messages and the buffer sizes of the nodes to infinite) and thus, more the network fills with messages (and message copies), and more increases the chance to deliver them.

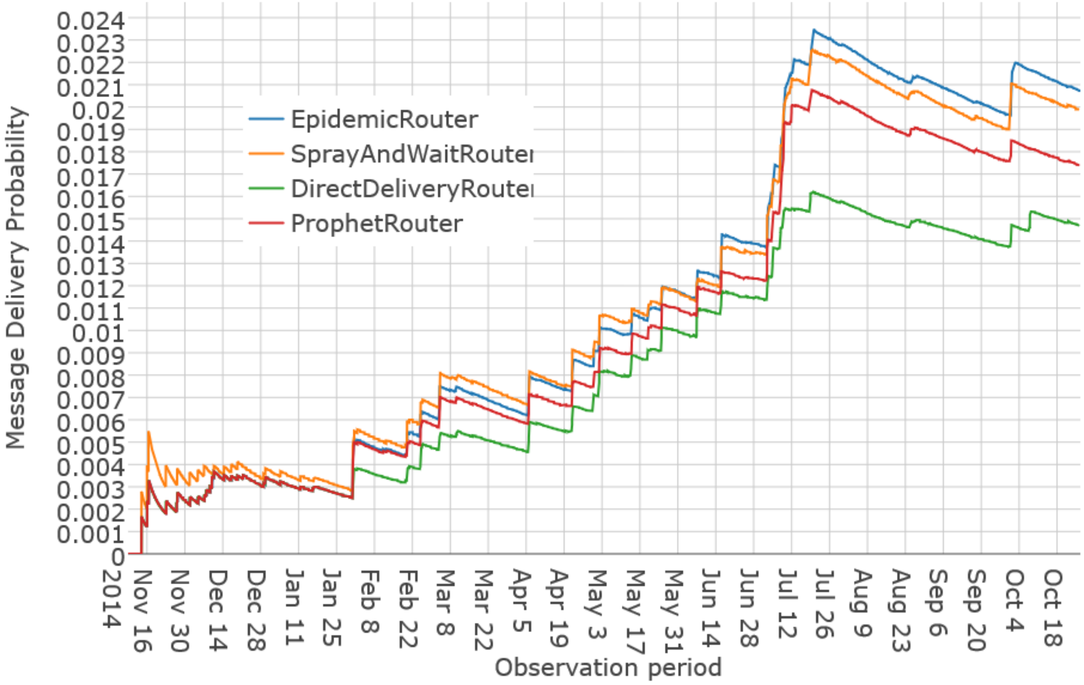


Figure 45 Message Delivery Probability using the value of the 10th percentile as transmission range (γ = 91.0 km)

We can observe that all the protocols register a sudden increase in the message delivery probability in a time period corresponding to mid July 2015. This behaviour is caused by the fact that during July more floats are connected compared to the rest of the observation period (see Section 3.3.7 where we analysed the variation of the connected components of the network).

From Figure 45 we can observe that the performances of the Epidemic, SprayAndWait and PROPHET routing protocols are very similar, while the performances of Direct Delivery differ, in particular we observe low results. The maximum value for the message delivery achieved is obtained by the Epidemic routing protocol (0.024), and by the end of the simulation, the Epidemic, SprayAndWait and PROPHET routing protocols achieve a message delivery probability that is close to 0.02. This means that at the end of the simulation only 2% of the messages are successfully delivered.

