SAPIENZA
UNIVERSITÀ DI ROMA

Design and evaluation of strategies to minimize operating costs and support in-orbit Distributed Learning within satellite networks for Earth Observation

Dipartimento di Ingegneria dell'informazione, elettronica e telecomunicazioni
Dottorato di Ricerca in Tecnologie dell'Informazione e delle Comunicazioni (XXXVI cycle)

Francesco Valente
ID number 1706964

Advisor
Ch.mo Prof. Vincenzo Eramo

Co-Advisor
Dr. Francesco Giacinto Lavacca

Academic Year 2022/2023

Design and evaluation of strategies to minimize operating costs and support in-orbit Distributed Learning within satellite networks for Earth Observation
PhD thesis. Sapienza University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: francesco.valente@uniroma1.it

Abstract

Earth Observation (EO) satellites currently acquire images to be then stored in their on-board memory, until they fly over a ground station, when they shall transmit a high amount of data in a short time. Thus, a high data rate is requested, but this is constrained by the power available on-board, in turn limited by the dimensions and masses of solar panels and batteries. However, not all data transmitted to ground are actually useful to the application. A solution can be obtained by endowing satellites with on-board processing capacity and Inter-Satellite Links (ISLs) to make them able to offload data processing to other satellites whenever they have not enough resources to accomplish the processing task. This would allow for an on-board extraction of the useful information from acquired images, leading to an increased efficiency in bandwidth usage and to a reduction of both the time needed to deliver information to the ground station and of the energy to be used by ground stations to process information. However, transmission, storage, and computational capacity available for in-orbit processing are valuable resources and could be not always available. For this reason, it is necessary to design strategies to appropriately allocate bandwidth and processing resources on satellites and to leverage the possibilities opened by the network of satellites made possible by ISLs, while optimizing a desired metric. In this thesis, I propose strategies to minimize operating costs of EO satellite networks, to save energy due to image processing on ground stations, and to support in-orbit training of machine learning models in a distributed manner to allow for faster accuracy convergence while reducing both the bandwidth and on-ground energy consumption with respect to centralized learning solutions. In particular, I first introduce and solve an optimization problem to allocate transmission, memory and processing resources to minimize the total operating cost to be paid for transferring, elaborating and storing EO data. Furthermore, since the proposed optimal strategy is NP-hard, I also define and evaluate two heuristics to be applied in real orbital scenarios, proving that they outperform benchmark solutions. Second, I introduce two optimal strategies to maximize energy saving on ground stations by leveraging at most in-orbit EO image processing. In particular, in the first strategy I do not take into account any constraint on the level of usage of on-board batteries to optimize operative life. Since this strategy results to be NP-complete, I also introduce a heuristic to be applied in real orbital scenarios. Instead, the second optimal strategy aims to maximize ground station energy saving while also optimizing satellite operative life by assuring that batteries are not discharged under a certain threshold. Results obtained with all the proposed strategies also provide useful insights on how the on-board CPU clock frequency has to be chosen to obtain optimal results, given limitations on energy available on satellites. Finally, I propose a communication strategy to support in-orbit distributed training of deep learning models by leveraging the satellite network made of both intra-orbital and inter-orbital ISLs. In particular, the proposed distributed learning strategy provides for satellites exchanging locally trained models within themselves, without having to lean on a central parameter server as it happens in federated learning schemes available in literature. Obtained results show that such strategy allows for reaching model convergence in a shorter time if compared to federated learning-based schemes.

Contents

List of Figures

List of Acronyms

AG Always Ground

AI Artificial Intelligence

B5G Beyond 5G

COTS Commercial-Off-The-Shelf

CNN Convolutional Neural Network

DoD Depth-of-Discharge

DL Distributed Learning

DRL Deep Reinforcement Learning

EO Earth Observation

ESA European Space Agency

FL Federated Learning

FoG Always First or Ground

FPGA Field Programmable Gate Array

GEO Geostationary Orbit

GS Ground Station

GSESH Ground Station Energy Saving Heuristic

HAP High Altitude Platform

ILP Integer Linear Programming

IoT Internet of Things

IoV Internet of Vehicles

ISL Inter-Satellite Link

LEO Low Earth Orbit

LSTM Long-Short Term Memory

MEC Multi-access Edge Computing

MEO Medium Earth Orbit

ML Machine Learning

NFV Network Function Virtualization

NTN Non-Terrestrial Newtork

OEC Orbital Edge Computing

QoS Quality-of-Service

RL Reinforcement Learning

SAGIN Space-Air-Ground Integrated Network

SAR Synthetic Aperture Radar

SDN Software Defined Network

SFC Service Function Chain

UAV Unmanned Aerial Vehicle

vGS Virtual Ground Station

Introduction

One of the most important limitations Earth Observation (EO) satellites have to face is related to the transmission of a huge amount of information to the ground in a limited time, given by the instants during which a ground station is visible. In fact, nowadays satellites typically acquire images when flying over regions of interest, store them into on-board memories, and then downlink all gathered images to a ground station when flying in an appropriate visibility region. To accomplish the deliver of this high amount of data, it would be then necessary to increase either the downlink data rate or the number of ground stations, both being unscalable with the increase of collected data. In particular, the data rate for transmissions from a satellite is constrained by the power available on board, in turn constrained by the dimensions and masses of solar panels and batteries.

Instead, a novel way to face these limitations can be given by Orbital Edge Computing[1], providing for endowing satellites with processing capabilities enabling them to elaborate acquired data directly on-board, so as to reduce the amount of information to be delivered on ground. In fact, it is important to underline how not all data transmitted to the ground are useful to the application. For example, in a forest we may be only interested in detecting if there is a fire and transmit only this information, while we are not interested in downlinking the full image to be then elaborated. Thus, an on-board selection of the information to be transmitted thanks to an on-board data processing could avoid waste of bandwidth, since only useful information is transmitted[2, 3, 4, 5]. This also reduces the time needed to deliver information to the ground station, since inter-satellite links (ISLs) among satellites of the constellation can be used to transfer the data from the satellite which gathered it to the satellite in visibility of the station at any time[6, 7, 8]. However, to obtain these benefits, strategies have been identified to decide where data shall be processed (satellite or ground station) [9, 10, 11]. All of these solutions assume that when the task is processed by the satellite, the data processing can be only performed by the acquiring satellite, and the impact of the limited bandwidth in the LEO satellite network is not considered. Furthermore, energy usage minimization on-board of satellites is the focus of several research works. For example, a strategy to allocate processing and bandwidth resources in EO constellations to minimize the energy used by satellites, providing again for processing to happen either on the satellite acquiring the image or on ground has been proposed in [11], while the algorithm proposed by authors of [12] aims to optimize the use of satellite batteries to extend their operational life. However, these works do not take into consideration that once a satellite is in orbit, energy generation always happens (e.g., solar panels always generate power when exposed to solar radiation), and the amount of generated

energy is only dependent on the exposition of solar panels to the sun, regardless of the actual energy demand [13]. For this reason, this amount of energy could be leveraged to in-orbit process acquired images, in order to reduce the energy to be spent on ground for data elaboration purposes. This possibility is even more interesting by considering that energy harvested by satellites is completely renewable, while this is not always the case on ground. Thus, in-orbit processing may also help to achieve green communications objectives[14].

The next research frontier is in understanding how satellites can collaborate among themselves to accomplish in-orbit processing in a shared manner, in such a way that, if a satellite has not enough resources to elaborate information it gathered on-board, it can offload the processing task to another satellite which, instead, has enough resources to accomplish it [15]. This is made possible by the fact that satellites may form a network by means of ISLs, on which data can be shared. However, information can be routed on this network in several ways, and there may be more than a satellite able to accomplish data processing within the constellation. For this reason, it is essential to define strategies to jointly allocate resources and place processing to obtain desired performance, for example, in terms of minimization of the operating cost of the constellation, or of reduction of the energy usage on ground stations taking full advantage of in-orbit operations. The definition of such strategies has an increased difficulty if compared to similar solutions designed for terrestrial applications, due to the fact that, in an orbital environment, link availability depends on the relative position between each couple of nodes, and this changes with time [16]. Thus, such strategies shall be defined by taking into account dynamic topologies.

It is also important to underline how satellite network made possible by ISLs can also support in-orbit training of machine learning (ML) models without having to transfer datasets of acquired images to a central node where training happens. In this way, it is possible to save bandwidth and, in case the central training node is given by a ground station, an energy saving on ground is also obtained, since the amount of training to be done on the Earth is reduced or even zeroed. In particular, at the best of my knowledge federated learning (FL) is the most investigated solution to train machine learning model in-orbit. In fact, FL provides for each satellite to train a local model only with a local dataset made of the images it acquired, and then to share only locally trained model with a node in charge of aggregating all local models to calculate a global model which is finally sent to all satellites[17, 18, 19, 20]. However, limited communication opportunities between satellites and the aggregating node may still limit the ability of this learning scheme to converge in a reasonable time. For this reason, it may be interesting to leverage satellite networks to make model sharing among couples of satellites possible, in such a way that each satellite has all the needed information to calculate the global model by itself, without having to rely on a central aggregating node.

This thesis contributes to the research in the application of OEC to EO missions by proposing and investigating communication strategies to support three different goals. First, an optimal strategy to jointly allocate resources and place processing in a time-varying topology representing an EO satellite constellation is proposed to minimize the operating costs due to transmission, storage and processing of acquired images, by taking advantage of both ISLs and data reduction associated to potential

in-orbit processing. Differently from other works, in my proposal satellites leverage ISLs to form orbit-wide networks where information can be exchanged, and in this way service processing is not restricted to the source satellite or its neighbours, but can happen on any satellite of the constellation. Since the proposed optimal solution is NP-hard, two heuristics are also introduced for applications to real orbital scenarios. Second, an optimal strategy to leverage at most in-orbit processing to minimize the energy consumption on ground stations due to image processing is proposed and investigated, and, again, since the optimal solution is NP-complete, a heuristic to be applied to real orbital scenarios is introduced and validated. Another strategy which jointly optimizes the on-ground energy saving and the on-board battery operative life is also proposed and investigated. Finally, a communication strategy to allow for training machine learning models directly in-orbit by means of distributed learning strategies is discussed. The performance of the proposed distributed learning-based schemes with respect to federated learning-based ones are investigated, in particular to evaluate the improvement in time needed to reach validation convergence when distributed learning is applied.

The rest of this thesis will be organized as follows. In Chapter 1, an overview of the literature concerning the research areas within which the thesis can be placed is provided. In Chapter 2, strategies to jointly allocate bandwidth and processing resources to minimize the total operating costs are proposed and evaluated against state-of-the-art solutions. Instead, Chapter 3 is dedicated to the proposal and discussion of strategies to minimize the energy consumption due to image processing on ground by leveraging in-orbit processing, again by numerically demonstrating their performance over state-of-the-art solutions. Chapter 4 will be devoted to the proposal and investigation of a communication strategy enabling distributed learning in networks of EO satellites. Finally, the main results of this thesis are summarized in the Chapter dedicated to Conclusions.

Chapter 1

Background and Literature Overview

In this chapter, I introduce the research context within which this thesis can be included. The different research areas related to this work are first introduced in Section 1.1. For each of them, a detailed discussion of the works available in literature is proposed in Section 1.2.

1.1 Motivation and Scope

The exponential development of digital technologies and services is completely coupled with the development of networks able to guarantee the possibility to interconnect a high number of devices displaced even in remote zones. For example, digital services supporting air and maritime transportation, or disaster prevention and relief, provide for a continuous interaction among remote devices and control stations placed in urban centres. However, terrestrial networks are not able to guarantee connection in remote areas, because of physical impediment (e.g., difficulties in placing antennas supporting airplanes or ships in the ocean), or scarce return of investment, since antennas in remote areas support only a small number of users. A solution to this issue can be given by satellite constellations in Low Earth Orbit (LEO), which can assure global coverage while maintaining low latency with respect to older Geostationary Orbit (GEO) telecommunication constellations. For this reason, commercial solutions like SpaceX Starlink[21] or Amazon Kuiper[22] currently being designed and deployed are thought to assure high bandwidth, low latency connectivity everywhere. In general, Non-Terrestrial Networks (NTNs) are gaining more and more importance because of their potential application in 5G[23] and future 6G[24], in particular integrated to terrestrial or aerial networks (e.g., networks of unmanned aerial vehicles or high altitude platforms) to obtain what is known as Satellite-Air-Ground Integrated Networks (SAGINs)[25].

However, connectivity could be not enough. In fact, recent applications rely always more on offloading tasks to a device different from the user one (e.g., Artificial Intelligence-based or Internet of Things applications), with constraints in the latency experienced by the user[26]. Thus, even though satellites can guarantee connectivity in remote areas, since the application requires a processing far from the user (e.g.,

in cloud), the latency could exceed the Quality of Service (QoS) requirements. An innovation, thus, can be given by extending the Multi-access Edge Computing (MEC) techniques to the space segment, to leverage satellites not only to guarantee connectivity, but also to provide computational capacity to execute services near to the user. This technique has been introduced in literature under the name of Orbital Edge Computing (OEC)[1, 26].

This innovation can lead to advantages not only for terrestrial users, but it can also support Earth Observation (EO) missions to overcome currently issues related to the transmission of a high amount of data to ground stations only during the short visibility time. In fact, an EO satellite currently acquires images which are then stored in the on-board satellite memory, until it passes over a ground station, when it shall transmit a high amount of data in a short time[1]. Thus, a high data rate is requested, but this is constrained by the power available on the satellite, which is in turn constrained by the dimensions and masses of solar panels and batteries. However, not all data transmitted to the ground are useful to the application. For example, for a hypothetical service providing optical imagery of cities, images where clouds cover the scene of interest are useless, but nowadays they are still transmitted to the ground station where they are finally discarded. This is a waste of bandwidth resource which is very scarce for downlink because of constraints on available power on the satellite. However, an on-board selection of the information to be transmitted thanks to an on-board data processing could avoid waste of bandwidth, since only useful information is transmitted[2, 3].

In this context, a constellation of EO satellites endowed with Edge Computing capabilities can lead both to an increased efficiency in bandwidth usage, since processed data (i.e., useful information only) is transmitted to the ground station, and to a reduction of the time needed to deliver information to the ground station, since inter-satellite links (ISLs) among satellites of the constellation can be used to transfer the data from the satellite which gathered it to the satellite in visibility of the station at any time. This is different from the current scenario where the source satellite waits to pass over a ground station to transmit data, which can lead to a high delay between the information gathering and its availability on ground. However, to obtain these benefits it is necessary to deploy a strategy able to decide where data shall be processed (i.e., in-orbit or on ground and, in the former case, on which satellite of the constellation) and, jointly, determining the route the information has to follow.

A further important benefit of OEC can be found in the saving in energy consumption on ground stations obtained when data are processed in-orbit. In particular, it can be noticed that, once a satellite is in orbit, energy generation always happen (e.g., solar panels always generate power when exposed to solar radiation), and the amount of generated energy is only dependent on the exposition of solar panels to the sun. In other words, we allocate in advance an amount of energy by providing for a certain worst-case energy budget, and the satellite always makes it available, regardless of its actual use, and since it is generally generated through solar panels, it is completely sustainable energy. Instead, the energy consumption of ground stations is related to the actual energy request, being correlated to the amount of data to be processed, and that energy is generated on demand. In particular, the higher the processing on ground is, the higher the ground station

energy consumption is, and the more energy has to be produced to satisfy this need. Furthermore, energy available on ground is typically given by mixed resources, thus, it could be only in part renewable, if not completely non-renewable. Even in case ground stations are provided with totally renewable energy, any saving on the use of this energy can be made available to the energy grid. For this reason, by defining appropriate strategies to decide how information has to be routed within the satellite network made possible by ISLs, and where images have to be processed (taking into account limitations on transmission bandwidth, storage, processing capacity and energy available on board of satellites), it is possible to achieve green networking performance by minimizing the energy use on ground for data elaboration, while assuring that all images are processed.

Finally, under the application viewpoint, one of the key enabling solutions to extract information from acquired images, both on satellites and on ground, is given by deep learning techniques[2, 27]. However, the accuracy of these algorithms is strictly related to the availability of large datasets for training purposes[28]. In case of EO-related applications, these datasets involve the availability of a high number of satellite images on the device where model are trained. For example, in this scenario we would have to transfer all the acquired images to the training node (i.e., a ground station or a specific satellite), where model training is executed. However, this would again require for a high amount of bandwidth, i.e., high transmission data rate to transfer a high amount of data in a short visibility time. Another solution could be given by making satellite sharing their own datasets in such a way that each satellite can train the model by itself on a dataset given by the union of its own dataset and the ones appertaining to the other satellites in the constellation. However, this solution can be again limited by the available bandwidth and by the computational capacity available on-board, since training a model on a larger dataset requires an increased computational effort. Instead, federated learning can be fruitfully leveraged in this scenario, since this technique provides for each satellite to train a local model only with its own dataset, and then to share its trained model with a central server, which receives models trained by the different satellites and aggregates them into a new global model which is finally shared with all the satellites, and this repeats until convergence is reached[17]. It is important to underline that this solution is more appropriate in the orbital environment because only the models are shared instead of all datasets, and models have a reduced size with respect to datasets of satellite images. Furthermore, local training happens with a reduced amount of data (i.e., a single satellite dataset, being obviously smaller than the union of the datasets appertaining to all satellites). However, since communication with a ground station (or, in general, with a node acting as a central parameter server) is limited by short visibility time, local model gathering and consequent global model transmission to all satellites may need a long time because of limited communication windows, and this has a negative impact on the time needed to reach model convergence, which is strictly related to the completion of these model sharing rounds. For this reason, a distributed learning solution where satellites share their own models among themselves, without leaning on a central parameter server, is worthy further investigation.

1.2 Literature Overview

From the scope of this thesis presented in Section 1.1, it follows that this research work can be framed into several research areas. First, since I am exploring the possibilities offered by a satellite network, this work is related to NTNs and their integration with other networks to obtain the SAGINs. However, since I am providing for endowing satellites with on-board processing capabilities, this thesis also falls into the research field related to the extension of edge computing capability to satellites. Finally, since I am focusing on the specific application to EO, this thesis can be also framed in the context of the exploitation of mega-constellation to in-orbit process EO data, with a particular focus on research related to energy aspects in this specific field and on how machine learning can be leveraged within EO constellations.

1.2.1 Non-Terrestrial Networks and Space-Air-Ground Integrated Networks

Research on NTNs is mainly related to the potential leverage of satellites in the context of 5G and 6G. Authors of [23] provide a survey on how such satellite networks may be leveraged in 5G. In particular, this work first provides a description of NTNs, then moving to the discussion of the use cases and architectures that these networks may assume in the 5G context, underlying their fundamental role in guaranteeing service continuity (i.e., 5G can be accessed from remote areas not covered by terrestrial networks by leveraging NTNs), ubiquity (i.e., making the 5G network available even though terrestrial networks are not because of, for example, natural disasters) and scalability (i.e., traffic can be offloaded to the NTN from the terrestrial one). This work also includes an overview of the activities by 3GPP on NTNs in 5G, topic further explored in [29] with a focus on design aspects related to standardization. Finally, the survey identifies mobility, propagation delay and radio resource management as open issues in the leverage of NTNs in 5G, and it introduces the importance that such networks may have in future 6G applications. In particular, the same authors propose a scheme to deal with the problem of radio resource management in [30].

The application of NTNs to 5G and 6G is also the topic of survey [31], which focuses on the integration of NTNs and Terrestrial Networks in Space-Air-Ground Integrated Networks (SAGINs), whose space component is given by satellites, while the air component is made by unmanned aerial vehicles (UAVs) and High Altitude Platforms (HAPs). In this survey, the integration of such networks in the 5G and 6G context are investigated under the point of view of the time evolution, of use cases (i.e., Internet of Things and MEC) and architecture (in particular, physical, media access control and network layer), providing the state-of-the-art of both academic and industrial research and development effort.

More on satellite networks to support 6G can be found in [32], whose authors shine a light on both the possibilities opened by NTNs in the 6G context and on the design challenges of such networks to obtain the desired performance in terms of latency and coverage. The same authors, in [33] proves how satellites can support millimeter wave (mmWave) communication with high capacity, and for this reason they represent an enabling technology in the implementation of 6G.

All the previously mentioned articles underline how latency represents one of the most important issues in NTNs and SAGINs. Consequently, strategies to deal with latency appears to be one of the most explored topics in literature. For example, the authors of [34] identify some of the most important issues related to LEO mega-constellations integrated with ground networks (e.g., latency, jitter, unstable routing and scarce network reachability) and formulate and solve an optimization problem aiming to integrate the two network segments while minimizing latency and ensuring routing stability. Latency is also the focus of the work [35], presenting a solution providing for both flow allocation and the selection of cloud or satellite relay servers to ensure real-time communications exploiting mega-constellations.

It is also interesting to notice an increasing interest in the application of Artificial Intelligence (AI) techniques for resource allocation and network management in NTNs and SAGINs. For example, the ANChOR project[36] financed by the European Space Agency (ESA) provides for an AI-based tool to orchestrate resources in 5G satellite and terrestrial networks by following a data-driven approach[37]. This can be extended also to beyond 5G (B5G) and 6G networks, for example by leveraging techniques based on Long Short-Term Memory (LSTM)[38]. This approach represents an extension to the orbital domain of AI-based resource allocation strategies for terrestrial network function virtualization (NFV), providing for the use of LSTM to predict the future amount of traffic, and, consequently, the number of cores to be allocated for a virtual network function instance, as proposed by authors of [39, 40, 41]. Instead, in the specific context of 6G, AI can be applied to integrate terrestrial networks and NTNs while improving the energy-efficiency of maritime networks[42].

Several applications of AI techniques to NTNs and SAGINs provide for the use of reinforcement learning (RL)[43]. For example, authors of [44] propose a Q-learning-based strategy to allocate capacity resources in satellites networks made by three layer, given by satellites in LEO, Medium Earth Orbit (MEO) and GEO. In work [45], RL is leveraged to obtain a routing algorithm in LEO satellite networks. Similarly, in [46], a strategy based on deep reinforcement learning (DRL) to deal with the multi-commodity flow problem in NTNs made by LEO satellites is proposed. Instead, authors of work [47] propose a solution based on DRL to deal with traffic offloading, since traditional strategies appears to unsatisfactorily cope with traffic offloading in a context of high dynamism related to both topology and traffic. Finally, in the specific context of the use of SAGINs in 5G, B5G and 6G, resources to new users can be reserved by means of RL-based network slicing, as described in work [48].