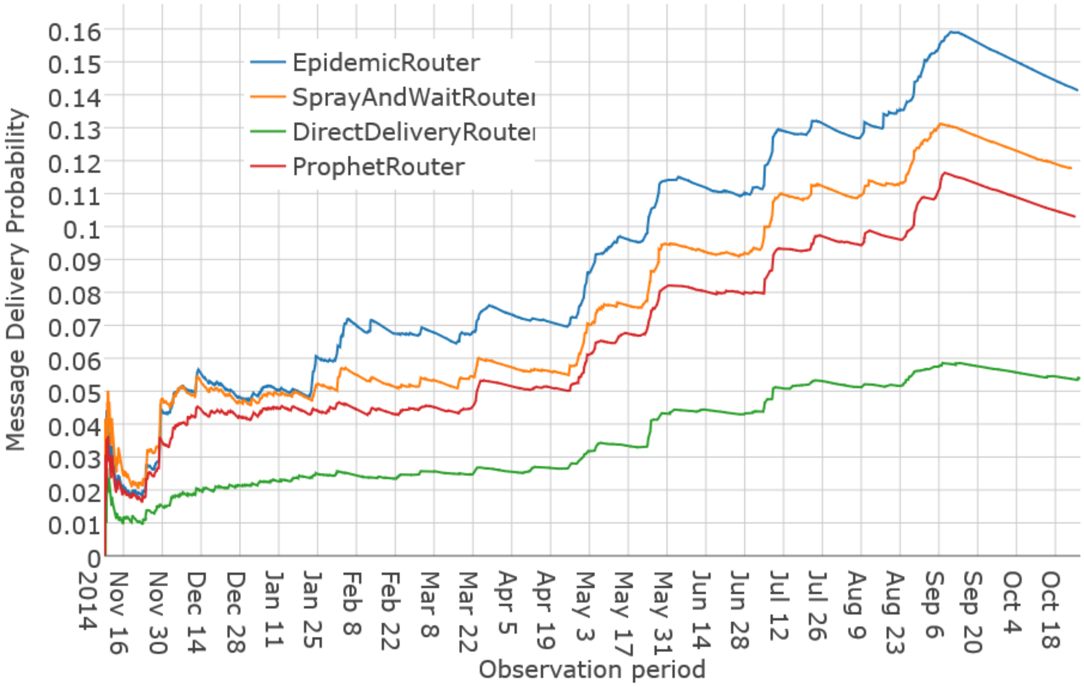


Figure 46 Message Delivery Probability using the value of the 45th percentile as transmission range (γ=201.6 km)

By analysing Figure 46, we can see that with γ = 201.6 km, all the routing protocols perform significantly better, specifically the message delivery probability is 5-7 times higher. Also we can observe that the performances of the Epidemic, SprayAndWait and PROPHET differ remarkably with respect to the results obtained with γ = 91.0 km. We can see that Epidemic (that as we mentioned in paragraph 4.1.2 represents the optimal strategy in ideal conditions) performs significantly better than the other protocols. In particular, the message delivery probability observed is 0.16. The SnW protocol outperforms PROPHET, reaching message delivery probability 0.13 at its maximum and 0.12 at the end of the simulation. The message delivery probability achieved by Direct Delivery is much lower than the one achieved using the other three protocols, specifically it is always lower than 0.06.

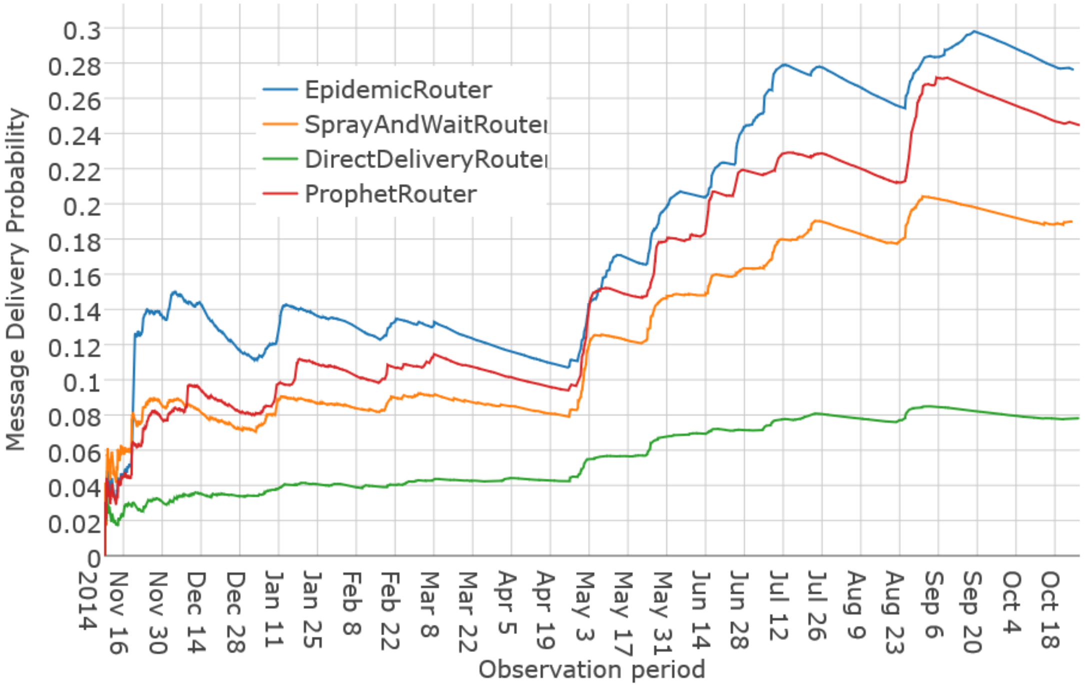


Figure 47Message Delivery Probability using the value of the 70th percentile as transmission range (γ=284.6 km)