Another AI-based technique that appears to be often applied in the context of NTNs and SAGINs is Federated Learning (FL)[49]. For example, in case NFV is used in SAGINs, authors of [50] propose a FL-based algorithm to deal with the embedding of service function chains (SFCs) in SAGINs, showing significant improvements in terms of revenues, revenue-cost ratio and acceptance rate with respect to other state-of-the-art solutions. Instead, in work [51], FL is leveraged to allow for anomaly detection in traffic within a SAGIN. Finally, the application of FL in the context of B5G and 6G SAGINs has been also investigated in literature. In particular, authors of [52] first introduce how FL-based strategies can be applied to optimize different objectives by appropriately managing network resources; then,

the specific application to traffic offloading is evaluated. Instead, authors of [53] demonstrate the ability of FL-based NTNs for 6G to guarantee lower communication overhead and latency than state-of-the-art solutions.

1.2.2 Edge Computing in orbital environment

As far as edge computing applied to satellite networks is concerned, this technology can be leveraged to both provide service to terrestrial mobile users by satellite or support EO missions. In this subsection, I will only introduce works related to the former context, while research on the latter application will be discussed in detail in the next subsection, dedicated to the literature on leveraging satellite networks and in-orbit processing in EO missions.

Authors of [54] carried out a study of architectures, technologies and challenges related to the extension of MEC to SAGIN. MEC techniques are leveraged in [55] to improve the QoS in networks made by satellite and terrestrial nodes, by means of appropriate resource allocation, computation offload and task scheduling schemes. Strategies to deal with resource scheduling are also the topic of work [56], which proposes two algorithms to allocate computing resources and manage ISLs by leveraging advanced K-means algorithm and a spanning tree strategy based on breadth-first-search. However, authors of [57] shine a light on the fact that most of the research on task-offloading in satellite networks does not take into account the actual ability of the different satellites in taking part into offloading, related to how loaded they are. For this reason, the authors propose an algorithm based on particle swarm optimization to schedule tasks taking into account delay and energy consumption. Another interesting strategy to tackle the problem of computation offloading is proposed in [58], whose authors propose a game-theoretic scheme to obtain an optimal solution under the point of view of response time and energy consumption. Instead, authors of [59] propose an online algorithm to place processing within satellite networks in such a way to provide robustness-aware service coverage.

Several research papers focus on the support that satellite networks endowed with edge computing capability can give to terrestrial Internet of Things (IoT) applications. In particular, architecture and scheduling strategies to support IoT through satellite edge computing have been proposed[60]. Authors of [61] point out how, by endowing satellites with computing capability to support IoT, it is possible to obtain an improvement of the QoS in terms of time needed to accomplish the computation and energy consumption. However, this improvement is dependent on the task offloading strategy applied. For this reason, there is a high research interest in designing such strategies, taking into account the peculiar characteristics of IoT applications. For example, authors of [62] propose a scheme based on directed acyclic graphs and on an attention mechanism combined with proximal policy optimization to deal with the problem of multi-task offloading in satellite IoT, allowing for a cost-effective resource allocation even in case tasks are not independent from each other. Instead, work [63] is devoted to the proposal of a computation offloading and resource allocation algorithm to minimize the energy consumption of terrestrial terminals. Finally, it is interesting to notice how satellite edge computing can also support Internet of Vehicles (IoV), as it is shown in [64], which proposes a framework where a SAGIN endowed with edge computing capability supports the IoV, with

the leverage of a deep imitation learning strategy to deal with task offloading and caching to optimize both the time needed to complete the task and the use of satellite resources.

Research on the extension of MEC to the orbital environment is also strictly related to the research on the application of NFV to satellite networks. In fact, NFV can be seen as a particular application of in-orbit processing, where the computation is used to provide network functions. For this reason, it may be interesting to provide an overview of the state-of-the-art on this field. For example, authors of [16] first propose an optimization problem to deal with the virtual network function (VNF) placement and routing in a satellite network, modelled by means of a software defined time evolving graph; then, an algorithm to solve the proposed problem is introduced and evaluated. The same authors further explored this research strand by proposing a method to obtain the optimal solution of the VNF orchestration problem in orbital environment[65], and a linear programming problem to deal with service provision and resource allocation in a LEO network aiming to minimize the use of ISLs because of their instability[15]. Another interesting approach to place VNFs is presented in [66], where game theory is leveraged to obtain a placement able to minimize the deployment cost while maximizing the number of served user requests. Instead, authors of [67] present a genetic algorithm-based strategy to place VNF by balancing delay and resource allocation. Finally, authors of [68] focus on an intent-driven approach to the VNF placement, where user intents are parsed by means of a bidirectional LSTM network.

NFV is in turn strictly related to software-define networking (SDN). It may be thus interesting to explore how this technique has been studied for applications in the aerospace domain. For example, authors of [69] provide a first proposal for an implementation of SDN in networks of UAVs and electric vehicles, where a satellite acts as a network controller. Instead, authors of [70] underline how the application of SDN to NTNs can allow to reach the objectives of 5G and B5G networks. However, architectures based on satellites have peculiar characteristics making traditional terrestrial algorithms not directly applicable in this context. For this reason, they propose an *ad-hoc*, centralized routing algorithm, able to take into account the QoS requirements, based on the Bresenham's and Dijkstra's algorithms. However, authors of [71] point out how a centralized strategy introduces overhead related to reconfiguration and migration. For this reason, they focus on a strategy to optimally place the controller and assign satellites to the controller, and they evaluate the amount of this overhead. Also authors of [72] focus on the problem of controller placement, and propose a strategy to deal with it by taking into account the typical properties of a satellite constellation (e.g., the fact that satellites move and links are not always available). A way to deal with the controller and placement problem in this context with a reduced computational effort is given by [73], whose authors propose techniques based on simulated annealing and genetic algorithm to solve this problem with near-optimal results. Finally, authors of [74] present a software-based testbed to study SDN in networks integrating terrestrial nodes and satellites occupying orbits different from GEO.

1.2.3 Satellite Networks and Edge Computing in Earth Observation Constellations

Mega-constellations are gaining more and more interest in supporting EO applications. For example, authors of [75] point out how the current scheme providing for EO data gathering made by GEO satellites or ground stations leads to high latency and shows scarce scalability. For this reason, they propose a solution to reduce EO data delivery latency by taking advantage of LEO mega-constellations, considering the dynamical topology of such a network. Always to deal with the problems related to delivery time and scalability, authors of [76] present both optimization-based and heuristic-based solutions leveraging multi-path in LEO satellite networks for EO applications. Work [77] can be placed into the same research strand, since its authors propose an optimization-based strategy to deliver data to ground by leveraging networks of EO satellites endowed with ISLs, by taking into account node mobility and resource constraints. Instead, authors of [78] focus on proposing a strategy to increase the throughput in the context of EO by jointly optimizing image acquisition and transmission schedule. Similarly, the author of [79] proposes different algorithms to tackle the problem of observation and communication scheduling by considering random requests for acquisition tasks, while a different optimization problem formulation, neighbourhood search and genetic algorithm are also proposed in [80].

It may be interesting to understand how this technology may have even a stronger impact if applied not only on a single satellite, but within a constellation. Under this point of view, authors of [75] introduced a solution to obtain lower EO data delivery latency by taking advantage of mega-constellations, while authors of [6] improved the timeliness of EO data by leveraging load-balancing in EO constellations endowed with in-orbit processing capability. Since on-board data processing requires energy, authors of [11] propose an algorithm to minimize the energy consumption on satellites while guaranteeing desired latency. Instead, the focus of the algorithm proposed by authors of [12] is the optimization of the use of satellite batteries to extend their operating life by minimizing the depth-of-discharge (DoD). However, this may lead to an increased latency in service processing, at the expense of possible requirements on the QoS in terms of timeliness.

Given the possibilities opened by leveraging satellite networks in the context of EO in case satellites have the possibility to process data on-board, it may be interesting to discuss the state-of-the-art of on-board processing technology, mainly related to AI-based applications. In this context, authors of [81] identify processing capability and radiation environment as potential issues preventing the application of AI in-orbit, and at the same time discuss how satellites designed to have a shorter operative life and mainly based on commercial-off-the-shelf (COTS) components may foster the leverage of AI algorithms on-board. In-orbit processing capability has been demonstrated by the ESA Φ-Sat-1 Mission [2], where AI techniques are run on a satellite to select data to be downlinked, discarding the ones not useful to the application (e.g., by recognizing the presence of clouds in acquired images). Further demonstration of the opportunities opened by running AI algorithms on-board of EO satellites will be provided by ESA Φ-Sat-2 Mission[82]. Among these, AI-based image compression scheme will be also experimented on-board[83]. Such

demonstration reinforces the research on AI algorithms to be executed in-orbit. In particular, an algorithm for ship detection on board of synthetic aperture radar (SAR) based on deep learning has been proposed by authors of [84], while authors of [85] present a pipeline to process hyperspectral images on board of cubesats. Work [86] focuses on the proposal of an algorithm based on convolutional neural networks (CNNs) to on-board detect volcanic eruption by processing acquired multispectral images. Instead, the robustness of the application of CNN on hyperspectral images when atmospheric perturbation or noise is added is analysed by authors of [87]. However, it is important to underline that in-orbit processing consume energy on-board of satellites. For this reason, authors of [88] investigates the application of spiking neural networks in this context as an energy-efficient solution to process data on satellites, in particular in case of scarce energy availability, like in cubesats. Finally, it may be interesting to discuss how these algorithm may be implemented on board of satellites. Field programmable gate array (FPGA) appears to be the most interesting technology to implement such strategies. For example, authors of [89] leverage FPGA to implement an on-board ship detection scheme on SAR images. Similarly, a solution to detect ship, wind and sea state on SAR images directly on-board by means of an FPGA and a CPU is proposed in [90]. Instead, authors of [91] propose the leverage of FPGA to obtain an hardware acceleration of quantized CNN dedicated to the on-board classification of cloud coverage in acquired images.

Research works cited in the previous paragraph mainly provides for the application of AI strategies on a single satellite. However, it would be interesting to leverage the computational capacity available within the entire constellation, by appropriately making use of the satellite network made available by ISLs. On this research strand, it is possible to identify a high interest in federated learning techniques in LEO satellites, which can be frutifully applied in the EO mission context. A comprehensive presentation of the state-of-the-art federated learning strategies in mega-LEO constellations is given by [17], where three federated learning scenarios in orbital environment are discussed, depending on the availability of links between a node acting as a parameter server (where local models are aggregated to update a global model) and the satellites, as well as the communication opportunities within the constellation itself. In particular, the paper identifies the scenario where inter-orbital ISLs are available as the most promising for further research. The same authors proposed a deeper investigation of the three federated learning scenarios in further works. In particular, in case of sporadic connection possibilities to a parameter server, as it happens in case no ISL is available or only a ground station takes part to the federated learning process, an asynchronous federated learning strategy has been proposed in [19]. Similarly, authors of [92, 93] propose an asynchronous FL strategy leveraging HAP as a parameter server. However, the convergence speed of such a scheme may be compromised by model staleness. As explored in [20], a solution to this problem can be given by leveraging the predictability in communication opportunities with the ground station, which allows for the proposal of a scheduling algorithm to optimally decide when the model parameter exchange between ground stations and satellites shall happen. Instead, in [18], authors focused on the case in which only intra-orbital ISLs are available, proposing a communication scheme enabling synchronous federated learning with a parameter server placed on the ground or in a satellite not appertaining to the constellation.

Chapter 2

Processing and Bandwidth Cost Optimization in Orbital Edge Computing satellite networks

In this chapter, I formalize and solve an optimal bandwidth and computing resource allocation problem in LEO satellite constellation for EO applications aiming to minimize the total operating cost due to data transmission, storage and processing. In order to deal with the complexity of the proposed optimization problem, I also present two heuristics requiring different computational effort. In the proposed problem formalization, processing can happen on any node of the network (i.e., either on the data source satellite, on any other satellite of the constellation or on ground station). This is an important difference with other strategies proposed in literature[9, 10, 11], providing for processing to happen either on the acquiring satellite or on ground stations. Instead, the strategies proposed in this chapter allows for leveraging computational capacity available not only on the single satellites, but within the entire constellation. Another difference is in the fact that while the proposed strategies aim to reduce the total operating cost, most of literature mainly focuses on the optimization of the time needed to deliver information on ground[6, 7, 8]. After having validated the proposed heuristics by comparing their results to the optimization problem ones, I apply them to a real orbital scenario, showing their ability to reduce total cost and, at the same time, also to reduce data delivery delay to ground with respect to state-of-the-art solutions.

Results shown in this chapter have been published in [94, 95].

2.1 Reference Scenario and Problem Statement

In the analysed scenario, I consider satellites to be arranged on different orbital planes. I also assume that satellites can communicate among themselves by leveraging ISLs. In particular, communication is not restricted to couples of satellites belonging to the same orbital plane, but can also happen among couples of satellites on different orbital planes, depending on the distance. Finally, I will consider each satellite to have a given processing capacity to elaborate data on-board, and a memory to store information.

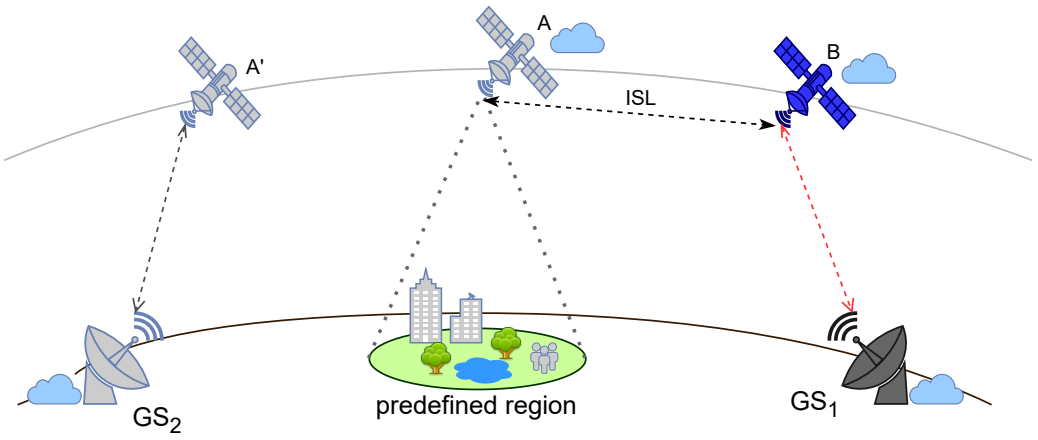

Figure 2.1. Example of the reference scenario.

In the considered scenario, the source of any service is placed into the constellation itself, while, since I am studying EO applications, data shall be always downlinked to the Earth, which can be seen as a sink for all services. In particular, whenever a satellite flies over a predefined region, it acquires images of it. At this point, it can either process the image on-board, store it or send it to a visible ground station (if any) or to another satellite in the constellation. In case data is sent to another node, the receiving node can in turn process the received data, store it in its memory or send it to another node, and so forth. To further clarify this scenario and its difference with the state-of-the-art operational scheme, let me introduce the example shown Fig.2.1. In this case, satellite A flies over a region to be monitored and acquires images. In the current state-of-the-art operations, it can only store the acquired information in its memory until it reaches position $A^{'}$ in its orbital motion, when it enters the visibility region of ground station GS_2 and has to downlink the data. Instead, in my solution satellite A can leverage the ISL with satellite B and transfer data to it, in such a way that the information can be delivered to ground station GS_1 passing through B, reducing the delivery delay. Furthermore, in my solution both A or B, as well as the ground stations, can process data. This is particularly important in case, for example, the bandwidth on link B-GS_1 is enough to transmit a processed data, but not to host an unprocessed one. In this case, by processing on either A or B, link B-GS_1 would be able to host data and, thus, would again allow for obtaining a reduction in information delivery delay with respect to the state-of-the-art operations.

It is then evident that appropriate strategies to be followed by satellites in choosing which action shall be taken while dealing with a produced or received piece of information can be designed to obtain any desired behaviour. In particular, in this chapter I focus on proposing centralized joint resource allocation and processing placement strategies aiming to minimize the overall operating cost by leveraging the satellite network and on-board processing capacity, in such a way that an image generated by a satellite can be processed on any network node (i.e., the source satellite, any other satellite or a ground station), optimizing the use of valuable

resources like available bandwidth and on-board processing capacity. The proposed strategies are centralized since they rely on the fact that link availability between each couple of nodes is known by orbital mechanics and, for the same reason, image acquisition periods of each satellite are scheduled by control stations on Earth. Furthermore, both topology and service generation repeats periodically, because of the periodic motion of both Earth and satellites. Thus, it is possible to run algorithms on Earth to offline define the route and processing node for each service in order to optimize any desired metric, and then instruct the satellites to behave as defined by the algorithm output.

2.2 Network and Service Modeling

After having described the application scenario in Section 2.1, let me translate it into an abstract model. In particular, I will introduce a model for the dynamic topology and one for the services. Sets and parameters introduced in the following description are summarized in Tab.2.1.

2.2.1 Dynamic topology

If compared to routing and processing resource placement in most terrestrial networks, the orbital scenario has an increased complexity given by the satellite and Earth motion making topology time-varying. However, dynamical topology and service generation repeats after a period T_c (i.e., it can be assumed a cyclostationary behaviour for both topology and service generation), since both the satellite motion and the Earth rotation are periodic. Formally, this scenario can be generalized by using a time-evolving graph $\mathcal{G} = (\mathcal{N}, \mathcal{E}, \mathcal{T})$[15], a mathematical structure able to model the dynamic behaviour of the network topology due to satellite and Earth motion. In particular, in a time-evolving graph, nodes (identified by the set $\mathcal{N}$), edges (identified by the set $\mathcal{E}$) and layers (identified by the set $\mathcal{T}$) (see Fig.2.2) can be distinguished.

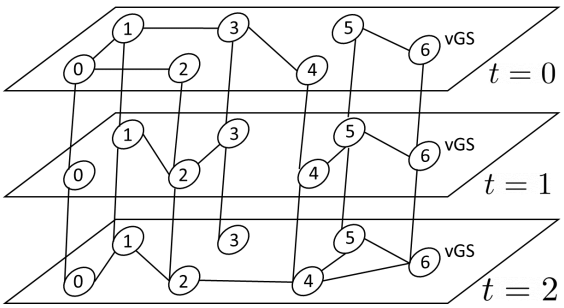

Figure 2.2. Example of time-evolving graph referring with $T = 3$ time periods (i.e., layers), $N_S = 7$ nodes.

Table 2.1. Reference scenario sets and parameters

Set or parameter	Description
$\mathcal{G} = (\mathcal{N}, \mathcal{E}, \mathcal{T})$	time-evolving graph identified by nodes (set $\mathcal{N}$), edges (set $\mathcal{E}$) and layers (set $\mathcal{T}$)
T_c	cyclostationary period for both topology and service generation
τ	discrete time cycle duration
T	number of time cycle of duration τ in a cyclostationary period
l_t	layer in the set $\mathcal{T}$, with $t \in [0, \ldots, T-1]$ (i.e., time cycle)
n_i	node in the set $\mathcal{N}$, with $i \in [0, \ldots, N_S - 1]$
N_S	number of nodes (either satellites or vGS)
Γ_i	processing capacity associated to the n_i node (in Mbps)
γ_i^p	cost to be paid for unit processing on the n_i node (in \$/Mbps)
vGS	Virtual Ground Station represented by the n_{N_S-1} node
$\mathcal{E}_e$	intra-layer edge set (i.e., transmission links)
$\mathcal{E}_m$	inter-layer edge set (i.e., memory links)
$e_{i,j}^t$	intra-layer edge (i.e., transmission link) in $\mathcal{E}_e$, with $i, j \in [0, \ldots, N_S - 1] \mid n_i, n_j \in \mathcal{N}$
$C_{i,j}^t$	capacity associated to the $e_{i,j}^t$ transmission link
$\gamma_{e_{i,j}}$	cost of the unit transmitted data amount on the $e_{i,j}^t$ transmission link (in \$/Mb)
m_i^t	inter-layer edge (i.e., memory link) in $\mathcal{E}_e$, with $i \in [0, \ldots, N_S - 1] \mid n_i \in \mathcal{N}$
M_i	data storage capacity of the m_i^t memory link
γ_{m_i}	cost to be paid for the unit data stored on the i-th node (i.e., on m_i^t memory link, in \$/Mb)
Σ	set of all generated services
f^h	service in Σ, with $h \in [0, \ldots, N_T - 1]$
N_T	total number of generated services
f_s^h	f^h service source satellite index
f_0^h	f^h service pre-processing size
f_1^h	f^h service post-processing size
f_t^h	f^h service generation time cycle
f_d^h	number of time cycles after generation within which f^h service shall be delivered to the vGS

Layers are the structures of the time-evolving graph making possible to deal with dynamic topology. In particular, it is possible to discretize the time in cycles of duration τ during which topology remains fixed (i.e., links between couples of satellites or between a satellite and a ground station remain available during the complete time cycle). Thus, each layer $l_t \in \mathcal{T}$ with $t \in [0, \ldots, T-1]$, where T is the number of time cycles of duration τ in a cyclostationary period of duration T_c, represents a particular topology realization (i.e., links between nodes at the time cycle to which the layer is referred).

As far as nodes are concerned, each node $n_i \in \mathcal{N}$ with $i \in [0, \ldots, N_S - 1]$, where N_S is the total number of nodes, represents either a satellite or any ground station enabled to receive data on the planet Earth. The i-th node is endowed with a

processing capacity Γ_i (in Mbps), to the use of which a cost γ_i^p (in \$/Mbps) is associated, representing the cost to be paid for the unit processing amount. It is not necessary to introduce a node for each ground station on Earth enabled to receive and elaborate data from the constellation. In fact, since the destination is always any enabled ground station on Earth, it is only necessary to introduce a node representing the sink of information (hereafter named as Virtual Ground Station, *vGS*), which will be connected, during each time period, to all the satellites having visibility of any enabled ground station within that period. It is important to underline how this choice does not affect the solution of the joint routing and processing placement problem. In fact, I will consider that all the enabled ground stations are equal (i.e., having enough memory and processing capacity to process all services, as well as same operating costs) and that all the satellites being able to communicate with a ground station in a certain time are connected to the *vGS* node. Thus, since any ground station is always the destination for all services, this does not affect the routing within the satellite constellation. As an arbitrary convention, the index of the *vGS* node will be represented by $i = GS = N_S - 1$.