Figure 47 shows that the Epidemic and the SnW routing protocols perform two times better when compared to the previous scenario. Specifically, Epidemic reaches 0.3 as message delivery ratio and SnW achieves 0.2. However, it is important to point out that using γ = 284.6 km, PROPHET outperforms SprayAndWait. In addition, the performance of PROPHET, unlike in the previous scenario, are close to the one achieved by Epidemic.

This behaviour is due to the fact that using the γ = 284.6 km there are more floats that report an higher number of encounters (Figure 43). Considering that PROPHET exploits history of encounters, as the number of encounters increase, PROPHET can compute more accurately the delivery predictability of the nodes. A more accurate computation of the delivery predictability leads to a better choice of the next hop during the routing phase. The message delivery probability for PROPHET is 0.27.

The performances of Direct Delivery using the value of the 70th percentile as transmission range, are very close to the one achieved with γ=201.6. This behaviour is caused by the fact that for γ=284.6 km, as shown in Figure 43, there are many floats registering a low number of unique encounters. These floats without the possibility to exploit other nodes as relays, cannot deliver messages to recipients that are far away.

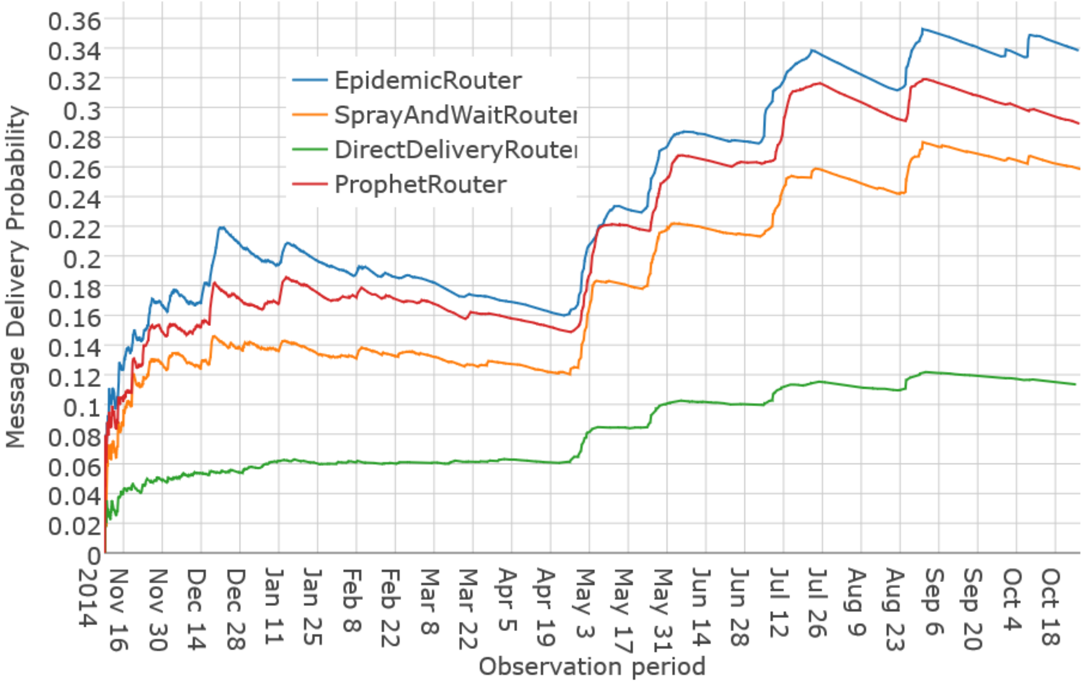


Figure 48 Message Delivery Probability using the value of the 85th percentile as transmission range (γ=390.8 km)

Using the value of the 85th percentile as transmission range, the performances of the chosen routing algorithms are only slightly better than the performances achieved by setting γ= 284.6 km. The only routing protocol reporting a significant increase in the message delivery ratio is Direct Delivery. This improvement can be explained by observing Figure 43 and noting that the for γ = 390.8 km there are less floats reporting a low number of unique encounters and the maximum number of unique encounters reported is significantly higher than the one reported for γ < 390.8 km.

In Figure 48, we can notice that even using a very large transmission range, the message delivery is lower than expected. This is caused by the fact that, as we discussed in Section 3.2.2 not all the floats are active for the entire observation period. This means that, if a node carries one or more messages and becomes inactive until the end of the observation period, it becomes unable to forward the messages it is carrying. In addition, if a message *m* is created with destination *d*, and *d* becomes inactive before receiving m until the end of the simulation, than *m* could never be delivered.

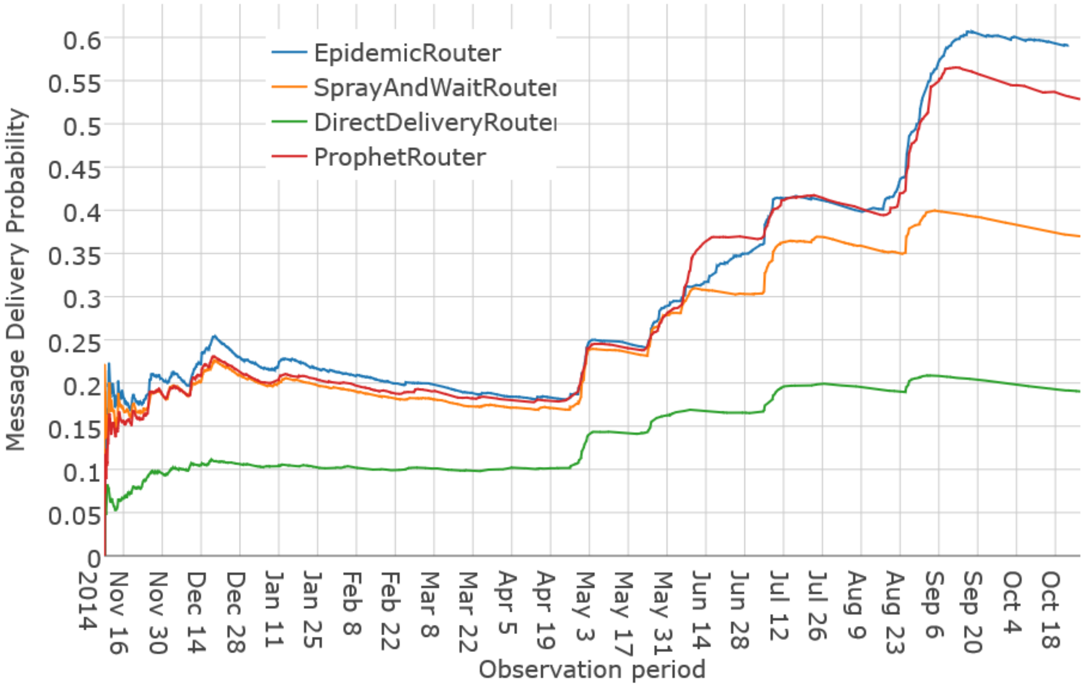


Figure 49 Message Delivery Probability using the value of the 95th percentile as transmission range (γ=593.8 km)

Using the value of the 95th percentile of the MST as transmission range, we can see that the performances of all the routing protocols are considerably higher, compared to the previous scenarios. In Section 3.3.7, we noticed that for γ = 593.8 km, that for nearly all the duration of the observation period, the network is divided in two connected components

For this reason, the message delivery ratio for Epidemic, PROPHET and SnW is very close for most of the simulation time. If at a certain time of the simulation, a message *m* is created, and the source and the destination of the message are in the same connected component, than *m* can be delivered in few hops by any of the three aforementioned protocols.

Furthermore, in Section 3.3.7[14], we noticed that for a fraction of the observation period, corresponding to September 2015, the network is connected. This explains the sudden increase of the message delivery ratio for Epidemic, PROPHET and SnW recorded in Figure 49.

However, as we mentioned while describing the simulation run with γ = 390.8 km, the message delivery ratio is lower than expected because of the floats that becomes inactive until the end of the simulation.

[14] Subsection about the analysis of the Largest Connected Components.