Finally, the set of edges $\mathcal{E}$ is given by the union of two sets, one representing intra-layer edges ($\mathcal{E}_e$), the other inter-layer edges ($\mathcal{E}_m$). Intra-layer edges $e_{i,j}^t \in \mathcal{E}_e$, with $i, j \in [0, \dots, N_S - 1] \mid n_i, n_j \in \mathcal{N}$ and $t \mid l_t \in \mathcal{T}$, represent the availability of a transmission link from node n_i to node n_j during the t-th time period (i.e., they have value equal to 1 if the link is available, 0 otherwise). The link availability is known by orbital mechanics. In particular, I assume an Additive White Gaussian Noise channel[15], and I consider a link between two satellites to be available whenever the satellites have a distance $d_{i,j} \leq d_{i,j_{max}}$, where $d_{i,j_{max}}$ is the maximum distance for which a minimum desired $Eb/N0|_{min}$ ratio is assured and is defined as follows:

$$d_{i,j_{max}} = \frac{c}{4\pi\nu}\sqrt{\frac{P_t G^2}{k_B \; T_s \; R_{i,j} \; Eb/N0|_{min}}} \tag{2.1}$$

where c is the speed of light, ν is the carrier frequency, P_t is the transmission power, G is the antenna gain, k_B is the Boltzmann's constant, T_S is the system noise temperature and $R_{i,j}$ is the transmission data rate. Instead, I assume a link between a satellite and a ground station to be available whenever the satellite is visible from a ground station with a minimum elevation angle El_{min}. Link availability influences the amount of data that can be transmitted. In particular, even though transmissions have a fixed data rate, actually each transmission link is associated to a capacity $C_{e_{i,j}^t} = R_{i,j} \cdot \hat{\tau}_{i,j}^t / \tau$ in Mbps, whose value depends on the transmission data rate $R_{i,j}$ of i-th node towards the j-th node and the actual visibility time $\hat{\tau}_{i,j}^t$ between the couple of nodes during the t-th time period, and a cost $\gamma_{e_{i,j}}$ (in \$/Mb) representing the cost to be paid for the unit transmitted data amount between the two nodes. Instead, inter-layer edges $m_i^t \in \mathcal{E}_m$ represents a memory link on the i-th node. In particular, if a service occupies the m_i^t edge, it means that it is kept in the memory of the i-th node during the full t-th time period, until the beginning of the next time period. Each memory link m_i^t is associated with a capacity M_i in Mb, representing the maximum data storage capacity on the i-th node, independent of time period, and with a cost γ_{m_i} (in \$/Mb) representing the cost to be paid for the unit data stored on the i-th node. It is important to underline how the

obtained time-evolving graph is a quasi-static representation of the dynamical orbital environment. In fact, instead of having a graph where each node simply represents a satellite (i.e., a graph with N_S nodes) and the presence of edges between nodes is a function of time, each node of the graph does not simply represent a satellite, but a satellite at a given time cycle (i.e., a graph having $N_S \cdot T$ nodes), where the presence of edges is not function of time and the actual visibility time between satellites or satellites and vGS is embedded in the capacities associated to each edge. Let me clarify this claim with an example. Let me suppose to have two satellites i and j and a cyclostationary period made of $T = 2$ time cycles of duration τ. Let me suppose that by analyzing the satellite orbital motion, their distance is small enough to assure communication with the desired Eb/N0 from the beginning of the cyclostationary period to time $\tau/6$. In my formalization, instead of having a graph with two nodes representing the two satellites and a time-dependant edge, which is available only for $\tau/6$ with capacity $R_{i,j}$, there will be a graph with four nodes, the first two representing the two satellites in the time cycle from time 0 to time τ, and the second two representing the two satellites in the time cycle from time τ to time 2τ. Since the couple of satellites can communicate only for a time equal to $\tau/6$ from the beginning of the repeat cycle, an edge connecting the nodes representing the two satellites in the first time cycle is only added, while the second couple of nodes will have no edge. Furthermore, in order to take into account that communication is possible only during a fraction $\hat{\tau}^0_{i,j} = \tau/6$ of the first time cycle, the added edge will not have a capacity equal to the transmission data rate, but a reduced capacity given by $C^0_{i,j} = R_{i,j} \cdot \hat{\tau}^0_{i,j}/\tau = R_{i,j}/6$. It is important to underline how the power of this formalization is given by the fact that the time-varying properties of links within the constellation have been translated into *ad-hoc* nodes and edges, in such a way that each layer of the time-evolving graph, which includes all nodes and edges referred to the same time cycle, represents a snapshot of the topology in time, and the sequence of these snapshots, i.e., the full graph, returns a representation of the dynamical topology enabling the application of classical graph algorithms without any modification to deal with time-varying properties. This simplifies both the formulation of the optimization problem and the proposal of heuristic algorithms, since the classical Dijkstra's algorithm can be leveraged, as it will be discussed in the following sections.

2.2.2 Services

After having introduced the time-evolving graph, let me elaborate more on services generated by the satellites. Service generation is related to image acquisition which happens when a satellite flies over a region of interest. But the occurrence of this event is related to the relative position of satellites and Earth, and since their motion is periodic, also service generation will be cyclostationary, and it will have the same period as the topology variation. The cyclostationary behaviour of both topology and service generation, characterizing the Earth Observation application I am considering in this thesis, allows to only have to consider allocations in a cyclostationary period, since these will repeat all mission long. In particular, I suppose that each satellite can generate a service requiring a processing given by a single task, aiming at reducing the data size. Obviously, what is discussed in this chapter can be extended to take

into account services requiring more than a single task. Each service $f^h \in \Sigma$, where Σ is the set of all generated services in a cyclostationary period and $h \in [0, \ldots, N_T]$, where N_T is the total number of generated services in a cyclostationary period, is described by a tuple $\{f_s^h; f_0^h; f_1^h; f_t^h; f_d^h\}$, where f_s^h represents the source node, f_0^h the service size before processing in Mb, f_1^h the task size after processing (again, in Mb), f_t^h the time cycle during which the service is generated and f_d^h the maximum number of periods after the service generation one within which the service shall be delivered to a ground station.

2.3 Optimization Problem

The design of a framework able to jointly deal with routing and processing placement in an EO constellation endowed with OEC capabilities can be based on different tools. The most rigorous one is given by an optimization problem, which I will formalize in this section, starting from the formalization given in [15], aiming to orchestrate virtual network function over a LEO constellation modelled as a time-evolving graph by minimizing the use of ISLs, and by extending it to take into account the data reduction due to processing. Instead, the main idea underlying my optimization problem definition is that in the scenario described in Section 2.2 it is possible to identify three operating costs due respectively to data transmission, storage and processing. Thus, an optimization problem aiming to propose a routing and processing placement able to minimize a total operating cost given by the sum of transmission, memory and processing costs can be defined. From the scenario description it is also easy to identify the constraints to be defined, mainly linked due to transmission link bandwidth, storage and processing capacity of each node. Furthermore, appropriate constraints to assure that all services are processed (by any satellite or ground station) and to guarantee flow conservation on nodes are necessary. However, routing and processing placement are coupled and this could result in non-linear constraints (e.g., link bandwidth usage depends on if that link is crossed by a processed or unprocessed service, since data size would be different). For this reason, in order to obtain an Integer Linear Programming formulation (ILP) for the optimization problem, two virtual links to deal with different data size before and after processing will be introduced, and their mapping to physical links leads to additional constraints. This formulation is similar to the Service Function Chain (SFC) mapping in elastic optical network[96]. However, the problem stated in this chapter differs from well studied virtual network mapping ones because of the dynamicity of the substrate network and because of the data reduction which happens when processing happens, which at the best of our knowledge has not been considered in other works on virtual network mapping problem in dynamic networks like satellite ones. In particular, the introduction of virtual links leads to a virtual topology composed by three nodes and two links (see Fig.2.3). Two of the virtual nodes have a straightforward mapping with the physical topology, since they correspond to the source satellite at generation period and the destination (i.e., the virtual ground station node at the deadline period). Instead, the remaining virtual node represents the processing node, which maps to the physical node chosen as processing node. As far as the two virtual links are concerned, they connect,

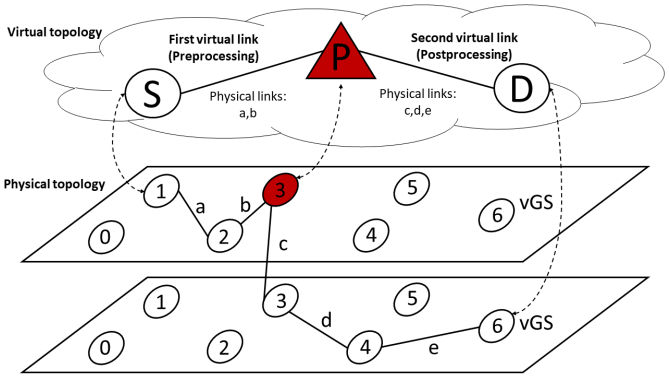

Figure 2.3. Example of virtual and physical topology mapping. Virtual source node S is mapped to node 1 of the physical topology, while virtual destination node D is mapped to node 6 (GS) of the physical topology. Virtual processing node P is mapped to physical node 3. For this reason, the first virtual link is mapped on links *a*, *b*, since they carry unprocessed information, while the second virtual link is mapped on links *c*, *d*, *e*, since they carry processed data.

respectively, the source virtual node to the processing virtual node and the processing virtual node to the destination virtual node. Thus, the first virtual link will only host services with a data amount given by the pre-processing data size, while the second virtual link will only host services with a data amount given by the post-processing data size. Finally, in this model I will consider negligible contention and the limitations due to power available on-board of satellite will be included in the transmission and computational capacities.

Let me introduce the three binary decision variables of the optimization problem:

$$y_{i,j}^{t,h,p} = \begin{cases} 1 & \text{if the } p\text{-th virtual link of the } h\text{-th service is mapped on the} \\ & e_{ij}^{t} \text{ edge (transmission link)} \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

$$z_{i}^{t,h,p} = \begin{cases} 1 & \text{if the } p\text{-th virtual link of the } h\text{-th service is mapped on the} \\ & m_{i}^{t} \text{ edge (memory link)} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

$$w_{i}^{t,h} = \begin{cases} 1 & \text{if the } h\text{-th service is processed by the } i\text{-th node} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

The first two variables are linked to the routing part of the problem, dealing with transmission and memory links, respectively. Instead, the third variable refers to the processing resource placement of the problem.

The objective of the presented optimization problem is to minimize the total cost given by the sum of the following expressions:

$$\kappa_p = \sum_{h=0}^{N_T-1} \sum_{t=0}^{T-1} \sum_{i=0}^{N_S-1} w_i^{t,h} \cdot \frac{f_0^h}{\tau} \cdot \gamma_i^p \tag{2.5}$$

$$\kappa_e = \sum_{h=0}^{N_T-1} \sum_{t=0}^{T-1} \sum_{i=0}^{N_S-1} \sum_{\substack{j=0 \\ j\neq i}}^{N_S-1} \sum_{p=0}^{1} y_{i,j}^{t,h,p} \cdot f_p^h \cdot \gamma_{e_{i,j}} \tag{2.6}$$

$$\kappa_m = \sum_{h=0}^{N_T-1} \sum_{t=0}^{T-1} \sum_{i=0}^{N_S-1} \sum_{p=0}^{1} z_i^{t,h,p} \cdot f_p^h \cdot \gamma_{m_i} \tag{2.7}$$

where (2.5) represents the processing cost (i.e., the component of the total cost due to service processing), (2.6) the transmission cost (i.e., the component of the total cost due to service transmission), while (2.7) is the memory cost (i.e., the component of the total cost due to service storage).

Let me now introduce the constraints of the optimization problem as follows:

$$\sum_{h=0}^{N_T-1} \sum_{p=0}^{1} y_{i,j}^{t,h,p} \cdot f_p^h \leq C_{i,j}^t \cdot \tau, \;\; \forall i,j \in [0..N_S-1], \; j \neq i, \; t \in [0..T-1] \tag{2.8}$$

$$\sum_{h=0}^{N_T-1} \sum_{p}^{1} z_i^{t,h,p} \cdot f_p^h \leq M_i, \;\; \forall i \in [0..N_S-1], \; t \in [0..T-1] \tag{2.9}$$

$$\sum_{h=0}^{N_T-1} w_i^{t,h} \cdot f_0^h \leq \Gamma_i \cdot \tau, \;\; \forall i \in [0..N_S-1], \; t \in [0..T-1] \tag{2.10}$$

$$\sum_{i=0}^{N_S-1} \sum_{t=f_t^h}^{f_t^h+f_d^h} w_i^{t \bmod T,h} = 1, \;\; \forall h \in [0..N_T-1] \tag{2.11}$$

$$\begin{cases} z_{f_s^h}^{f_t^h \bmod T,h,0} + \sum\limits_{\substack{j=0 \\ j\neq f_s^h}}^{N_S-1} y_{f_s^h,j}^{f_t^h \bmod T,h,0} = 1 - w_{f_s^h}^{f_t^h \bmod T,h} \\ z_{f_s^h}^{f_t^h \bmod T,h,1} + \sum\limits_{\substack{j=0 \\ j\neq f_s^h}}^{N_S-1} y_{f_s^h,j}^{f_t^h \bmod T,h,1} = w_{f_s^h}^{f_t^h \bmod T,h} \end{cases}, \;\; \forall h \in [0..N_T-1] \tag{2.12}$$

$$\begin{aligned} &\left(1-\delta_{t,f_t^h} \; \delta_{i,f_s^h}\right)\left(1-\delta_{t,f_t^h+f_d^h} \; \delta_{i,N_S-1}\right)\left\{\left[\left(\left(1-\delta_{t,f_t^h}\right) z_i^{(t-1),h,0} + \sum_{\substack{a=0 \\ a\neq i}}^{N_S-1} y_{a,i}^{t,h,0}\right)+\right.\right. \\ &-\left.\left(\left(1-\delta_{t,f_t^h+f_d^h}\right) z_i^{t,h,0} + \sum_{\substack{b=0 \\ b\neq i}}^{N_S-1} y_{i,b}^{t,h,0}\right)\right] - \left[\left(\left(1-\delta_{t,f_t^h}\right) z_i^{(t-1),h,1} + \sum_{\substack{a=0 \\ a\neq i}}^{N_S-1} y_{a,i}^{t,h,1}\right)+\right. \\ &\left.\left.-\left(\left(1-\delta_{t,f_t^h+f_d^h}\right) z_i^{t,h,1} + \sum_{\substack{b=0 \\ b\neq i}}^{N_S-1} y_{i,b}^{t,h,1}\right)\right] - 2w_i^{t,h}\right\} = 0 \\ &\forall h \in [0..N_T-1], \; i \in [0..N_S-1], \; t \in [f_t^h..f_t^h+f_d^h] \; mod \; T \end{aligned} \tag{2.13}$$

$$\sum_{p=0}^{1}\left(z_i^{(t-1)\ mod\ T,h,p}+\sum_{\substack{a=0\\a\neq i}}^{N_S-1} y_{a,i}^{t\ mod\ T,h,p}\right)\leq 1, \quad \forall h\in[0..N_T-1],\ i\in[0..N_S-1],\ t\in[f_t^h..f_t^h+f_d^h]\ mod\ T \tag{2.14}$$

$$\sum_{p=0}^{1}\left(z_i^{t\ mod\ T,h,p}+\sum_{\substack{b=0\\b\neq i}}^{N_S-1} y_{i,b}^{t\ mod\ T,h,p}\right)\leq 1, \quad \forall h\in[0..N_T-1],\ i\in[0..N_S-1],\ t\in[f_t^h..f_t^h+f_d^h]\ mod\ T \tag{2.15}$$

$$\sum_{p=0}^{1} z_{N_S-1}^{(f_t^h+f_d^h-1)\ mod\ T,h,p}+\sum_{p=0}^{1}\sum_{i=0}^{N_S-2} y_{i,N_S-1}^{(f_t^h+f_d^h)\ mod\ T,h,p}=1,\ \ \forall h\in[0..N_T-1] \tag{2.16}$$

(2.8) imposes that, during each time period, each transmission link cannot transmit more data than the maximum data amount transferable during a time period. Similarly, each node cannot store more data than its own storage capacity (2.9), and cannot process more data than possible, given the maximum processing capacity (2.10). (2.11) enforces that all services are processed, either by a satellite or by a ground station. (2.12)-(2.16) deals with both virtual link mapping and flow conservation. In particular, (2.12) ensures that the link outgoing the source node at service arrival period shall belong to the first virtual link if the service is not processed on the source node at the service generation period, to the second virtual link otherwise. Instead, on each intermediate node (i.e., any node different from both the source satellite at the service generation period and the vGS at the deadline period), (2.13) guarantees that if the node has an incoming link carrying the service data, it will also have an outgoing link. In particular, if the node processes the service, the incoming link shall appertain to the first virtual link, while the outgoing link shall appertain to the second virtual link. Instead, if no processing occurs on the node, incoming and outgoing links shall appertain to the same virtual link. Furthermore, it is important to take into consideration that nodes related to the service generation period cannot receive data from memory links; conversely, nodes related to the service deadline period cannot store data on memory links. This is obtained by taking advantage of the Kronecker delta notation such that:

$$\delta_{uv}=\begin{cases}1 & \text{if } u=v\\ 0 & \text{otherwise}\end{cases} \tag{2.17}$$

Furthermore, (2.14) and (2.15) prohibit an intermediate node to have, respectively, more than one incoming or more than one outgoing link related to a certain service. Instead, (2.16) ensures that there is one and only link incoming the ground station node and carrying the service data at the service deadline period and ensures that the information reaches the ground station at least at the deadline cycle. Furthermore,

by leveraging the fact that in EO applications the destination of the information is any of the enabled ground stations, not a specific one, it is only needed to write this constraint with respect to the vGS node, without worrying about the actual number of ground stations enabled to receive data. This also assures the scalability of this formulation with the number of ground stations, since increasing the number of ground stations, any extra complexity is added to the problem. It is also important to underline how the cyclostationary behaviour of both topology and especially service generation related to the EO application allows to write an optimization problem formulation related to a cyclostationary period only, since what happens in this period repeats during all the operative life. Instead, in case of aperiodical service generation, but still having the possibility to determine services *a priori*, the problem (either optimization or heuristic) should have been formulated for the full time span during which resources have to be allocated, at the expense of scalability. Indeed, in case of aperiodical service generation without the possibility to predetermine the service demand, the optimization problem formulation should have been completely different, since the problem should have considered computing and processing resource allocation one task at a time, on a network being partially loaded by services already allocated. In particular, the definition of the problem on a cyclostationary period only can be noticed by the fact that constraints (2.8)-(2.10) are defined only on time cycles within a cyclostationary period, while in constraints (2.11)-(2.16), since the service delivery deadline could be greater than the cyclostationary period duration, the modulo operation is leveraged to consider that all that would happen after the end of the cyclostationary period shall be considered at its beginning, because of the periodic repetitions.

Finally, it is important to notice that constraints and objective introduced in this section suppose to have a perfect knowledge of the unit processing, transmission and memory costs. However, at the best of my knowledge, precise values for these costs are not known at the moment, but may be in the future. For this reason, in the numerical evaluation of the proposed strategy I will consider these costs as multiple of the unit processing cost to be paid to process data on the ground station, which I suppose to be the easiest unit cost to be found. In particular, I will consider an α parameter representing the ratio between the unit processing cost on satellites over the unit processing cost on ground, and a ρ parameter identifying the ratio between the unit cost of handling information on a link (either for transmission or memory) and, again, the unit cost of data processing on ground station. Furthermore, the only impact of the data processing on this solution is given by the data reduction which can be obtained, that I will represent by means of the ε parameter identifying the ratio between unprocessed and processed service size.

2.4 Heuristics

In order to overcome the complexity of the proposed optimization problem, I introduce two heuristics aiming to jointly route and place processing of services in satellite constellations endowed with OEC capability. A summary of additional sets and variables used in the pseudocodes with respect to the reference scenario is given in Tab.2.2. The first heuristic is an Exhaustive Search, formalized in Alg.1.

Table 2.2. Heuristics sets and variables

Set or variable	Description
Λ	ordered set containing, for each f^h service, a tuple containing the chosen path and another tuple with the chosen processing node and time cycle
Ω	set of rejected services
$\Theta_{i,j}^t$	tensor variable tracking the memory and transmission resource allocation in each time period
Π_i^t	matrix variable tracking the processing resource allocation in each time period
κ^*	variable tracking the best cost for the current service
i^*, t^*	variables tracking the index of the best processing node and time cycle for the current service, respectively
λ^{0^*}, λ^{1^*}	ordered lists of nodes tracking the best path hosting the unprocessed and processed current service, respectively
$\mathcal{E}_e^0$, $\mathcal{E}_e^1$	sets of transmission links able to host unprocessed and processed current service, respectively
$\mathcal{E}_m^0$, $\mathcal{E}_m^1$	sets of memory links able to host unprocessed and processed current service, respectively
$\mathcal{T}^0$	set of layers (i.e., time cycles) between current service generation and delivery deadline time cycles
$\mathcal{T}^1$	set of layers (i.e., time cycles) between current service candidate processing time cycle and delivery deadline time cycle
$\mathcal{G}^0$	graph including only links able to host an unprocessed service and only time cycles between current service generation and deadline cycles
$\mathcal{G}^1$	graph including only links able to host an unprocessed service and only time cycles between current service generation and deadline cycles
Υ^0	set of shortest paths to any reachable nodes from the service source satellite
Υ^1	set containing one element, i.e., the shortest paths from the service source satellite to the vGS at service delivery deadline period
ζ^0	element of Υ^0, list of tuples node index/time cycle on the shortest path from the source satellite to the candidate processing node
ζ^1	element of Υ^1, list of tuples node index/time cycle on the shortest path from the candidate processing node to the vGS
$\tilde{i}$, $\tilde{t}$	candidate processing node index and candidate processing time cycle, respectively
i_s, t_s	current starting node index and time cycle, respectively
i_d, t_d	current destination node index and time cycle, respectively
$\tilde{\kappa}$	cost of the current path/processing node couple for the current service

The proposed heuristic provides for the following steps. For each service (Line 2):

(i) using the Dijkstra's Algorithm, calculate the minimum cost path to all nodes reachable from the source satellite at the service generation cycle within the service delivery deadline (Line 8), considering only links able to transmit or store an unprocessed service (Lines 4-7). Each reachable node is candidate to process the service;

(ii) for each candidate service processing node:

(ii.a) If the node has enough computational capacity to execute the task (Line 12):

(ii.a.1) Apply the Dijkstra's Algorithm to obtain the cheapest path from the selected node to the ground station at service deadline cycle (Line 17), considering only links able to transmit or store a processed service (Lines 13-16)

(ii.a.2) Calculate the total cost due to processing on the candidate node (Line 20), as well as transmission and storage both from the source satellite at service generation cycle to the candidate processing node and from the candidate processing node to the ground station at the service delivery deadline cycle (Lines 21-23)

(ii.a.3) If the cost of the candidate couple processing node/path is smaller than the cost related to the cheapest processing node/path couple so far, assign the current processing node/path couple as best solution (Lines 26-29)

(ii.b) If a couple processing node/path has been found (Line 34), append the cheapest path and processing node couple to the list storing the routing and processing decision for each service (Line 35) and update the matrices tracking the current processing allocation (Line 36) and the bandwidth/memory allocations (Lines 37-39); otherwise, append the service to the list of rejected services (Line 42).