5 Conclusions

Underwater Acoustic Sensor Networks (UASN) represent an interesting and promising research field as they represent an effective and cheap solution for monitoring the underwater environment on a global scale. Currently, the Argo project consists of more than 3900 floats deployed across the world. The Argo floats communicate their measurements only with a satellite link; we consider that such aspect limits the potentialities of the Argo floats. To this purpose, in this thesis we investigated the feasibility of the implementation of the Argo network as an UASN. We built the UASN by exploiting the mobility traces of the Argo floats drifting in the Mediterranean Sea from 5th November 2014 to 10th November 2015.

We first explored the structure of the Argo network by using different connectivity levels based on different transmitting ranges. Such ranges are not yet economically viable, however the analysis we presented in this thesis can be used as a meaningful benchmark on the performance of different routing strategies in a very sparse and challenging network. We modeled the UASN as a Temporal Graph and we analysed some interesting aspects of the network such as the mobility, the average degree, the density, the cardinality of the connected components and the Temporal-Out Components. We studied such metrics by varying the communication range of the floats. The lessons we learned are:

- The Argo Underwater Sensor Network is a sparse network, even when we assume long transmission ranges for the nodes. For this reason, we exploited opportunistic routing algorithms to cope with the limitations posed by the structure of the network.
- Most of the floats of the network are active only for a fraction of the observation period, and thus, analyzing the message forwarding process is a challenging task.
- We observed that most of the floats we considered follow a looping trajectory while drifting, wandering often in the same area. Floats following such mobility pattern encounter often a same small group of other floats. This is an important aspect, since such floats could be not used as relay nodes for carrying information across the network.

After such network analysis, we explored the potentialities of some routing algorithms designed for Opportunistic Networks. We chose four well-known routing algorithms, designed to diffuse messages in partially connected networks. We selected the Direct Delivery, Epidemic, SprayAndWait and PROPHET algorithms. Each of these algorithms is representative of a class of message forwarding strategies. Direct Delivery is representative of the single-copy strategy, Epidemic represents the unlimited-copies epidemic approach, SprayAndWait represents a compromise between the two, adopting an n-copies forwarding strategy and lastly, PROPHET, which is representative of the probabilistic approach.

We measured the Message Delivery Probability that is a commonly used metric for assessing the performance of the routing. Under this respect, we observed that:

- In general, for all the chosen algorithms the overall message delivery is rather poor (lower than 0.6), even using long transmission ranges. This is mainly due to the sparseness of the floats, the fact that most of the floats are not active for the entire observation period, and the fact that the floats cannot be used effectively as relays, because they tend to wander often in the same area.
- The results we obtained show that, also for the underwater networks, the routing protocols based on past information on the frequency of encounters among nodes provide better performances. We observed that the performances of PROPHET are close to the ones obtained with Epidemic, despite the fact that PROPHET requires fewer resources.
- We observed periods during out observation time in which the Message Delivery Probability sharply increases. These increases are caused by changes in the underlying connectivity of the network. In other words, during such periods, the topology of the network allows a better percolation of the messages through the network.

The work done with this thesis represent, to the best of our knowledge, a first attempt to study the structure of the UASN and the performance of routing with real-world mobility traces.

We consider two further interesting aspects that can be investigated. Firstly, perform the analysis presented along with this thesis on different areas, such as the Arabic Sea and the Bay of Bengal and compare the results obtained with respect to the results obtained with the Mediterranean Sea case study. Secondly, developing a realistic 2D mobility model that could be used to study denser deployments of free-drifting floats.

6 References

[1] I. F. Akyildiz, D. Pompili, and T. Melodia, “Underwater acoustic sensor networks: research challenges,” *Ad hoc networks*, vol. 3, no. 3, pp. 257–279, 2005.

[2] L. Lanbo, Z. Shengli, and C. Jun-Hong, “Prospects and problems of wireless communication for underwater sensor networks,” *Wirel. Commun. Mob. Comput.*, vol. 8, no. 8, pp. 977–994, 2008.

[3] J. Partan, J. Kurose, and B. N. Levine, “A survey of practical issues in underwater networks,” *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 11, no. 4, pp. 23–33, 2007.

[4] D. Roemmich, O. Boebel, Y. Desaubies, H. Freeland, K. Kim, B. King, P.-Y. Le Traon, R. Molinari, B. W. Owens, S. Riser, and others, “Argo: The global array of profiling floats,” 2001.

[5] D. Roemmich, O. Boebel, H. Freeland, B. King, P. LETRAON, R. Molinari, W. B. Owens, S. Riser, U. Send, K. Takeuchi, and Others, “On The Design and Implementation of Argo A Global Array of Profiling Floats,” *Argo Sci. Team*, 1999.