It can be proven that the proposed algorithm shows polynomial complexity. In fact, for each of the N_T services (Line 2), I first apply the Dijkstra's algorithm on a graph having at most $N_S \cdot T$ nodes and $|\mathcal{E}|$ edges (Line 8), then I apply again the Dijkstra's algorithm on a graph having at most $N_S \cdot T$ nodes and $|\mathcal{E}|$ edges (Line 17) for each node reachable from the source satellite at service generation cycle within the service delivery to ground station deadline. In the worst case, the number of reachable nodes is given by all nodes of the graph, thus the second call to Dijsktra's algorithm is executed for at most $N_S \cdot T$ times. Recalling that the complexity of the Dijkstra's algorithm, in its fastest proposal, is $\mathcal{O}\left(E + V \log_2 V\right)$, where E and V are the number of edges and vertices of the graph, respectively, it is straightforward to prove that Exhaustive Search Algorithm complexity is $\mathcal{O}\left(N_T N_S T \left(|\mathcal{E}| + N_S T \log_2 \left(N_S T\right)\right)^2\right)$, where $|\mathcal{E}|$ represents the cardinality of the edge set.

Algorithm 1: Exhaustive Search

Input: Σ, $\mathcal{N}$, $\mathcal{E}$, $\mathcal{T}$

Initialize: $\Theta^t_{i,j} \leftarrow 0$, $\Pi^t_i \leftarrow 0$, $\forall i,j \in [0,\dots,N_S-1]$, $t \in [0,\dots,T-1]$;
for $f^h \in \Sigma$ **do**
  Initialize: $\kappa^* \leftarrow \infty$, $i^* \leftarrow -1$, $t^* \leftarrow -1$, $\lambda^{0^*} \leftarrow \emptyset$, $\lambda^{1^*} \leftarrow \emptyset$;
  $\mathcal{T}^0 \leftarrow \{l_t \in \mathcal{T} \mid t \in [f^h_t,\dots,(f^h_t+f^h_d)]\ mod\ T\}$;
  $\mathcal{E}^0_e \leftarrow \{e^t_{i,j} \in \mathcal{E}_e \mid \tau C^t_{i,j} \geq \Theta^t_{i,j} + f^h_0,\ \forall i,j \in [0,\dots,N_S-1],\ i \neq j,\ t \in [f^h_t,\dots,(f^h_t+f^h_d)]\ mod\ T\}$;
  $\mathcal{E}^0_m \leftarrow \{m^t_i \in \mathcal{E}_m \mid M^t_i \geq \Theta^t_{i,i} + f^h_0,\ \forall i \in [0,\dots,N_S-1],\ t \in [f^h_t,\dots,(f^h_t+f^h_d)]\ mod\ T\}$;
  $\mathcal{G}^0 \leftarrow (\mathcal{N}, \mathcal{E}^0_e \cup \mathcal{E}^0_m, \mathcal{T}^0)$;
  $\Upsilon^0 \leftarrow Dijkstra\left(\mathcal{G}^0,\ \left\{f^h_s; f^h_t\right\},\ all\ nodes\right)$;
  if $\left|\Upsilon^0\right| \neq 0$ **then**
    for $\zeta^0 \in \Upsilon^0$ **do**
      $\tilde{i}, \tilde{t} \leftarrow \zeta^0_{last}$;
      if $\Gamma_{\tilde{i}} \cdot \tau \geq \Pi^{\tilde{t}}_{\tilde{i}} + f^h_0$ **then**
        $\mathcal{T}^1 \leftarrow \{l_t \in \mathcal{T} \mid t \in [\tilde{t},\dots,\tilde{t}+f^h_d]\ mod\ T\}$;
        $\mathcal{E}^1_e \leftarrow \{e^t_{i,j} \in \mathcal{E}_e \mid \tau C^t_{i,j} \geq \Theta^t_{i,j} + f^h_1,\ \forall i,j \in [0,\dots,N_S-1],\ i \neq j,\ t \in [\tilde{t},\dots,\tilde{t}+f^h_d]\ mod\ T\}$;
        $\mathcal{E}^1_m \leftarrow \{m^t_i \in \mathcal{E}_m \mid M^t_i \geq \Theta^t_{i,i} + f^h_1,\ \forall i \in [0,\dots,N_S-1],\ t \in [\tilde{t},\dots,\tilde{t}+f^h_d]\ mod\ T\}$;
        $\mathcal{G}^1 \leftarrow (\mathcal{N}, \mathcal{E}^1_e \cup \mathcal{E}^1_m, \mathcal{T}^1)$;
        $\Upsilon^1 \leftarrow Dijkstra\left(\mathcal{G}^1,\ \left\{\tilde{i};\tilde{t}\right\},\ \left\{N_S-1; \left(f^h_t+f^h_d\right)\ mod\ T\right\}\right)$;
        if $|\Upsilon^1| \neq 0$ **then**
          $\zeta^1 \leftarrow \Upsilon^1_0$;
          $\tilde{\kappa} \leftarrow \frac{f^h_0}{\tau} \cdot \gamma^p_{\tilde{i}}$;
          for $p \in \{0,1\}$, $a \in [0,\dots,\max\{0,|\zeta^p|-2\}]$ **do**
            $i_s, t_s \leftarrow \zeta^p_a$; $i_d, t_d \leftarrow \zeta^p_{\min\{a+1,|\zeta^p|-1\}}$;
            $\tilde{\kappa} \leftarrow \tilde{\kappa} + (1-\delta_{i_s,i_d}\delta_{t_s,t_d}) \cdot f^h_p \cdot [(1-\delta_{i_s,i_d}) \cdot \gamma_{e_{i_s,i_d}} + \delta_{i_s,i_d} \cdot \gamma_{m_{i_s}}]$;
          end
        end
        if $\tilde{\kappa} \leq \kappa^*$ **then**
          $i^* \leftarrow \tilde{i}$, $t^* \leftarrow \tilde{t}$;
          $\lambda^{0^*} \leftarrow \zeta^0$; $\lambda^{1^*} \leftarrow \zeta^1$;
          $\kappa^* \leftarrow \tilde{\kappa}$;
        end
      end
    end
  end
  if $i^* \neq -1$ **then**
    $\Lambda \leftarrow \Lambda \cup \{\{\lambda^{0^*} \cup \lambda^{1^*}, \{i^*;t^*\}\}\}$;
    $\Pi^{t^*}_{i^*} \leftarrow \Pi^{t^*}_{i^*} + f^h_0$;
    for $p \in \{0,1\}$, $a \in \left[0,\dots,\max\{0,\left|\lambda^{p^*}\right|-2\}\right]$ **do**
      $i_s, t_s \leftarrow \lambda^{0^*}_a$; $i_d, t_d \leftarrow \lambda^{0^*}_{\min\{a+1,|\lambda^{p^*}|-1\}}$;
      $\Theta^{t_s}_{i_s,i_d} \leftarrow \Theta^{t_s}_{i_s,i_d} + (1-\delta_{i_s,i_d}\delta_{t_s,t_d})\, f^h_p$;
    end
  else
    $\Omega \leftarrow \Omega \cup \{f^h\}$;
  end
end

Output: Λ, Ω

Algorithm 2: Shortest Path-based Heuristic

Input: Σ, $\mathcal{N}$, $\mathcal{E}$, $\mathcal{T}$
1 Line (1)-(7) of Alg.1;
8 $\Upsilon^0 \leftarrow Dijkstra\left(\mathcal{G}^0,\ \left\{f_s^h; f_t^h\right\},\ \left\{N_S - 1; \left(f_t^h + f_d^h\right) mod\ T\right\}\right)$;
9 **if** $\left|\Upsilon^0\right| \neq 0$ **then**
10 $\quad \zeta^0 \leftarrow \Upsilon_0^0$;
11 $\quad$ **for** $\{\tilde{i}, \tilde{t}\} \in \zeta^0$ **do**
12 $\quad\quad$ Line (12)-(31) of Alg.1;
32 $\quad$ **end**
33 **end**
34 Line (34)-(44) of Alg.1;
Output: Λ, Ω

In order to further reduce complexity, I also propose another algorithm (Shortest Path-based Algorithm, Alg.2) which modifies the Exhaustive Search Algorithm only in the way the candidate processing nodes are selected (Line 8). In fact, in the Shortest Path-Based Algorithm, the first call to Dijkstra's algorithm does not return the paths to all reachable nodes from the source satellite at service generation cycle within the service delivery to ground station deadline cycle, but only the shortest path from the source satellite at service generation cycle to the ground station at the deadline cycle.

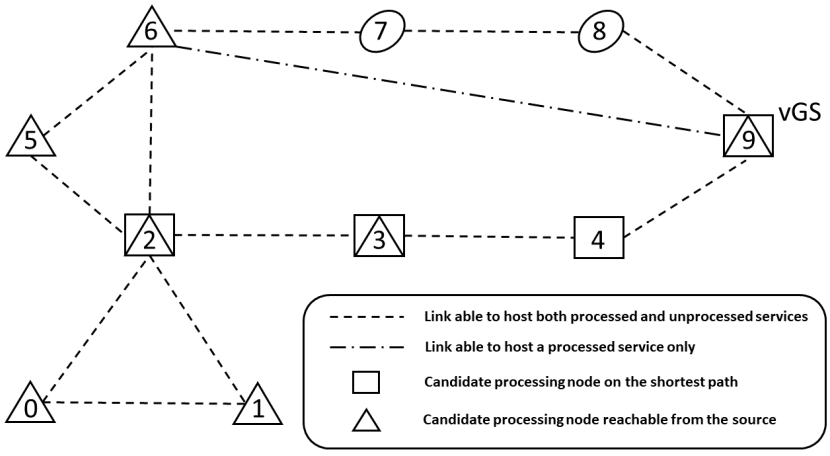

Figure 2.4. Example of the difference in candidate processing node identification between Exhaustive Search and Shortest Path-based Heuristic.

An example of the difference between the two algorithms is given in Fig.2.4, where it can been noticed that, in case a service generated by node 2 shall be delivered to the vGS, the Shortest Path-based heuristic would choose only the nodes on the shortest path between 2 and vGS on a graph containing only links able to host an unprocessed service (i.e., nodes 2, 3, 4, 9) as candidate processing nodes. Instead, the Exhaustive Search selects as candidate processing nodes all the ones reachable from the source node (i.e., in the example, nodes 0, 1, 2, 3, 5, 6, 9), thus leading to a higher number of processing possibilities to be evaluated. However, this higher complexity is counterbalanced by an increased ability of the Exhaustive Search in

returning the cheapest solution with respect to the Shortest Path-based Heuristic. In fact, always referring to the example of Fig.2.4, let me assume that the source (node 2) is not able to process the service, while any candidate processing node has enough processing capacity to accomplish the task. Let me also assume that processing has the same cost on any node, and it is equal to the transmission one. In this case, the Shortest Path-based Heuristic would place the processing on node 3, with an overall path made of three links (between nodes 2-3, 3-4, 4-9). Instead, the Exhaustive Search would place the processing on node 6, with an onverall path reduced to two links (between nodes 2-6 and 6-9), thus, it returns a cheaper solution by taking advantage of a link only able to host a processed service. Moving to a formal discussion of the complexity, also in this case the first Dijkstra's algorithm call is executed on a graph providing only for transmission and memory links able to host an unprocessed service. Then, only nodes appertaining to the found path will be considered as candidate processing node, thus, the second Dijkstra's algorithm call will be executed fewer times if compared to the Exhaustive Search algorithm. By introducing μ as the highest number of nodes crossed by a service in its path from the source satellite at service generation cycle to the ground station at the service delivery deadline cycle, it is then straightforward to prove that the complexity for the Shortest Path-Based Algorithm reduces to $\mathcal{O}\left(N_T \mu \left(|\mathcal{E}| + N_S T \log_2 \left(N_S T\right)\right)^2\right)$.

It is important to underline that, as in the case of the optimization problem, also heuristics take advantage of the cyclostationary behaviour typical of EO applications in both nodes motion (i.e., topology) and service generation. This can be noticed in the selection of layers, nodes and edges, where I leverage modulo operations to consider that if the service delivery deadline exceeds the cyclostationary period, all events occurring after the end of the cyclostationary period shall be considered at its beginning because of periodic repetitions. Furthermore, the fact that in EO applications the information destination is any of the enabled ground stations and not a specific one allows to just calculate the shortest path to the vGS node at service delivery deadline cycle as the destination, without worrying about the different available ground stations. This makes the heuristics perfectly scalable with the number of ground stations, since no further complexity is added by increasing the number of ground stations.

Finally, it can be noticed that the proposed heuristics try to mimic the optimal solution, but in a simplified way with respect to the optimization problem, since they consider one service at a time. This is made possible by the modelling presented in Section 2.2, representing a dynamic environment into a graph, thus, allowing for obtaining solutions by leveraging the Dijkstra's algorithm. In case these heuristics show good results with respect to the optimization problem, it is then clear that they can be applied to evaluate the performance of the proposed allocation scheme (providing for the possibility of task offloading) with respect to state-of-the-art benchmarks where limited processing possibilities are considered in complex scenarios, like a real orbital application, where the optimization problem solution would be impractical.

2.5 Numerical results

In this section, I will first validate the proposed heuristics by comparison with the results obtained by solving the optimization problem. Because of its high complexity, the solution of the optimization problem requires a high computational effort in a real orbital scenario. Thus, I solve it on a mock-up case study identified to mimic the orbital scenario but requiring a lower computational effort to obtain the optimization solution.

After studying the performance of the proposed heuristics, I apply them to a real orbital scenario, in order to compare them with two benchmark heuristics providing for, respectively, service processing on the source satellite only or on the ground station only. Parametric analysis on the processing cost, transmission/memory cost, ratio between preprocessing and postprocessing data size and evaluation on the delivery delay to the ground station will be proposed.

Let me introduce parameters which will be the same for both the heuristics validation and their application to the orbital scenario. The time cycle duration is given by $\tau = 10$ min, which is a compromise value between the granularity of the dynamic topology representation and the complexity of obtaining a solution. I will consider unlimited processing capacity for ground stations, since I assume to have datacenters with enough capacity to handle all the data amount produced by the constellation, with a cost $\gamma^p_{GS} = \gamma^p_{N_S-1} = 1$ \$/Mb, while satellites will have:

$$\Gamma_i = \max_{h\in[0,\dots,N_T-1]} f^h_0/\tau,\ \forall i \in [0,\dots,N_S-2] \tag{2.18}$$

processing capacity (i.e., each satellite can at most process a data amount equal to the heaviest service in a time cycle), with a cost depending on a parameter α such that:

$$\gamma^p_i = \alpha \cdot \gamma^p_{GS},\ \forall i \in [0,\dots,N_S-2] \tag{2.19}$$

Thus, I assume that all satellites have the same unit processing cost, and for the sake of clarity I name this cost γ^p_{Sat}, where $\gamma^p_{Sat} = \gamma^p_i = \alpha \cdot \gamma^p_{GS},\ \forall i \in [0,\dots,N_S-2]$ and α represents the ratio between the unit processing cost on any satellite over the unit processing cost on ground. Notice that, since it is difficult to obtain a real γ^p_{GS} cost, a unit value has been chosen to ease both the readability of the presented results and their interpretation whenever a real γ^p_{GS} cost is available. Following [97], for the transmission links I will consider a transmission power $P_t = 5$ W, antenna gain $G = 27$ dBi, carrier frequency $\nu = 26.375$ GHz, system noise temperature $T_s = 700$ K, minimum required Eb/N0 ratio $Eb/N0|_{min} = 6$ dB, a mean data rate $R_{ISL} = 1$ Mbps in case of ISLs (to study the behaviour of the algorithm in facing scarce resource availability), $R_{dl} = 520$ Mbps in case of downlink to the ground station (as in the case of Sentinel-2 satellites[98]), and a cost depending on a parameter ρ such that:

$$\gamma_{e_{i,j}} = \rho \cdot \gamma^p_{GS},\ \forall i,j \in [0,\dots,N_S-1],\ i \neq j \tag{2.20}$$

Finally, memory cost depends again on a parameter ρ such that:

$$\gamma_{m_i} = \rho \cdot \gamma_{GS}^{p},\ \forall i \in [0, \dots, N_S - 2] \tag{2.21}$$

while I will consider ground stations having unlimited storage capacity with no associated cost, since I assume again to have datacenters with enough capacity to handle all the data amount produced by the constellation.

In my evaluation, I will present results related to the total cost (given by the sum of processing, transmission and memory costs) and to its components processing cost and link cost, where link cost is given by the sum of transmission and memory costs. This is related to the fact that both optimization and heuristics have to balance routing (thus, selection of transmission and memory links) and processing placement in order to minimize the total cost. Thus, it is more interesting to just compare these two components, without further divide the link cost into transmission and memory costs.

Finally, I set preprocessing size $f_0^h = 100$ Mb $\forall h \in [0, \dots, N_T - 1]$, and postprocessing size depending on a parameter ε such that:

$$f_1^h = f_0^h/\varepsilon,\ \forall h \in [0, \dots, N_T - 1] \tag{2.22}$$

Before moving to the core of the numerical result presentation, it is important to underline that, once the topology and services are fixed (i.e., given the orbit, the number of satellites and their motion within the constellation and with respect to the Earth), the performance of the proposed strategies will be only related to three parameters: α, ρ and ε. In fact, the proposed strategies aim to minimize the total operating cost, and these parameters are all linked to unit costs to be paid while handling data. In particular, the α parameter represents the ratio between the unit processing cost on satellites over the unit processing cost on ground. Thus, I expect that, the higher the α value is, the more a robust strategy should choose to leverage transmission and memory links to reach the ground station, where the processing should happen because of lower costs with respect to satellites. However, this behaviour also depends on the value of ρ parameter, identifying the ratio between the unit cost of handling information on a link (either for transmission or storage) and, again, the unit cost of data processing on ground station. In fact, even though processing on-board of satellites costs more than on ground, placing processing on a ground station is convenient only if the cost to be paid for making an unprocessed service reach the ground station is small enough to lead to a lower total cost with respect to the case the processing happens in orbit. Instead, ε represents the data reduction ratio due to processing (i.e., the ratio between unprocessed and processed service size). In particular, I fix the preprocessing size, in such a way that the choice of ε parameter changes the postprocessing size. I expect that robust strategies are influenced by this parameter since by processing data in orbit, the transmission of a reduced amount of information should lead to an increase in transmission resource availability and to an overall lower total operating costs. Finally, it is important to underline how I propose a parametrical analysis of the proposed strategies because of the difficulty in obtaining unit cost for data handling (either processing or transmission/storage), with the unit processing cost on ground stations used as reference in both α and ρ parameter definitions since I suppose it to be the easiest cost to be modeled. Similarly, I do not limit my analysis to a specific

data processing application, but I study the contribution of data elaboration by only considering the impact it has on the network by means of the ε parameter (i.e., the reduction in data amount to be transmitted to the ground station).

2.5.1 Comparison between Optimization Problem and Heuristics

Topology and Services

In order to compare the results obtained by solving the proposed optimization problem with the two presented heuristics, I will consider a synthetic topology to mimic the orbital mechanics in a simpler, but not trivial, scenario. In particular, I will consider $N_S = 16$ nodes (15 of them standing for satellites, the last one for the vGS) and $T = 5$ time cycles. Satellite nodes are arranged so as to follow the geometrical structure: in particular, they are placed on the vertexes of contiguous squares. In order to simulate the fact that only near satellites are able to communicate, I define as possible links only the ones connecting nodes on the sides or the diagonal of the squares. Then, only a certain percentage of all links will be randomly sampled to be considered available during each time cycle. In order to have a not trivial number of services in such a small topology (in particular, because of the small number of considered time cycle to obtain the optimization problem results), I experimentally choose this percentage to be equal to 70%. Instead, the vGS node is connected to a different trio of nodes during each time cycle. Finally, let me introduce the parameters related to the topology. In this scenario, I will consider $\rho = 0.01$ and $\alpha \in [1, \dots, 20]$. I will also set each satellite node to have a memory $M_{m_i} = 100$ Mb $\forall i \in [0, \dots, N_S - 1]$, in order to better study what happens when a node has to deal with two services during the same time cycle.

As far as service generation is concerned, by noting that in orbital scenarios not all satellites generate services simultaneously, I implement this feature in my synthetic model by setting that only a percentage of randomly selected nodes can generate services during each time cycle. Again, in order to obtain a non trivial number of services in such a simplified scenario, I experimentally choose this percentage to be equal to 30%. Finally, 10 different service sets are generated, each containing 30 random services. Randomness in service generation is related to the source satellite and the service generation cycle. Instead, I will consider $\varepsilon = 1$ and a deadline equal to $T - 1 = 4$ time cycles for all services.

Results and discussion

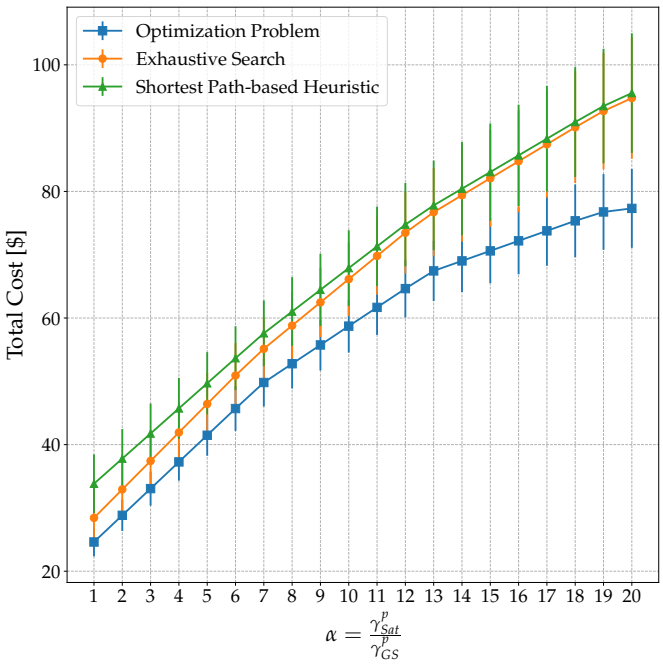

Figure 2.5. Total cost obtained by solving Optimization Problem or by applying Exhaustive Search and Shortest Path-Based Heuristic.

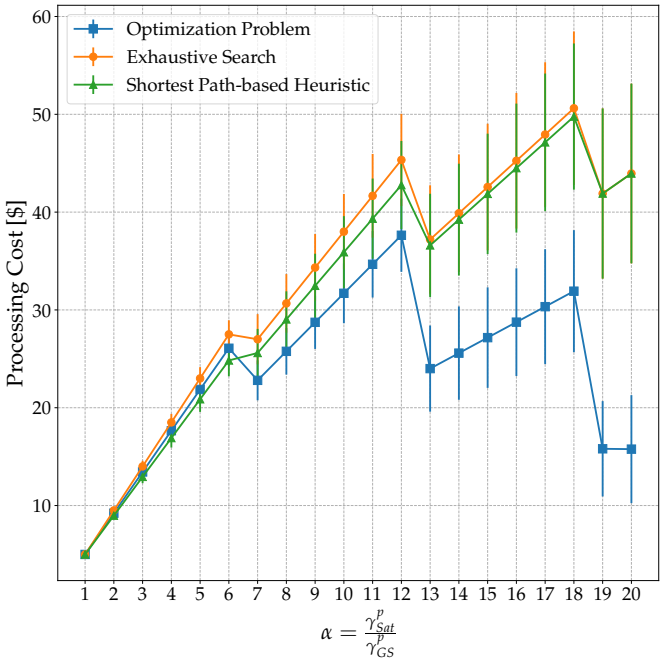

Figure 2.6. Processing cost obtained by solving Optimization Problem or by applying Exhaustive Search and Shortest Path-Based Heuristic.

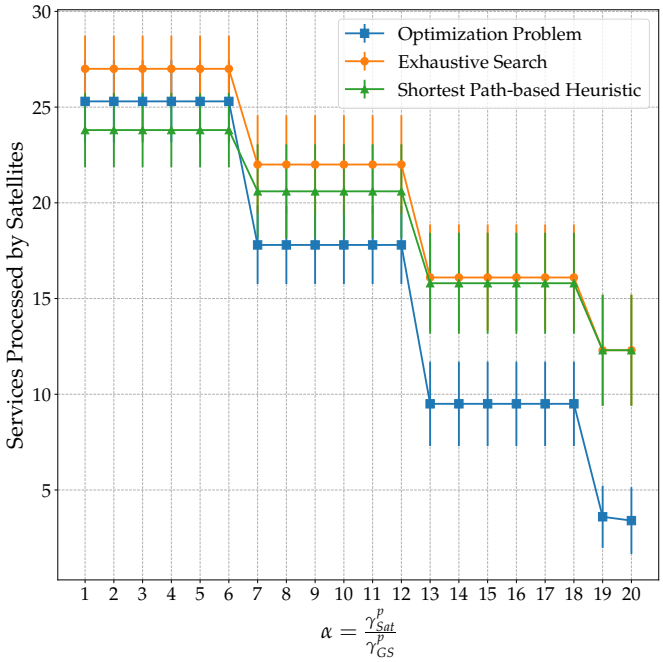

Figure 2.7. Number of services processed by satellites obtained by solving Optimization Problem or by applying Exhaustive Search and Shortest Path-Based Heuristic.

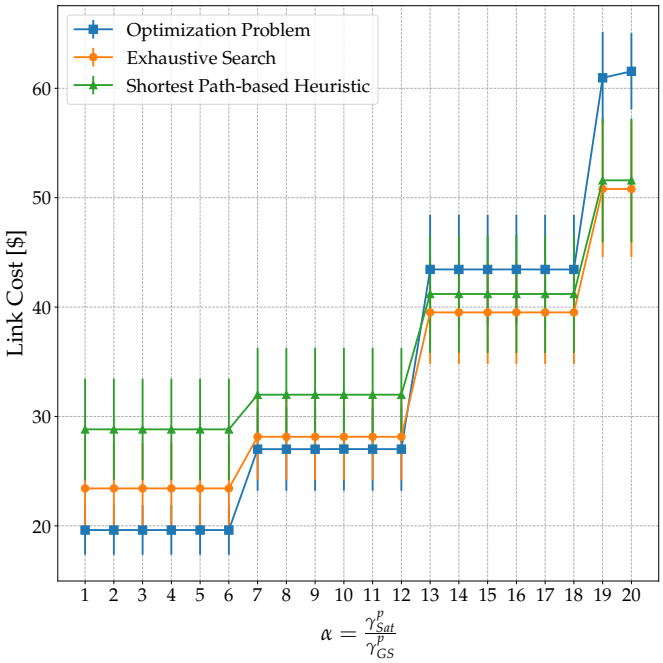

Figure 2.8. Link cost obtained by solving Optimization Problem or by applying Exhaustive Search and Shortest Path-Based Heuristic.

Fig.2.5 shows mean and standard deviation for the total costs, obtained by solving optimization problem or by applying Exhaustive Search or Shortest Path-Based Heuristic to the 10 random service sets, reporting the total cost variations for increasing values of the ratio α between the unit processing cost on satellites and on ground station. It can be noticed that heuristics lead to a higher total cost than optimization problem for any value α assumes. However, the three curves have a similar behaviour for small values of α (i.e., when the unit processing cost on satellites is almost equivalent to the unit processing cost on ground station). Instead, as the unit processing cost on satellites increases with respect to the one on ground station, there is a higher discrepancy (up to 24%) between the total cost obtained by applying heuristics with respect to the optimization problem. This is due to the fact that heuristics prefer processing on satellites to minimize the total cost associated to the single service, without having an holistic vision on all services as in the optimization problem case. This is also confirmed by looking at the processing cost in Fig.2.6, where it can be noticed how heuristics have higher processing cost than the optimization problem. Furthermore, for all the curves, the processing cost follows a piecewise linear behaviour. During each linear piece, the number of services processed by satellites remains the same, as it can be noticed in Fig.2.7, reporting the number of services processed by satellites for each α value, by applying the different proposed strategies. Consequently, the increasing linear behaviour of the processing cost in regions of α values where the number of services processed by satellites is constant is only due to the increase of α (i.e., to the increase of processing cost on satellites). Instead, a decrease in the number of services processed by satellites is linked to a break of the increasing linear trend. This behaviour can be also confirmed by looking at the transmission cost values in Fig.2.8, where it can be noticed that, in the regions where the processing cost increases (i.e., the number of services by satellites is constant), the link cost remains constant (i.e., the amount of data transmitted on links remains the same), while when the increasing linear trend breaks (i.e., a lower number of services are processed by satellites), the link cost increases, since there is more unprocessed data to be transmitted on links. It is also interesting to underline how, for small α values, heuristics tend to use links more than optimization problem, while, as α increases, optimization problem tends to pay more for the link use than the heuristics. This is due to the fact that, thanks to its holistic vision on the complete service set, the more α grows, the more the optimization tends to place processing on ground with respect to the heuristics, thus, the more unprocessed data shall be transmitted on the links. However, this higher cost related to transmission/memory links is counterbalanced by the much lower cost to be paid for processing, leading to a total cost smaller for optimization problem.

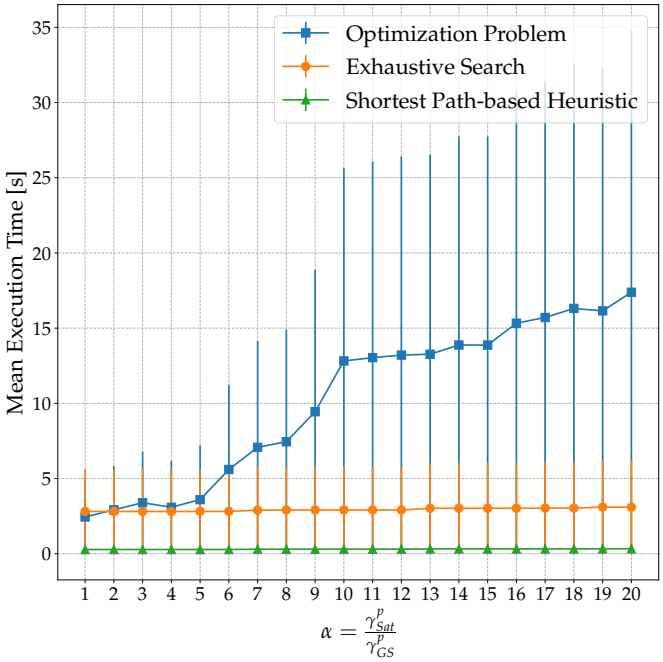

Figure 2.9. Mean Execution Time in the simplified, yet not trivial, comparison scenario.

Finally, Fig.2.9 shows the mean and standard deviation of the execution time of Optimization Problem, Exhaustive Search and Shortest Path-based Heuristic in the considered comparison scenario. It can be noticed that the optimization problem solution takes a sensibly increased time with respect to the heuristics, and both mean and standard deviation increase when α increases in case of optimization problem solution, while they remain constant in case heuristics are applied. This is due to the fact that α influences the simplicity in finding the optimal solution, while when using heuristics, the computational time is only related to the number of executions of Dijkstra's algorithm and on the complexity of the graph, both being uncorrelated to α. In particular, it can be noticed how the Shortest Path-based Heuristic leads to the lowest mean execution time thanks to its reduced number of Dijkstra's algorithm applications. Finally, it shall be underlined that even though the reported mean execution time values are quite low, this is due to the fact that I am considering a simplified, yet not trivial, scenario. However, execution time is expected to increase with the number of satellites, time cycles and services, with a much stronger increase in case of optimization problem solution due to its complexity.

2.5.2 Application of Heuristics to Orbital Scenario

Topology and services

In order to evaluate the proposed joint routing and processing placement scheme on a real scenario, I developed a Python tool to generate Walker constellations and propagate the satellite and ground station positions over time. In particular, I considered a constellation of 24 satellites divided into 6 orbital planes (each containing 4 satellites). Since I am studying applications for EO, I choose circular

orbits with typical height (712.84 km) and inclination (98.24 deg), which allows to obtain a repeat cycle (i.e., the cyclostationary period) equal to 2 sidereal days. In the orbital simulation, I do not take into account effects due to the Earth's J2. As far as ground segment is concerned, I will consider three ground stations of the European network, placed in Matera (Italy), Kiruna (Sweden) and Kourou (French Guyana). All this translates into a time-evolving graph having $N_S = 25$ nodes (24 satellite and 1 vGS), and $T = 287$ time cycles (as a consequence of the 2 sidereal days repeat cycle divided into 10 minutes slots, as previously discussed). In particular, the satellite positions over time obtained in the Walker constellation generation are used to set the intra-layer edges of the time-evolving graph. Specifically, for each time cycle, if the distance between two satellites i and j remains small enough to guarantee the communication with the desired $Eb/N0|_{min}$ during a time interval $\hat{\tau}^t_{i,j}$ within the t-th time cycle, then an intra-layer edge in the time-evolving graph is added between the nodes representing the two satellites in the considered time cycle. This edge represents an ISL with a capacity given by $C^t_{i,j} = R_{ISL} \cdot \hat{\tau}^t_{i,j}/\tau$. By repeating this check over each couple of satellites during each time cycle, all ISLs are translated into the time-evolving graph which is fed to the resource allocation strategies. In this way, ISLs are taken into consideration while determining the best route the information should follow to reach the processing node and, finally, the ground station. A similar approach is followed while determining the intra-layer edges between the vGS node with any other node. Starting again from the positions propagated over time in Walker constellation generation, if the relative position between a satellite i and any ground station is such that the elevation angle is smaller than 5 deg during a time interval $\hat{\tau}^t_{i,j}$ within the t-th time cycle, then an intra-layer edge between the node representing the satellite during the t-th time cycle and the vGS in the same time cycle is added, with a capacity $C^t_{i,N_S-1} = R_{dl} \cdot \hat{\tau}^t_{i,j}/\tau$. I also consider $\alpha \in [1, \ldots, 34]$, while ρ will be defined for each of the proposed analysis. Finally, memory on each satellites is limited to $M_i = 3.3$ Gb, $\forall i \in [0, \ldots, N_S - 2]$ to support the worst case load of data in the service generation scenario described further on.

As far as service generation is concerned, I developed a Python tool to generate a service as soon as a satellite starts flying over user-defined regions. Furthermore, if after τ_s seconds after the previous passage over the region the satellite is still flying over the selected area, another service is generated. ε parameter value will be defined for each of the following analysis, while the deadline is chosen as the first time cycle after the service generation during which there is visibility between any ground station and the service source satellite (i.e., the earliest time cycle during which the satellite can potentially downlink all the acquired data, representing the best solution when no satellite network is available). In this chapter, I will consider as selected regions Australia and Mexico, with $\tau_s = 5$ min in order to have at most two services generated by a satellite during a time cycle as a compromise between the analysis of the heuristic behaviour and the solution complexity. This leads to the generation of 673 services.

Results

I first want to evaluate the behaviour of the proposed heuristics in comparison with two benchmark algorithms providing for, respectively, processing services on the source satellite only (during the service generation time cycle or in following one, named Always First) and processing services on the ground station only (named Always Ground). By choosing parameters $\rho = 0.01$ and $\varepsilon = 100$, I obtain the total cost due to the application of the different algorithms, as function of the α ratio between unit processing cost on satellites and on ground stations, shown in Fig.2.10.

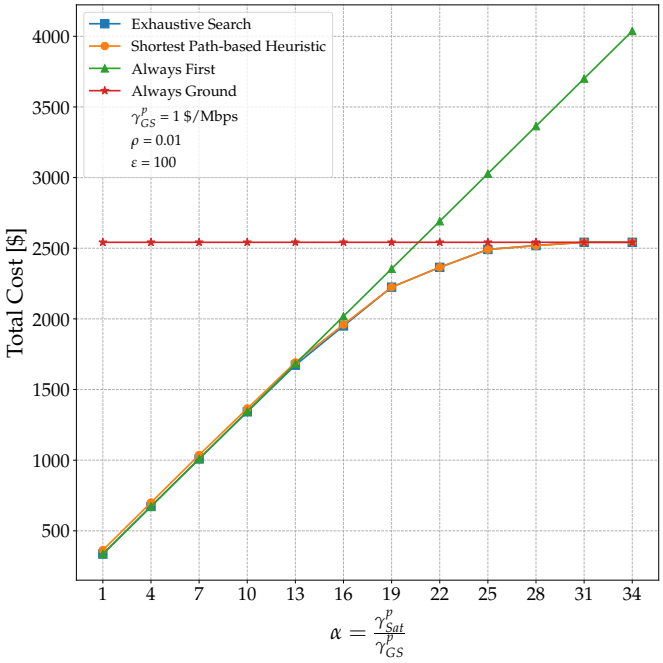

Figure 2.10. Total cost obtained by applying the two proposed heuristics and the two benchmarks, fixing ρ and ε parameters.

It can be noticed how the proposed heuristics well follow the Always First and Always Ground benchmark, respectively, for low α values and for high α values. This is due to the fact that, for low α values, the unit processing cost on satellites is not much higher than the unit processing one on the ground station, and the slightly higher processing cost is compensated by a lower link cost due to the transmission/storage of a reduced data amount. As α increases, this convenience is gradually lost, and the lower link cost is not counterbalanced by the higher cost to be paid for on-board processing, thus, the algorithm chooses to process more data on ground station. In this way, there is an intermediate region of α values where the proposed heuristics outperform the benchmark solutions, since the former have the ability to choose where placing the service processing in order to obtain the cheapest balance between processing and routing costs. It is finally evident that for high α values, there is no convenience in processing on-board any service and, thus, the proposed heuristics follow the same behaviour as the Always Ground solution.

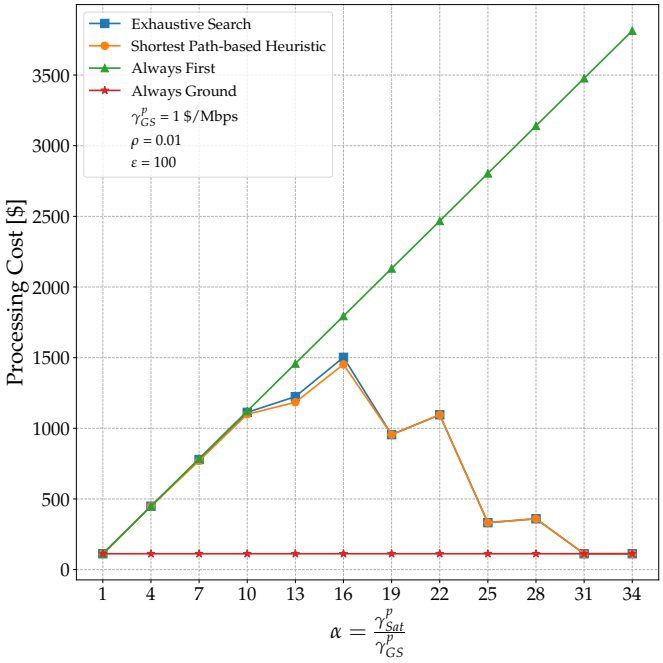

Figure 2.11. Processing cost obtained by applying the two proposed heuristics and the two benchmarks, fixing ρ and ε parameters.

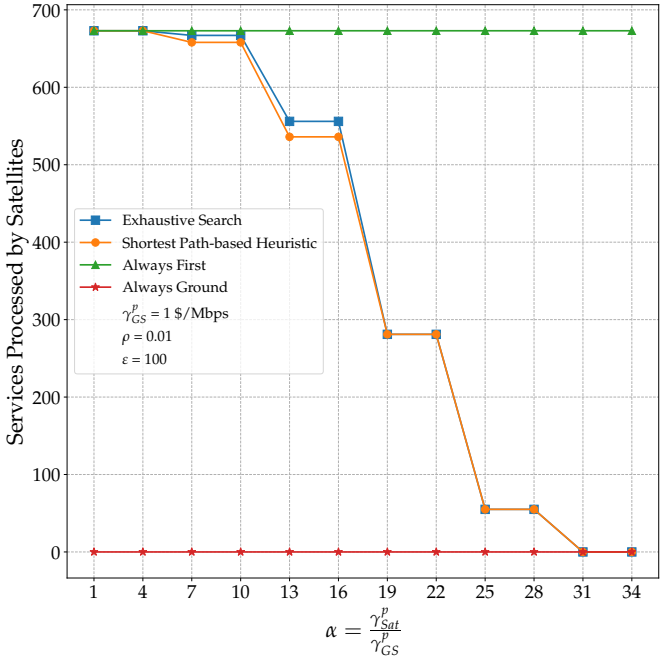

Figure 2.12. Number of services processed by satellites, by applying the two proposed heuristics and the two benchmarks, fixing ρ and ε parameters.

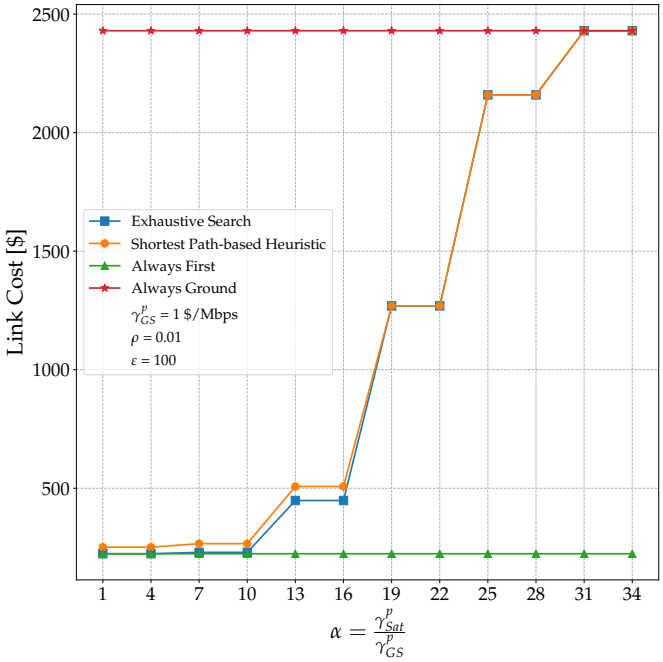

Figure 2.13. Link cost obtained by applying the two proposed heuristics and the two benchmarks, fixing ρ and ε parameters.

What has been just discussed is also confirmed by looking at the processing cost, the number of services processed by satellites and the link cost, reported, respectively, in Figs.2.11, 2.12 and 2.13. The three curves follow the same behaviour commented with respect to the heuristics and optimization problem comparison. On one hand, processing cost appears to have a piecewise linear behaviour, where changes in the slope and sudden decreases are due to a change in the number of services processed within the constellation. In particular, for small α values the processing cost follows the same behaviour as the Always First solution, while for high α values the same costs as the Always Ground benchmark are obtained. On the other hand, the link cost increases when an increase of the service to be processed on the ground stations occurs (because of the increase of the data amount to be transmitted or stored on links), while remains constant when the number of services processed on-board is constant. These behaviours allow to discuss in deeper detail the reason why the number of services processed by satellites decreases as α increases. In fact, in case the number of services processed by satellites remains the same when α increases, as it happens in case the Always First strategy is applied, there would be a linear increase of the processing cost with α, while the link cost would remain constant because the amount of data to be transferred within the network would not change. Thus, the total cost would increase linearly with α. However, it may be possible to obtain a lower total cost by deciding to process data on-ground instead of on-board as α increases, because when α increases, the cost to be paid for unit processing on satellites with respect to ground stations increases, too. For this reason, the proposed heuristics tend to place processing on ground when α increases. In fact, in this case, even though the link cost increases due to the higher amount of unprocessed data to

be transferred, the obtained saving in processing cost is such that the overall total cost is smaller than in case more data is processed on-board.