[6] P. Santi, *Mobility models for next generation wireless networks: ad hoc, vehicular and mesh networks*. John Wiley & Sons, 2012.

[7] S. Taneja and A. Kush, “A survey of routing protocols in mobile ad hoc networks,” *Int. J. Innov. Manag. Technol.*, vol. 1, no. 3, p. 279, 2010.

[8] Z. Zhang, “Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges,” *IEEE Commun. Surv. Tutorials*, vol. 8, no. 1, pp. 24–37, 2006.

[9] S. Merugu, M. H. Ammar, and E. W. Zegura, “Routing in space and time in networks with predictable mobility,” 2004.

[10] S. Jain, K. Fall, and R. Patra, *Routing in a delay tolerant network*, vol. 34, no. 4. ACM, 2004.

[11] O. Gnawali, M. Polyakov, P. Bose, and R. Govindan, “Data centric, position-based routing in space networks,” in *Aerospace Conference, 2005 IEEE*, 2005, pp. 1322–1334.

[12] R. Handorean, C. Gill, and G.-C. Roman, “Accommodating transient connectivity in ad hoc and mobile settings,” in *Pervasive Computing*, Springer, 2004, pp. 305–322.

[13] A. Caruso, F. Paparella, L. F. M. Vieira, M. Erol, and M. Gerla, “The meandering current mobility model and its impact on underwater mobile sensor networks,” in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.

[14] A. S. Bower, “A simple kinematic mechanism for mixing fluid parcels across a meandering jet,” *J. Phys. Oceanogr.*, vol. 21, no. 1, pp. 173–180, 1991.

[15] Y. Ren, W. K. G. Seah, and P. D. Teal, “Performance of pressure routing in drifting 3D underwater sensor networks for deep water monitoring,” in *Proceedings of the seventh ACM international conference on underwater networks and systems*, 2012, p. 28.

[16] P. Jiang, J. Liu, F. Wu, J. Wang, and A. Xue, “Node Deployment Algorithm for Underwater Sensor Networks Based on Connected Dominating Set.,” *Sensors (Basel).*, vol. 16, no. 3, p. 388, Jan. 2016.

[17] S. Farrell and V. Cahill, *Delay-and disruption-tolerant networking*. Artech House, Inc., 2006.

[18] T. Karagiannis, J.-Y. Le Boudec, and M. Vojnović, “Power law and exponential decay of intercontact times between mobile devices,” *Mob. Comput. IEEE Trans.*, vol. 9, no. 10, pp. 1377–1390, 2010.

[19] E. Hyytia and J. Ott, “Criticality of large delay tolerant networks via directed continuum percolation in space-time,” in *INFOCOM, 2013 Proceedings IEEE*, 2013,

pp. 320–324.

[20] V. F. S. Mota, F. D. Cunha, D. F. Macedo, J. M. S. Nogueira, and A. A. F. Loureiro, “Protocols, mobility models and tools in opportunistic networks: A survey,” *Comput. Commun.*, vol. 48, pp. 5–19, 2014.

[21] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, “Impact of human mobility on opportunistic forwarding algorithms,” *Mob. Comput. IEEE Trans.*, vol. 6, no. 6, pp. 606–620, 2007.

[22] R. Coutinho, A. Boukerche, L. Menezes Vieira, and A. Loureiro, “Geographic and Opportunistic Routing for Underwater Sensor Networks,” *IEEE Trans. Comput.*, vol. 65, no. 2, pp. 548–641, 2016.

[23] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora, “Graph metrics for temporal networks,” in *Temporal Networks*, Springer, 2013, pp. 15–40.

[24] R. Bullock, “Great circle distances and bearings between two locations,” *MDT, June*, vol. 5, 2007.

[25] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE simulator for DTN protocol evaluation,” in *Proceedings of the 2nd international conference on simulation tools and techniques*, 2009, p. 55.

[26] A. Keranen, “Opportunistic network environment simulator,” *Spec. Assign. report, Helsinki Univ. Technol. Dep. Commun. Netw.*, 2008.

[27] A. Vahdat, D. Becker, and others, “Epidemic routing for partially connected ad hoc networks.” Technical Report CS-200006, Duke University, 2000.

[28] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and wait: an efficient routing scheme for intermittently connected mobile networks,” in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, 2005, pp. 252–259.

[29] A. Lindgren, A. Doria, and O. Schelén, “Probabilistic routing in intermittently

connected networks," *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, 2003.