A further comparison on the link cost paid by Exhaustive Search and Shortest Path-based Heuristic allows to notice that the two solutions have the same trend but different values for small α values, with Shortest Path-based heuristic paying a slightly higher cost than Exhaustive Search. Since in the same region the two heuristics and the Always First benchmark pay the same amount for processing, this leads to a slightly higher total cost to be paid by the Shortest Path-based Heuristic than both Exhaustive Search and Always First solution. This effect is linked to the way in which services are handled by the source satellite during their generation cycle when the source satellite is busy in processing another service (recall that satellite processing capacity is set enough to process only one service during each time cycle). In this case, the processing shall be executed by another satellite or by the source satellite itself, but in a following time cycle. By applying the Shortest Path-based Heuristic, the candidate nodes for processing are limited to only the ones being on the shortest path between the source satellite and the ground station, evaluated by taking into account only links able to host an unprocessed service. Thus, only a subset of candidate processing nodes is considered, with no assurance on the fact that it includes the best one. In fact, the cheapest strategy could be, for example, reaching a node not being on the shortest path and, after processing, taking advantage of an higher number of links able to now host the processed service, due to its reduced data burden, to reach the ground station with a lower link cost. It is evident that the Shortest Path-based Heuristic is not able to identify these strategies, unless the best processing node is on the shortest path to the ground station. Furthermore, the application of this heuristic could also lead to a resource shortening on nodes and links on the shortest path if they are shared by an high number of services, and this can further lead to higher total costs. Instead, since Exhaustive Search evaluates all reachable nodes from the source satellite to the ground station within the service delivery deadline, it is able to identify the best strategy for each single service. This difference between the two heuristics reduces when α increases, since the increased unit cost required for processing on satellites increases the number of services to be processed on the ground station to pay a lower total cost. Finally, the Shortest Path-based Heuristic leads to slightly higher costs than Always First for low α values because in the latter case, since the processing can only happen on the source node (during the service generation cycle or in a next cycle), the shortest path from the source satellite at processing cycle to the vGS at service delivery deadline is obtained on a graph including all links able to host a processed service, thus, potentially containing an higher number of available links than the case of Shortest Path-based Heuristic.

I also propose a parametric analysis on the value of the ρ parameter (i.e., by varying the link cost to be paid for unit data transmission or storage). For this evaluation, I only consider costs obtained by applying Shortest Path-based Heuristic and I will fix $\varepsilon = 10$. Total cost is reported in Fig.2.14, processing cost in Fig.2.15, number of services processed by satellites in Fig.2.16 and link cost in Fig.2.17. From these plots it is possible to conclude that an increase in the ρ parameter (i.e., an higher transmission and storage unit cost) corresponds to an increase in total costs. However, the higher the ρ, the higher is the interval of α values (i.e., of on-board

unit processing cost) for which there is a convenience in processing services on-board. This can be easily noticeable by looking, for example, at the value of α after which the total cost is constant (i.e., it does not increase any more because all services are processed on the ground stations, thus there is no dependence on the unit cost for on-board processing). Similarly, it can be noticed that, for any α value, the number of services processed by satellites increases when ρ increases.

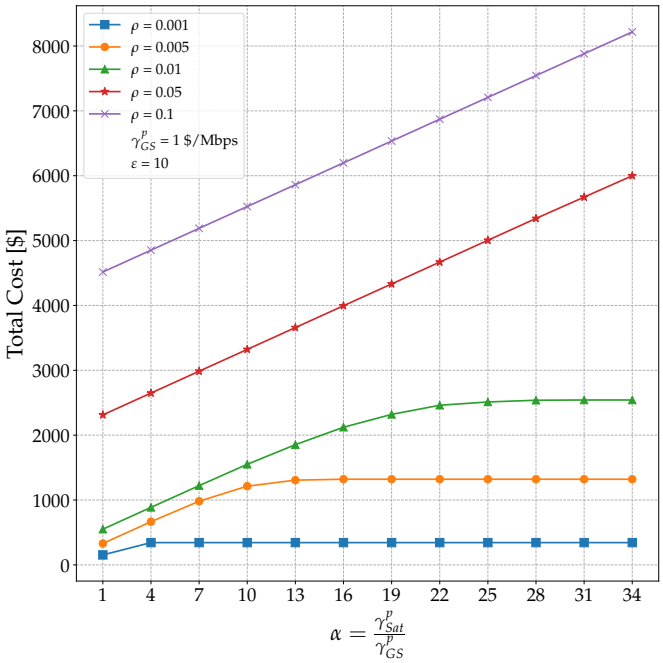

Figure 2.14. Total cost obtained by applying the Shortest Path-based Heuristics for different ρ values, fixing ε parameter.

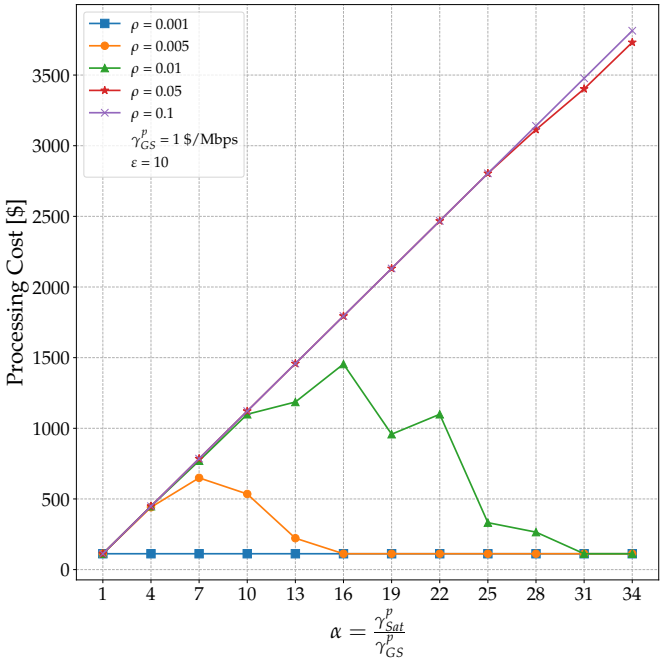

Figure 2.15. Processing cost obtained by applying the Shortest Path-based Heuristics for different ρ values, fixing ε parameter.

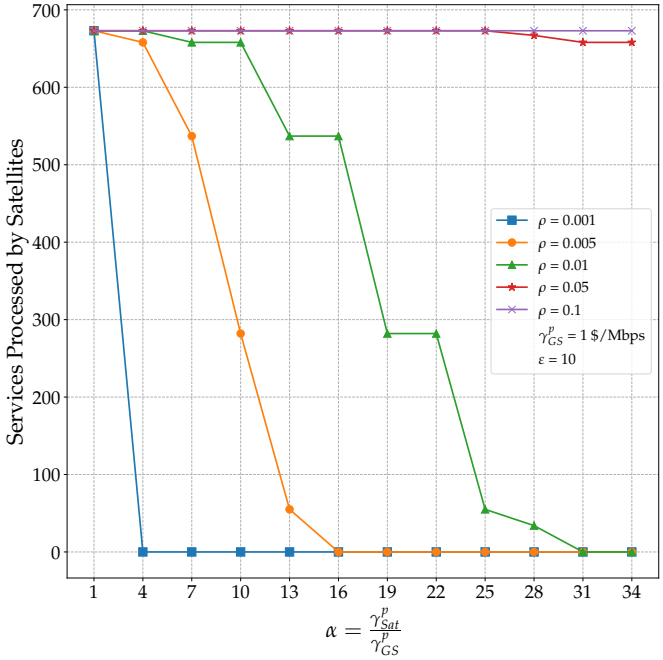

Figure 2.16. Number of services processed by satellites, obtained by applying the Shortest Path-based Heuristics for different ρ values, fixing ε parameter.

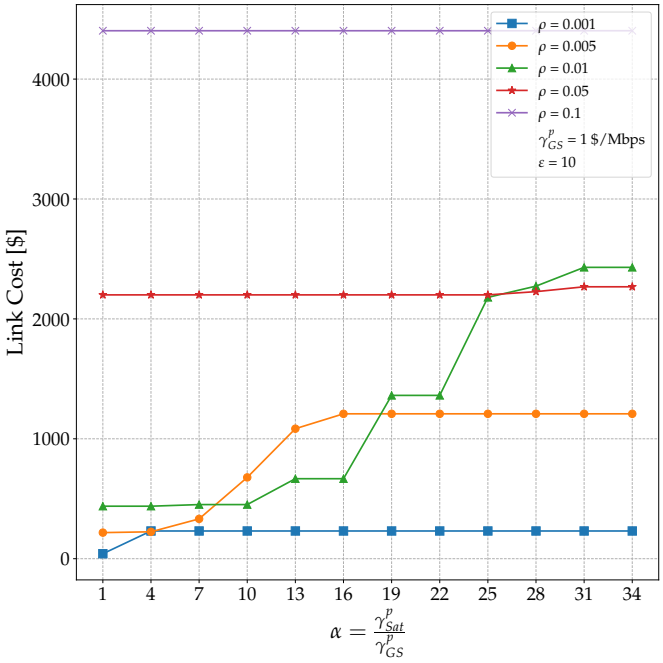

Figure 2.17. Link cost obtained by applying the Shortest Path-based Heuristics for different ρ values, fixing ε parameter.

Another parametric evaluation studies the behaviour of the Shortest Path-based Heuristic with respect to the value of parameter ε, to evaluate the impact of the data reduction amount achievable with processing on the total cost trend. Total cost is shown in Fig.2.18, processing cost is reported in Fig.2.19, number of services processed by satellites is represented in Fig.2.20, while link cost is in Fig.2.21. In this case, it can be noticed that the higher the ε parameter is (i.e., by fixing the preprocessing data size and reducing the postprocessing one), the lower the total cost is, and the wider the region of α values for which there is a convenience in processing services on-board is. The reduction of total cost with respect to the case in which no data reduction happens decreases as the α parameter increases, since the convenience in on-board processing progressively decreases as α increases. However, as ε increases, it is possible to obtain convenience for higher α values, because the data burden reduction leads to a decrease in link cost which could counterbalance the higher cost required for on-board processing. Finally, both the total cost reduction and the width of the region of α values for which there is a cost saving in on-board processing tends to an asymptotic value as ε increases. This is justified by the fact that, as the data burden reduction increases, the link cost saving becomes more marginal while the processing cost remains the same, since it depends on the preprocessing size, which remains fixed for each value of ε. A further confirmation of what has been discussed can be found in the number of services processed by satellites shown in Fig.2.20. In fact, for any ε value, this number decreases with α. Conversely, for any α value, the number of on-board processed services increases with ε up to a maximum. This also allows to conclude that the amount of information processed within the constellation depends also on the nature

of the service, in particular, on the amount of data reduction due to information processing.

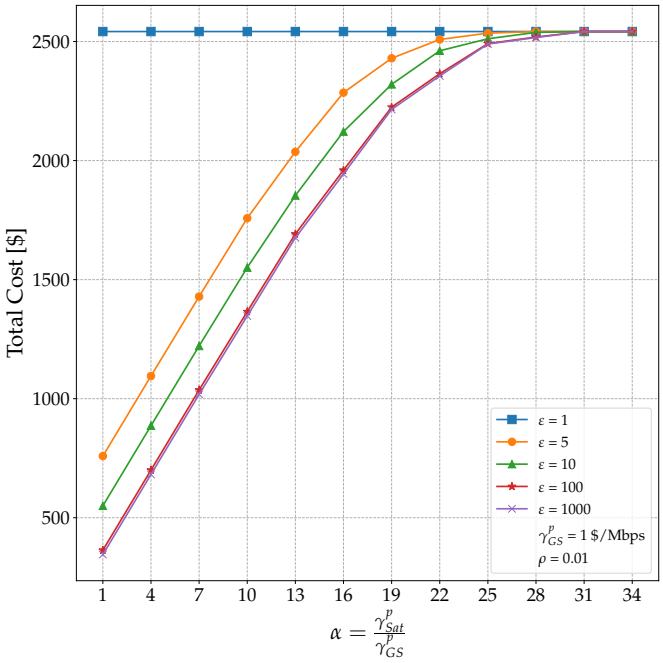

Figure 2.18. Total cost obtained by applying the Shortest Path-based Heuristics for different ε values, fixing ρ parameter.

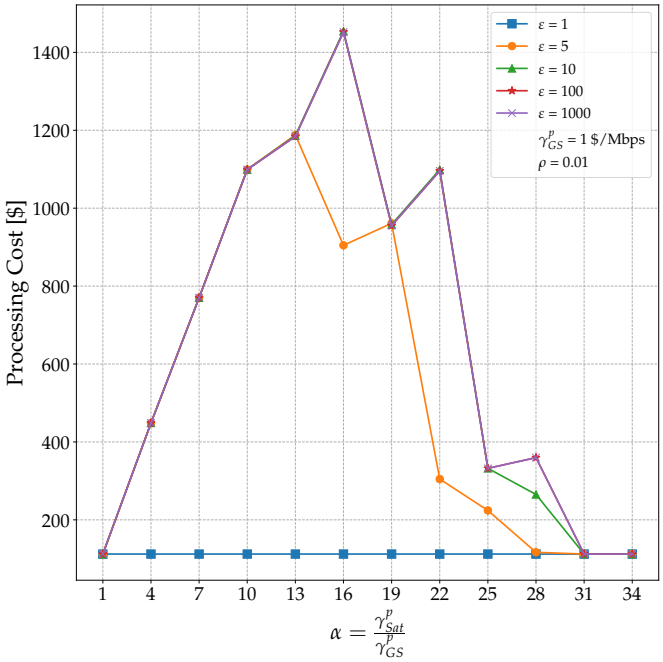

Figure 2.19. Processing cost obtained by applying the Shortest Path-based Heuristics for different ε values, fixing ρ parameter.

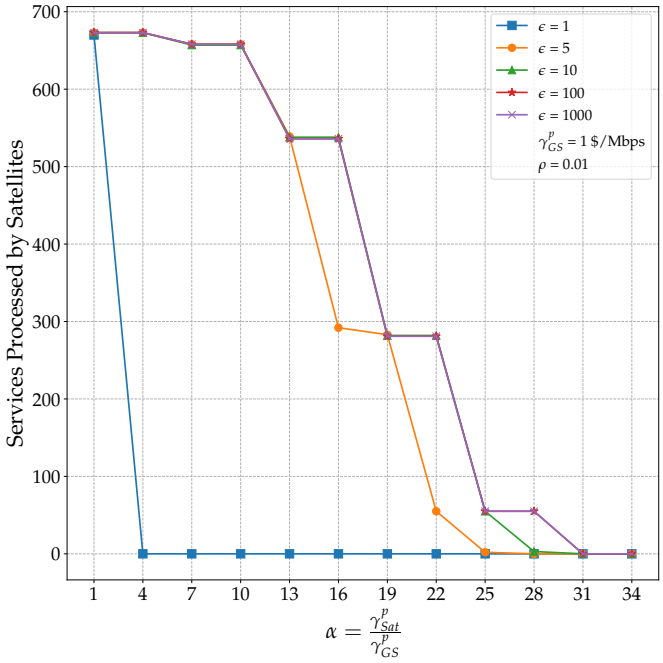

Figure 2.20. Number of services processed by satellites, obtained by applying the Shortest Path-based Heuristics for different ε values, fixing ρ parameter.

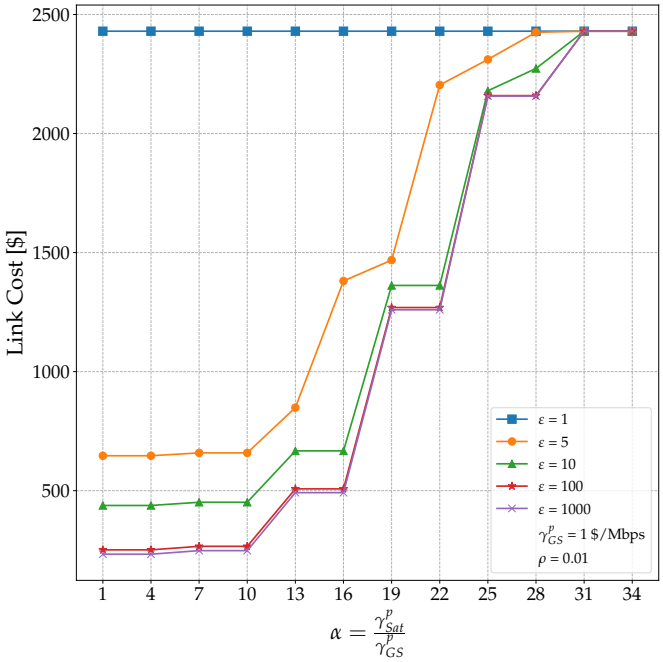

Figure 2.21. Link cost obtained by applying the Shortest Path-based Heuristics for different ε values, fixing ρ parameter.

I finally propose an analysis on the mean service delivery delay, defined as the number of time cycles between the service generation cycle and its deliver to any

ground station. In fact, minimizing the total cost is related to choosing paths with an appropriately low number of hops to maintain a low link cost and this should also translate in a reduced delivery delay. First, even though the objective of the proposed strategies is to minimize the total cost, not the delay, I show the impact that resource allocation obtained through the Shortest Path-based Heuristic has on the mean delivery delay. In particular, Fig.2.22 shows the mean delivery delay as function of α, fixing $\rho = 0.01$ and by varying the ε parameter. Recalling that, as α increases, convenience in processing data on-board of satellites is lost and, thus, the proposed heuristic behaves like the Always Ground case (i.e., all services are processed on ground) for high α values, it can be noticed that the mean delivery delay increases with α. This is due to the fact that the smaller α is, the higher the convenience in processing data in-orbit is, thus, an increased number of services are processed on-board of satellites and the following data reduction allows processed data to leverage an increased number of links to reach the ground station, since links with smaller capacity can be crossed by the processed, lighter, information. It follows that in-orbit processing opens the possibility to route information on a path with a smaller associated cost, and this has an impact on the mean delivery delay, since information arrives earlier on ground, too. This behaviour is also confirmed by noticing that the higher the ε, i.e., the higher data reduction due to processing is, the lower the mean delivery delay is. Finally, the similarity of curves shown in Fig.2.21 and Fig.2.22 confirms that the mean delivery delay is strictly related to the usage of transmission and memory links. In fact, by fixing ε value, an increase in link cost stands for an increase of transmission/memory link usage, and this is reflected in an increase of the mean delivery delay. Similarly, results reported in Fig.2.23 show that, by fixing $\varepsilon = 10$ and studying the behaviour of the mean delivery delay with α, having ρ as a parameter, again the mean delivery delay increases with α, but for each value of α, it decreases when ρ increases. This is due to the fact that as ρ increases, the unit cost to be paid for crossing a transmission or storage link increases, thus, there is more convenience in processing data on-board even for higher α values, in such a way that total cost lowers thanks to a reduced transmission/storage cost, since links are crossed by a reduced amount of data. Again, a higher amount of in-orbit processed information leads to the possibility of leveraging an increased number of links to reach the ground station, and, finally, to a reduced mean delay in making the information available on ground.

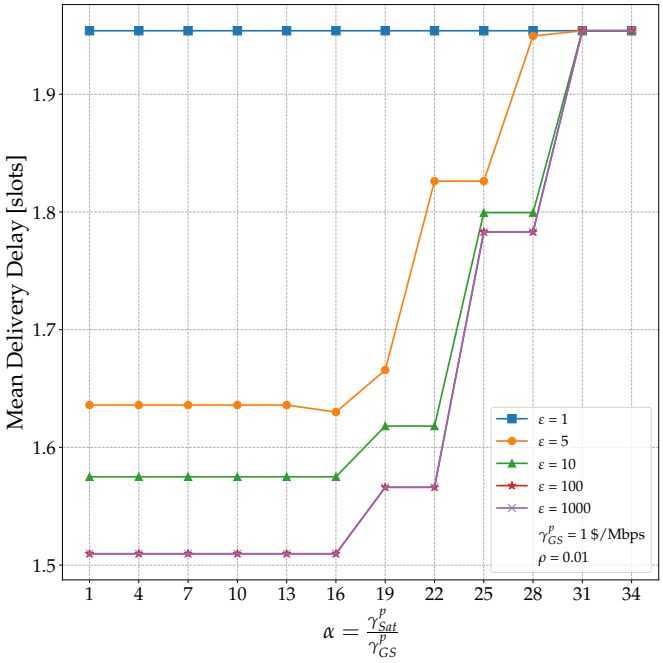

Figure 2.22. Mean delivery delay obtained by applying the Shortest Path-based Heuristics for different ε values, fixing ρ parameter.

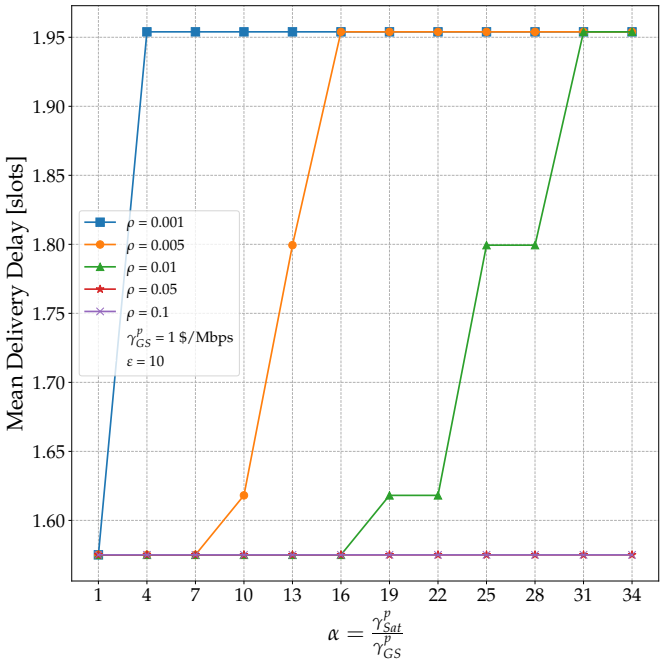

Figure 2.23. Mean delivery delay obtained by applying the Shortest Path-based Heuristics for different ρ values, fixing ε parameter.

Furthermore, since by increasing the number of ground stations there are potentially more possibilities to downlink data on ground, I now consider an increasing number of ground stations, most of which are made available by Amazon Web Services Ground Station[99]. They are grouped as follows:

- 3 GSs: Matera (Italy), Kourou (French Guyana), Kiruna (Sweden); this ground station set is the same considered in the previous analysis;
- 6 GSs: Hawaii (USA), Punta Arenas (Chile) and Singapore are added to the previous set;
- 9 GSs: Ohio (USA), Cape Town (South Africa) and Sidney (Australia) are added to the previous set;
- 12 GSs: Oregon (USA), Bahrein, Soul (South Korea) are added to the previous set.

I compare the mean delivery delays obtained by applying the Shortest Path-based Heuristic (i.e., endowing the constellation with both ISLs and on-board processing capability), the Always Ground benchmark (i.e., ISLs are available to obtain a satellite network, but no processing can happen on satellites) and another benchmark solution, where there are neither ISLs, nor processing capability within the constellation, thus, satellite can only store services in their own memory and deliver them to a ground station when they fly over it (named Dowlink to GS only). The results of this analysis are reported in Fig.2.24. By increasing the number of ground stations, there is a reduction of the mean delivery delay for all the compared solutions, since there are more possibilities to downlink data to any ground station. The highest delay is obtained when there are no ISLs within the constellation and the satellites cannot process services on-board (i.e., in the operational context of EO constellations at the time being). However, by only adding ISLs, there is a strong decrease in mean delivery delay, which is further improved when the constellation is endowed with both ISLs and the ability to process data on-board. In fact, by adding ISLs it is possible to reach any ground station in a shorter time than waiting for flying over it (and this is also linked to a link cost saving); furthermore, since the number of links able to host the processed data is higher than the ones able to host an unprocessed information, by endowing the satellite with on-board processing capability it is possible to obtain the shortest path on a graph with an increased number of links, and this contributes to further lower the mean service delivery delay.

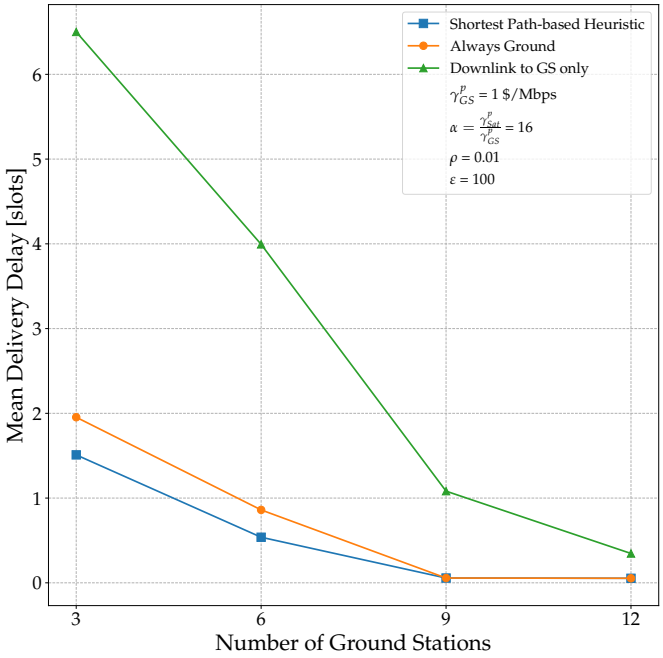

Figure 2.24. Mean service delivery delay obtained by considering an increasing number of ground stations and fixing α, ρ and ε values.

2.6 Conclusions

In this chapter, I formalized and solved an optimal bandwidth and computing resource allocation problem in LEO satellite constellations for EO applications, providing for the leveraging of the satellite network formed by ISLs and by considering that any node of this network can accomplish processing of EO data, with the aim of optimizing the use of valuable resources like bandwidth and in-orbit processing. Because of the high complexity of the optimization problem, I also proposed two heuristics which have been validated by comparison with the optimal solution in a simplified, yet not trivial scenario. Results showed the ability of the proposed heuristics in returning a solution with a total cost not higher than 24% with respect to the optimal one, but with a lower computational effort. Finally, I applied the proposed heuristics to a real orbital scenario to evaluate the impact of leveraging task execution offloading to a satellite different from the originating one, with respect to state-of-the-art solutions which do not consider this possibility. I showed that the proposed strategies allow to better leverage resources if compared to benchmark solutions, leading to lower total operating cost and data delivery delay with respect to state-of-the-art solutions.

Chapter 3

Optimization of the Energy Consumption on Ground Stations

Orbital Edge Computing can also help to achieve green communications objectives[14] by reducing the amount of energy required to process EO data on Earth, while guaranteeing that all acquired images are processed. In fact, even though energy is a valuable resource on satellites, the on-board energy is pre-allocated due to the presence of solar panels and batteries and it is always generated and available, regardless on its actual need and use in time. Instead, energy consumption on the ground is strictly dependent on the demand, and it increases with the increase of EO data to be processed by ground stations. Furthermore, while energy harvested on satellites is completely renewable, this is not always the case on ground. In this chapter, I first define and solve an optimization problem to jointly allocate resources and place processing within a constellation-wide network to leverage in-orbit processing as much as possible. This is very different from approaches followed in literature, focusing only on a minimization of the energy use on satellites [11] or on the optimization of battery use to extend satellite operative life [12]. Instead, the proposed optimization aims at reducing the amount of data to be processed on ground, thus, to maximize the energy saving on ground stations. Given the NP-hardness of the proposed optimization problem, I also propose the Ground Station Energy Saving Heuristic (GSESH) algorithm to evaluate the energy saving we would obtain on ground stations in a real orbital scenario. However, since these strategies may lead to a decrease of satellite operative life because of an increased use of batteries, I propose a more complex optimization problem allowing for jointly maximizing on-ground energy saving and optimizing on-board battery DoD to increase the satellite operative life.

Results presented in this chapter have been published in [100, 101].

3.1 Scenario Overview and Problem Outline

In this chapter, the same reference scenario described in Section 2.1 is considered. As it has been previously discussed, by designing appropriate strategies to decide

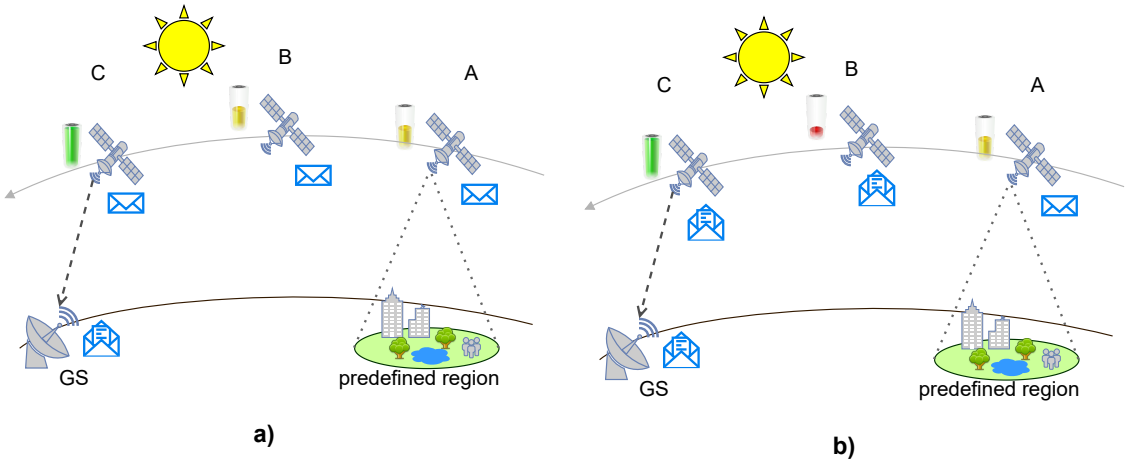

Figure 3.1. Example of different use of the energy available on board of a satellite. In case a), no processing happens in orbit, while, in case b), the satellite processes the acquired image. In both cases, when reaching position C, the satellite has fully charged batteries because it is sunlit.

how each satellite has to deal with each image at every moment (i.e., to determine whether it shall process it, store it, or transmit it), it is possible to optimize desired metrics. In this chapter, I focus on proposing resource allocation strategies which leverage the amount of energy available on satellites to process images directly in-orbit, allowing for saving energy on ground stations, due to a reduced amount of data processing to be done on Earth. This is made possible by the fact that energy on-board of satellites is pre-allocated by endowing them with a certain dimension of solar panels and batteries, and there may be time instants during which an amount of available, but unused energy can be leveraged to process data on-board. In particular, I propose a centralized strategy, justified by the fact that both ISLs availability and satellite overflight time over region of interests are known in advance by means of orbital mechanics, making it possible to define routing and processing placement by running the defined strategies on Earth and then loading the decisions on satellites. The proposed approach is very different from other works which instead provide for minimizing energy usage on satellites, since it is a valuable resource given the limitations in dimension and mass of solar panels and batteries. However, in my opinion such a minimization does not allow for an efficient use of the energy available on board. In fact, let me discuss the example in Fig.3.1. In particular, in both cases a) and b) there is a satellite first acquiring an image of a predefined region, then moving on its orbit until reaching a position B where satellite is sunlit and the battery can be charged by means of solar panels, and finally reaching a position C where the acquired information can be downlinked to the ground. In case a), no processing happens in-orbit. In particular, the satellite acquires the image, and when it reaches position B it has half battery charge. However, from B to C the satellite is sunlit, and during the movement from B to C, battery gets fully charged. Finally, when the satellite reaches the position C, the acquired image is downlinked and processed by the ground station. Instead, in case b), the satellite processes the acquired image when it reaches position B. For this reason, on board battery has a

smaller charge with respect to position B in the former case. However, in the path from B to C, the satellite receives enough solar radiation to charge the battery even though the initial energy is lower. Thus, battery is again fully charged when the satellite sends the already processed image to the GS in position C, and at the same time energy on ground is saved. From this example, it follows that, even though on board energy is a limited and valuable resource, it is renewed whenever the satellite is sunlit, and for this reason it is not important to minimize its usage, but to optimize it in such a way that by leveraging, when possible, the energy available on satellites, the consumption on ground can be reduced. Obviously, the usage of extra energy available on board of satellites may lead to an increased number of battery charge-discharge cycles, and this may reduce the operative life of each satellite. For this reason, in this chapter I first study the energy saving which can be obtained by leveraging OEC and satellite networks without considering the optimization of operative life, in order to identify how the processing capacity on-board of satellites and the amount of available energy influence the on-ground energy saving. Second, I propose a more complex optimization problem aiming to maximize the on-ground energy saving, while also optimizing the DoD of on-board batteries.

3.2 Network and Image Processing Service Representation

Given the similarity of the reference scenario and problem statement discussed in Section 3.1 with the ones introduced in Section 2.1, it follows that the service modelling is the same as described in Section 2.2. Instead, I propose a different formal network modelling to obtain a more elegant formulation of the optimization problem and heuristic discussed in the following sections. Network and service model sets and parameters introduced in this Section are summarized in Tab.3.1.

Let me thus introduce the network model leveraged in this chapter. As it has been previously discussed, since Earth and satellite motions are periodic, a cyclostationary behaviour for both topology and service generation can be assumed. Again, by calling T_c the repetition period of the satellites and Earth relative motion, this T_c period is considered to be discretized in T time cycles, each having τ duration. This leads to the definition of a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ represents the node set, while $\mathcal{E}$ is the edge set. In particular, to deal with the dynamical behaviour of the topology, nodes in the graph are not simply a representation of a physical node (i.e., a satellite or ground station), but of a physical node at a certain time cycle which only deals with services in a specific processing state (i.e., either unprocessed or processed). Let me clarify this assumption. The physical nodes composing the topology are satellites and ground stations. As far as ground stations are concerned, I still assume that all the ground stations are the same and thus there is no preference on which ground station images shall reach, and for this reason I only consider a Virtual Ground Station (vGS) node in the topology, representing any of the ground stations enabled to receive data from the constellation. Thus, the satellites and the vGS lead to a topology having N_S physical nodes. This translates into a graph whose node $n_i^{t,p} \in \mathcal{N}$, with $i \in [0, \dots, N_S - 1]$, $t \in [0, \dots, T - 1]$, $p \in [0, 1]$, represents not simply the i-th physical node, but the i-th physical node dealing only with services

Table 3.1. Network and service model sets and parameter

Set or parameter	Description
$\mathcal{G} = (\mathcal{N}, \mathcal{E})$	graph made by node set $\mathcal{N}$ and edge set $\mathcal{E}$
T_c	cyclostationary period for both topology and service generation
τ	discrete time cycle duration
T	number of time cycle of duration τ in a cyclostationary period
N_S	number of physical nodes (either satellites or vGS)
i	index in $[0, \ldots, N_S - 1]$ representing the physical node
t	index in $[0, \ldots, T - 1]$ representing the time cycle
p	index in $\{0, 1\}$ representing the processing state (0 standing for unprocessed, 1 processed)
$n_i^{t,p}$	node in the set $\mathcal{N}$
$e^t_{\{i_s,p_s\},\{i_d,p_d\}}$	edge in $\mathcal{E}$ between nodes $n_{i_s}^{t,p_s}$ and $n_{i_s}^{t,p_s}$
Γ_i	processing capacity associated to the i-th physical node (in Mbps)
$\gamma^{i_d,p_d}_{i_s,p_s}$	energy cost to be paid for a Mb of data crossing $e^t_{\{i_s,p_s\},\{i_d,p_d\}}$ edge (in J/Mb)
vGS	Virtual Ground Station represented by the $i = N_S - 1$ physical node
$C^t_{\{i_s,p_s\},\{i_d,p_d\}}$	capacity associated to the $e^t_{\{i_s,p_s\},\{i_d,p_d\}}$ edge
M_i	data storage capacity associated to the i-th physical node (in Mb)
ε_i	energy available at the beginning of each time cycle on the i-th physical node (in J)
G	antenna gain
ν_{tx}	carrier frequency
P	transmission power
R_{i_s,i_d}	transmission data rate between i_s-th and i_d-th physical nodes
B	transmission bandwidth
T_s	system noise temperature
$\hat{\tau}^t_{i_s,i_d}$	actual visibility time between i_s-th and i_d-th physical nodes during t-th time cycle
k	processor effective capacitance coefficient
n_{cyc}	number of CPU cycles to process 1 bit
ν_i	clock frequency of the processor on the i-th physical node
Σ	set of all generated services
f^h	service in Σ, with $h \in [0, \ldots, N_T - 1]$
N_T	total number of generated services
f^h_s	f^h service source satellite index
f^h_0	f^h service pre-processing size
f^h_1	f^h service post-processing size
f^h_t	f^h service generation time cycle
f^h_d	number of time cycles after generation within which f^h service shall be delivered to the vGS

in the p-th processing state (with 0 standing for unprocessed, 1 for processed) during the t-th time cycle. This will allow to define optimal strategies able to consider the dynamicity of the orbital environment, and by dividing nodes handling unprocessed and processed services, a straightforward optimization problem formulation can be obtained, without having to consider virtual links to deal with the impact of data reduction due to processing on network resources. I arbitrarily assume that the vGS is represented by the i-th node when $i = N_S - 1$. Furthermore, each physical i-th node is endowed with a maximum energy ε_i at the beginning of each time cycle (expressed in J).

As far as edges are concerned, $e^t_{\{i_s,p_s\},\{i_d,p_d\}} \in \mathcal{E}$, with $i_s,\ i_d \in [0,\dots,N_S-1],\ t \in [0,\dots,T-1],\ p_s,\ p_d \in [0,1]$ represents a link between two nodes. Each edge is associated with a capacity $C^t_{\{i_s,p_s\},\{i_d,p_d\}}$ expressed in Mb. In particular, the following link types can be distinguished:

- if $i_s \neq i_d$, $p_s = p_d = p$, the edge represents a transmission link, associated with a capacity representing the maximum amount of information that can be transferred during the actual visibility time between the two physical nodes in the t-th time cycle. In particular, assuming an Additive White Gaussian Noise channel, the actual visibility time $\hat{\tau}^t_{i_s,i_d}$ between two satellite nodes i_s and i_d, with $i_s,\ i_d \in [0,\dots,N_S-2]$, during the t-th time cycle is given by the time interval of the considered time cycle during which the distance $d^t_{i_s,i_d}$ between the nodes satisfies the following relationship:

$$d^t_{i_s,i_d} \leq \frac{G\ c}{4\pi\nu_{tx}} \sqrt{\frac{P}{\left(2^{\frac{R_{i_s,i_d}}{B}}-1\right) k_B T_s B}} \tag{3.1}$$

  where G is the antenna gain, c is the speed of light, ν_{tx} is the carrier frequency, P is the transmission power, R_{i_s,i_d} is the transmission data rate between the nodes, B is the bandwidth, k_B is the Boltzmann's constant, T_s is the system noise temperature. Instead, in case the destination node is the vGS, i.e., $i_d = N_S - 1$, I assume the actual visibility time $\hat{\tau}^t_{i_s,N_S-1}$ to be the time interval during the t-th time cycle during which the communication between the satellite and a ground station is possible with a minimum elevation angle El_{min}. Please notice that, during each time cycle, the vGS node will have an edge with each node related to satellite being able to communicate with any ground station during that time cycle. After having determined the actual visibility time, the capacity is given by:

$$C^t_{\{i_s,p\},\{i_d,p\}} = R_{i_s,i_d} \cdot \hat{\tau}^t_{i_s,i_d} \tag{3.2}$$

  Thus, the capacity of the transmission link between two different physical nodes in a certain time cycle represents the maximum amount of data in Mb that can be transferred from the sending node to the receiving one within a time cycle. Please notice that, even though the capacity of a transmission link is generally expressed in Mbps, in this modelling I prefer indicating it in Mb, since this allows for an elegant formulation of the optimization problem which

will be introduced further on. Finally, the use of this link is associated with an energy cost $\gamma_{i_s,p}^{i_d,p}$ (in J/Mb) representing the amount of energy spent in transmitting 1 Mb of data, given by:

$$\gamma_{i_s,p}^{i_d,p} = \frac{P}{R_{i_s,i_d}} \tag{3.3}$$

In other words, when a generic amount x (in Mb) of data is transferred from the i_s-th physical node to the i_d-th one, the former spends an amount of energy ε_{tx} given by (in J):

$$\varepsilon_{tx} = x \cdot \gamma_{i_s,p_s}^{i_d,p_d} \tag{3.4}$$

while I assume that no energy consumption happens on the receiving node. For this reason, since in the considered model vGS never sends data to the satellites, but it only receives it, it is straightforward that data transmission will not contribute to energy consumption on ground.

- if $i_s = i_d = i$, $p_s = p_d = p$, the edge represents the storage of the information on a node during the full t-th time cycle (i.e., a memory link), and it is associated with a capacity:

$$C_{\{i,p\},\{i,p\}}^{t} = M_i \tag{3.5}$$

representing the memory amount available on the i-th node; I assume that no energy is needed to store data, i.e., the use of this link is associated with an energy cost:

$$\gamma_{i,p}^{i,p} = 0 \tag{3.6}$$

This holds for any node. Thus, it follows that storage in memory on ground will not contribute to the energy consumption of ground stations.

- if $i_s = i_d = i$, $p_s = 0$, $p_1 = 1$, the edge represents the service processing accomplished by the node during the t-th time cycle (i.e., a processing link), which is associated with a capacity:

$$C_{\{i,0\},\{i,1\}}^{t} = \Gamma_i \cdot \tau \tag{3.7}$$

representing the amount of data in Mb which can be processed in a time cycle, since Γ_i represents the processing capacity of the node in Mbps. Again, the processing capacity is expressed in Mb instead of Mbps to obtain an elegant formalization of the optimization problem presented further on. The use of this link is associated with a unit energy cost (in J/Mb):

$$\gamma_{i,0}^{i,1} = k \cdot n_{cyc} \cdot \nu_i^2 \tag{3.8}$$

representing the amount of energy to be spent to process 1 Mb of data, which depends on the processor effective capacitance coefficient (k), on the number of CPU cycles to process 1 bit (n_{cyc}) and on the clock frequency of the processor on the i-th physical node (ν_i)[11]. It follows that the amount of energy that a i-th node spends to process an amount x in Mb of data is given by (in J):

$$\varepsilon_{proc} = x \cdot \gamma_{i,0}^{i,1} \tag{3.9}$$

Since any node, either satellite or vGS, can process data, this amount of energy contributes to the energy consumption of ground stations.

- all combinations of i_s, i_d, p_s and p_d not mentioned before represent edges not included in the graph, since they have no logical meaning. In the optimization problem formulation, they will be considered to have no associated capacity and no energy cost.

To summarize, the proposed modelling provides that each satellite consumes energy when it transmits or processes data. Instead, the only contribution to energy consumption on ground is given by the processing happening on ground stations, i.e., by the energy spent to cross processing links associated to the vGS in the proposed model.

3.3 Strategies without limitations on satellite battery Depth-of-Discharge

In this section, I propose and evaluate strategies to maximize the energy saving on ground stations related to EO image processing. In the following proposal, I will not consider any optimization of satellite operative life, being strictly related to the DoD of on-board batteries[12]. In particular, there will be no distinction on the on-board energy source (i.e., solar panels or batteries) during each time cycle. This assumption can be justified by the fact that when a satellite is in eclipse, batteries should be dimensioned in such a way that, at any time, they are able to provide at least the same energy generated by solar panels when the satellite is sunlit.

3.3.1 Optimal Strategy

The network and service modeling presented in Section 3.2 leads to a straightforward formalization of the optimal resource allocation problem to minimize the energy consumption on ground due to data processing (i.e., to maximize the on-ground energy saving), under energy, bandwidth, storage and processing capacity constraints. In particular, since each node in the graph on which the problem is defined represents a physical node at a certain time, the proposed formulation allows to consider the dynamic environment. Furthermore, since in my modelling I distinguished nodes dealing with unprocessed and processed tasks, it is possible to obtain an Integer Linear Programming (ILP) formulation able to take into account the data reduction because of processing without having to introduce virtual links and additional constraints which would increase the problem complexity.

First, I can introduce the binary optimization variables as defined in (3.10).

$$y^{t,h}_{\{i_s,p_s\},\{i_d,p_d\}} = \begin{cases} 1 & \text{if } h\text{-th task occupies the link from node } (i_s, p_s) \\ & \text{to node } (i_d, p_d) \text{ during } t\text{-th time cycle} \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

It can be noticed that the joint routing and processing placement problem can be defined as finding the path for each service globally optimizing the desired metric on the graph previously introduced, since in that formulation links can be related to either transmission/storage or processing. In particular, recalling that the only

contribution to energy consumption on ground is given by data processing, I can express the ground station energy consumption as follows:

$$\varepsilon_{GS}^{tot} = \sum_{h=0}^{N_T-1} \sum_{t=0}^{T-1} y^{t,h}_{\{N_S-1,0\},\{N_S-1,1\}} \; f_0^h \; \gamma^{N_S-1,1}_{N_S-1,0} \tag{3.11}$$

For this reason, I can introduce the objective function as follows, since maximizing the energy saving on ground stations is equivalent to minimize the energy consumption on ground:

$$\min \varepsilon_{GS}^{tot} \tag{3.12}$$

The proposed graph also simplifies the formalization of the constraints, which can be written as reported in (3.13)-(3.17). In their definition, I leveraged the Kronecker's introduced in (2.17).

$$\sum_{h=0}^{N_T-1} \left[(1-\delta_{p_s,p_d}) \; y^{t,h}_{\{i_s,p_s\},\{i_d,p_d\}} \; f^h_{p_s} + \delta_{p_s,p_d} \sum_{p=0}^{1} y^{t,h}_{\{i_s,p\},\{i_d,p\}} \; f^h_{p_s} \right] \leq C^t_{\{i_s,p_s\},\{i_d,p_d\}},$$
$$\forall i_s, i_d \in [0,\dots,N_S-1], \; \forall p_s, p_d \in \{0,1\}, \; \forall t \in [0,\dots,T-1] \tag{3.13}$$

$$\sum_{h=0}^{N_T-1} \sum_{i_d=0}^{N_S-1} \sum_{p_s=0}^{1} \sum_{p_d=0}^{1} y^{t,h}_{\{i_s,p_s\},\{i_d,p_d\}} \; f^h_{p_s} \; \gamma^{i_d,p_d}_{i_s,p_s} \leq \varepsilon^t_{i_s}, \tag{3.14}$$
$$\forall i_s \in [0,\dots,N_S-1], \; \forall t \in [0,\dots,T-1]$$

$$(1-\delta_{f_t^h,f_d^h}) \cdot y^{f_t^h,h}_{\{f_s^h,0\},\{f_s^h,0\}} + y^{f_t^h,h}_{\{f_s^h,0\},\{f_s^h,1\}} + \sum_{\substack{i_d=0 \\ i_d \neq f_s^h}}^{N_S-1} y^{f_t^h,h}_{\{f_s^h,0\},\{i_d,0\}} = 1, \tag{3.15}$$
$$\forall h \in [0,\dots,N_T-1]$$

$$\begin{aligned}(1-\delta_{i,f_s^h}\delta_{t,f_t^h}\delta_{p,0} - \delta_{i,N_S-1}\delta_{t,f_t^h+f_d^h}\delta_{p,1})\{ &\left[(1-\delta_{t,f_t^h}) \cdot y^{(t-1) \bmod T,h}_{\{i,p\},\{i,p\}} + \right. \\ \left. \delta_{p,1} \cdot y^{t \bmod T,h}_{\{i,0\},\{i,1\}} + \sum_{\substack{a=0 \\ a\neq i}}^{N_S-1} y^{t \bmod T,h}_{\{a,p\},\{i,p\}} \right] - &\left[(1-\delta_{t,f_t^h+f_d^h}) \cdot y^{t \bmod T,h}_{\{i,p\},\{i,p\}} + \right. \\ \left. \delta_{p,0} \cdot y^{t \bmod T,h}_{\{i,0\},\{i,1\}} + \sum_{\substack{b=0 \\ b\neq i}}^{N_S-1} y^{t \bmod T,h}_{\{i,p\},\{b,p\}} \right] \} = 0\end{aligned} \tag{3.16}$$
$$\forall i \in [0,\dots,N_S-1], \; \forall h \in [0,\dots,N_T-1],$$
$$\forall t \in [f_t,\dots,f_t^h+f_d^h], \; \forall p \in \{0,1\}$$

$$(1-\delta_{f_d^h,0})\cdot y^{(f_t^h+f_d^h-1)\ mod\ T,h}_{\{N_S-1,1\},\{N_S-1,1\}} + y^{(f_t^h+f_d^h)\ mod\ T,h}_{\{N_S-1,0\},\{N_S-1,1\}} + \sum_{i_s=0}^{N_S-2} y^{(f_t^h+f_d^h)\ mod\ T,h}_{\{i_s,1\},\{N_S-1,1\}} = 1, \quad \forall h \in [0,\dots,N_T-1] \tag{3.17}$$

In particular, constraint (3.13) ensures that the resource allocation does not exceed the capacity of each link (notice that thanks to the proposed modeling, this expression is valid for all transmission, storage and processing links). Constraint (3.14) limits the energy consumption during each time cycle on each node as the maximum energy amount available on the node at the beginning of the time cycle. Constraints (3.15)-(3.17) are related to flow conservations. Specifically, following the characteristics of the considered EO applications, constraint (3.15) sets the source node for each service as the one related to the satellite which acquired the image, dealing with an unprocessed data, at the service generation cycle, and ensures that node has only an outgoing flow for each service. Specularly, constraint (3.17) sets the destination node for each service as the vGS dealing with the processed data at service delivery deadline time cycle, and ensures that node has only an ingoing flow for each service. Please notice that constraint (3.17) also ensures that all services are processed and do not prevent a service to reach the ground station before the service delivery deadline. Finally, constraint (3.16) simply imposes flow conservation on intermediate nodes on the path each service crosses from source to destination.

As also discussed in Section 2.3, since both topology and service generation are cyclostationary, it is just necessary to define and solve the problem in a cyclostationary period, since what happens in that period repeats all mission long. For this reason, constraints (3.13)-(3.14) shall be written only for the time cycles composing a cyclostationary period, while in constraints (3.15)-(3.17), since a service delivery deadline cycle could exceed a the end of the current cyclostationary period, I leverage modulo operation to bring back all that would happen after the end of the cyclostationary period to its beginning, since cyclostationary periods repeat always the same one after another.

Finally, I discuss the complexity of the proposed formulation. In particular, it can be noticed that it is analogous to the decision multi-commodity flow problem, to which an additional constraint related to energy is added. For this reason, since the decision multi-commodity flow problem is NP-complete, the same applies to this problem.

3.3.2 Heuristic-based Strategy

Since the presented optimal strategy is NP-complete, in order to evaluate the benefits of the proposed allocation strategy in a complex scenario like a real orbital case, it is necessary to introduce a heuristic mimicking the optimization problem solution. I propose a heuristic leveraging the Dijkstra's algorithm, as detailed in Alg.3. This algorithm applies on a new graph $\tilde{\mathcal{G}} = \left(\tilde{\mathcal{N}}, \tilde{\mathcal{E}}\right)$ being a modified version of the graph presented in Section 3.2, where instead of distinguishing nodes dealing with only two processing state, a third processing state is also considered, the "in-processing" state. This is due to the fact that the introduction of the "in-processing" layer allows to

take into account that a node processing a task can transmit it in the same time cycle during which processing happens only if the energy remaining after processing is enough to accomplish the transmission of a processed task. In fact, while this condition was verified by means of appropriate constraints in the optimization problem definition, in the case of the proposed shortest-path based heuristic, it is necessary to input the algorithm with a graph containing only admissible paths. For this reason, while defining the heuristic, I consider $p \in \{0, 1, 2\}$, where $p = 0$ represent a node dealing with unprocessed tasks, $p = 1$ a node dealing with in-processing tasks and $p = 2$ a node handling processed tasks. In particular, while transmission and storage links remain the same as defined in Section 3.2, processing links are distinguished as follows:

- if $i_s = i_d = i$, $p_s = 0$, $p_1 = 1$, the edge represents the service processing accomplished by the i-th physical node during the t-th time cycle (i.e., a processing link), with associated capacity and energy cost as discussed in the processing link definition given in Section 3.2;
- if $i_s = i_d = i$, $p_s = 1$, $p_1 = 2$, the edge represents a fictitious link to bridge nodes in the processing state to nodes in the processed state, associated with an infinite capacity and no energy cost.

Algorithm 3: Ground Station Energy Saving Heuristic (GSESH)

Input: Σ, $\tilde{\mathcal{N}}$, $\tilde{\mathcal{E}}$, β
Initialize: $\Lambda^t_{\{i_s,p_s\},\{i_d,p_d\}} \leftarrow 0$, $\Psi^t_{i_s} \leftarrow 0$, $\forall i_s, i_d \in [0, \dots, N_S - 1]$, $p_s, p_d \in [0, 2]$, $t \in [0, \dots, T - 1]$;
Initialize: $\Pi \leftarrow \emptyset$;
for $f^h \in \Sigma$ **do**
  $\hat{\mathcal{N}}, \hat{\mathcal{E}} \leftarrow Extract\ Subgraph(f^h, \Lambda_l, \tilde{\mathcal{N}}, \tilde{\mathcal{E}}, \beta)$;
  $\hat{\mathcal{G}} \leftarrow (\hat{\mathcal{N}}, \hat{\mathcal{E}})$;
  $\hat{\Pi} \leftarrow Dijkstra\left(\hat{\mathcal{G}}, n^{f^h_t,0}_{f^h_s}, n^{(f^h_t+f^h_d)\ mod\ T,2}_{N_S-1}\right)$;
  if $\hat{\Pi} \neq \emptyset$ **then**
    $\Pi \leftarrow \Pi \cup \hat{\Pi}$;
    for $\{\hat{i}_s, \hat{p}_s, \hat{i}_d, \hat{p}_d, \hat{t}\} \in \hat{\Pi}$ **do**
      Calculate $\Lambda^{\hat{t}}_{\{\hat{i}_s,\hat{p}_s\},\{\hat{i}_d,\hat{p}_d\}}$ by means of eq.3.18;
      Calculate $\Psi^{\hat{t}}_{\hat{i}_s}$ by means of eq.3.19;
    end
    for $t \in [f^h_t, \dots, f^h_t + f^h_d]$ **do**
      $\beta^{t\ mod\ T}_{f^h_s} \leftarrow \beta^{t\ mod\ T}_{f^h_s} - \{f^h\}$;
    end
  else
    $\Omega \leftarrow \Omega \cup \{f^h\}$;
  end
end
Output: Π, Ω, Λ

Let me comment the main steps of the Ground Station Energy Saving Heuristic (GSESH) presented in Alg.3. After having initialized the matrices tracking the link

occupancy and node energy usage in time (Line 1), the following steps are applied for each service to be allocated. First, a subgraph containing only nodes related to time cycles between the service generation and delivery deadline cycles and only edges with enough capacity to host the unprocessed service and related to nodes with enough energy to accomplish its transmission and/or processing is extracted (Line 3). Subgraph extraction is obtained by applying Alg.4, which will be discussed in detail further on. The Dijkstra's algorithm is then applied on the extracted subgraph, in order to determine the shortest path from the node associated to the source satellite in the unprocessed state during the service generation cycle to the node representing the ground station in the processed state at service delivery deadline cycle (Line 5). While applying the Dijkstra's algorithm, distances are related to energy consumption due to crossing edges between couples of nodes, but since the goal is to minimize the energy consumption on ground stations, only energy to be paid to process data on ground stations is considered while calculating path distances, zeroing any contribution given by crossing a link different from a processing link of the *vGS* node. Then, if a path has been found, the path is added to the list of paths (Line 7), thus, in Lines 8-11, first the matrix tracking the link occupancy is updated to consider the hosting of the current service on the crossed links by means of the following expression:

$$\Lambda^{\hat{t}}_{\{\hat{i}_s,\hat{p}_s\},\{\hat{i}_d,\hat{p}_d\}} = \Lambda^{\hat{t}}_{\{\hat{i}_s,\hat{p}_s\},\{\hat{i}_d,\hat{p}_d\}} + f^h_{\lceil \hat{p}_s/2 \rceil} \tag{3.18}$$

second, similarly, the matrix tracking the node energy usage in time is updated to consider the hosting of the current service on nodes crossed by the service with the following expression:

$$\Psi^{\hat{t}}_{\hat{i}_s} = \Psi^{\hat{t}}_{\hat{i}_s} + \gamma^{\hat{i}_d,\hat{p}_d}_{\hat{i}_s,\hat{p}_s} \cdot f^h_{\lceil p_s/2 \rceil} \tag{3.19}$$

and, finally, the current service is removed from the buffer tracking the services still to be handled by the algorithm (Lines 12-14); otherwise, the current service is included into the set of rejected services, i.e., services which cannot be handled because of lack of resources (Lines 15-17).

As far as the *Extract Subgraph* algorithm illustrated in Alg.4 is concerned, it first allows for extracting from the graph only the nodes which can potentially be traversed by the current service, i.e., all nodes between the service generation time cycle and the service delivery deadline one (Lines 8-9). Furthermore, this algorithm also allows the following Dijkstra's algorithm application dealing with the fact that links have a finite capacity and nodes have a finite amount of energy. In particular, in Lines 7-15, the first condition to be verified to include an edge between a couple of nodes in the subgraph is that the edge has enough capacity to handle the current service, by checking the following condition:

$$\Lambda^{\hat{t}}_{\{i_s,p\},\{i_d,p\}} + f^h_{\lceil p/2 \rceil} \leq C^{\hat{t}}_{\{i_s,p\},\{i_d,p\}} \tag{3.20}$$

Then, a transmission edge between two nodes in the unprocessed (processed) state is included in the subgraph only if the transmitting node has enough energy to transmit the unprocessed (processed) service plus a margin. This is checked by means of the following expression:

Algorithm 4: Extract Subgraph

Input: f^h, Λ_l, $\mathcal{N}$, $\mathcal{E}$, β

1 Initialize: $\hat{\mathcal{N}} \leftarrow \emptyset$, $\hat{\mathcal{E}}_{ts} \leftarrow \emptyset$, $\hat{\mathcal{E}}_{pf} \leftarrow \emptyset$;
2 **for** $i_s \in [0, \dots, N_S - 2]$ **do**
3 Initialize: $\phi \leftarrow 1$;
4 **for** $t \in [f_t^h, \dots, f_t^h + f_d^h]$ **do**
5 $\hat{t} \leftarrow t \ mod \ T$;
6 Calculate $\mu_{i_s}^{\hat{t}}$ by means of eq.3.25;
7 **for** $p \in [0, \dots, 2]$ **do**
8 $\hat{\mathcal{N}} \leftarrow \hat{\mathcal{N}} \cup \{n_{i_s}^{\hat{t},p} \mid n_{i_s}^{\hat{t},p} \in \mathcal{N}\}$;
9 $\hat{\mathcal{N}} \leftarrow \hat{\mathcal{N}} \cup \{n_{N_S-1}^{\hat{t},p} \mid n_{N_S-1}^{\hat{t},p} \in \mathcal{N}\}$;
10 **for** $i_d \in [0, \dots, N_S - 1]$ **do**
11 **if** *conditions in 3.20 & 3.21 are verified* **then**
12 $\hat{\mathcal{E}}_{ts} \leftarrow \hat{\mathcal{E}}_{ts} \cup \{e_{\{i_s,p\},\{i_d,p\}}^{\hat{t}} \mid e_{\{i_s,p\},\{i_d,p\}}^{\hat{t}} \in \mathcal{E}\}$
13 **end**
14 **end**
15 **end**
16 **if** $\phi = 1$ **then**
17 **if** *conditions in 3.22 & 3.23 are verified* **then**
18 $\hat{\mathcal{E}}_{pf} \leftarrow \hat{\mathcal{E}}_{pf} \cup \{e_{\{i_s,0\},\{i_s,1\}}^{\hat{t}} \mid e_{\{i_s,0\},\{i_s,1\}}^{\hat{t}} \in \mathcal{E}\}$;
19 **if** *condition in 3.24 is verified* **then**
20 $\hat{\mathcal{E}}_{pf} \leftarrow \hat{\mathcal{E}}_{pf} \cup \{e_{\{i_s,1\},\{i_s,2\}}^{\hat{t}} \mid e_{\{i_s,1\},\{i_s,2\}}^{\hat{t}} \in \mathcal{E}\}$
21 **end**
22 $\phi \leftarrow 0$
23 **end**
24 **else**
25 $\hat{\mathcal{E}}_{pf} \leftarrow \hat{\mathcal{E}}_{pf} \cup \{e_{\{i_s,1\},\{i_s,2\}}^{\hat{t}} \mid e_{\{i_s,1\},\{i_s,2\}}^{\hat{t}} \in \mathcal{E}\}$
26 **end**
27 $\hat{\mathcal{E}}_{pf} \leftarrow \hat{\mathcal{E}}_{pf} \cup \{e_{\{N_S-1,0\},\{N_S-1,1\}}^{\hat{t}} \mid e_{\{N_S-1,0\},\{N_S-1,1\}}^{\hat{t}} \in \mathcal{E}\}$;
28 $\hat{\mathcal{E}}_{pf} \leftarrow \hat{\mathcal{E}}_{pf} \cup \{e_{\{N_S-1,1\},\{N_S-1,2\}}^{\hat{t}} \mid e_{\{N_S-1,1\},\{N_S-1,2\}}^{\hat{t}} \in \mathcal{E}\}$;
29 **end**
30 **end**

$$\Psi_{i_s}^{\hat{t}} + f_{\lceil p/2 \rceil}^h \cdot \gamma_{i_s,p}^{i_d,p} + \mu_{i_s}^{\hat{t}} \cdot (1 - \delta_{i_d,N_S-1}) \leq \varepsilon_{i_s} \tag{3.21}$$

Please notice that, in case of memory links, since I assume no energy is required to store data, this energy condition is automatically verified. Instead, a processing edge is included in the subgraph only if the node has enough energy to process the service plus a margin, and I assume that processing on a node can happen only in the first cycle during which it has enough capacity and energy to accomplish it (Lines 16-18).

In particular, the availability of processing capacity is checked by means of the following expression:

$$\Lambda_{\{i_s,0\},\{i_s,1\}}^{\hat{t}} + f_0^h \leq C_{\{i_s,0\},\{i_s,1\}}^{\hat{t}} \tag{3.22}$$

while if the following expression is verified, there is enough energy to accomplish service processing:

$$\Psi_{i_s}^{\hat{t}} + f_0^h \cdot \gamma_{i_s,0}^{i_s,1} + \mu_{i_s}^{\hat{t}} \leq \varepsilon_{i_s} \tag{3.23}$$

Finally, a fictitious edge from a node in the processing state to the processed one is included whenever the node in the processing state has not an incoming processing edge (i.e., it is reachable only through a storage edge in the processing state), or, in case there is an incoming processing edge, when the node has enough energy to process the service and then transmit it in its processed form (i.e., with a reduced size), plus a margin (Lines 19-21), with this condition being checked by means of the following expression:

$$\Psi_{i_s}^{\hat{t}} + f_0^h \cdot \gamma_{i_s,0}^{i_s,1} + f_1^h \cdot \left(\max_{i_d \in [0, N_S - 1]} \gamma_{i_s,2}^{i_d,2} \right) + \mu_{i_s}^{\hat{t}} \leq \varepsilon_{i_s} \tag{3.24}$$

Note that the introduction of the in-processing layer and fictitious links with respect to the model introduced in Section 3.2 is necessary to make the algorithm consider that, if processing happens on a node in a certain time cycle, the node may run out of energy in that time cycle, thus, it cannot transmit the service after having processed it. Instead, the introduction of a margin in the calculation of energy availability is necessary because the proposed heuristic is greedy: it tries to optimize the on-ground energy consumption for the current task without taking into account that the current resource allocation has an impact on the resources available for the other tasks still to be handled, leading to a non-optimal solution or even to the impossibility to allocate resources for all the services. For this reason, while selecting a link in the subgraph, it is only added if the energy the current service requires to cross it is small enough to allow the satellite to transmit all its remaining tasks to the ground station. In particular, the margin calculation is strictly related to the buffer tracking the services still to be handled by the algorithm in a certain time period and having a specific satellite as source. This buffer can be imagined as a list, whose entry $\beta_{i_s}^{\hat{t}} \in \beta$, with $i_s \in [0, \ldots, N_S - 2]$, $\hat{t} \in [0, \ldots, T - 1]$, is a set containing services $f^h \in \Sigma$, with $h \in [0, \ldots, N_T - 1]$ generated by the i_s-th satellite and such that $f_t^h \leq \hat{t} \leq f_t^h + f_d^h$. Consequently, the energy margin to be left on the i_s-th satellite during the $\hat{t}$-th time cycle is given by the sum of the preprocessing size and postprocessing size of all services $\varphi \in \beta_{i_s}^{\hat{t}}$, multiplied by the maximum unit transmission energy cost, following the expression:

$$\mu_{i_s}^{\hat{t}} = \sum_{\varphi \in \beta_{i_s}^t} (\varphi_0 + \varphi_1) \cdot \max_{i_d \in [0, N_S - 1]} \gamma_{i_s,2}^{i_d,2} \tag{3.25}$$

It is important to underline that the proposed margin calculation is empirical, and its optimization could open interesting research prospective which are out of the scope of this chapter.

Finally, the complexity of Alg.4 can be easily computed by considering that there are three nested loops repeating at most for $N_S - 2$, T and 2, respectively, leading to a complexity given by $\mathcal{O}\left(N_S^2 T\right)$. Instead, in order to calculate the complexity of Alg.3, I recall that the most efficient complexity of the Dijkstra's algorithm is given

by $\mathcal{O}\left(|\mathcal{N}|\log_2\left(|\mathcal{N}|\right)+|\mathcal{E}|\right)$. In the presented scenario, the number of nodes is given by $3N_ST$, while in calculating the maximum number of edges, it is necessary to take into account that the maximum number of transmission links is reached when each couple of satellites can communicate during each time cycle (i.e., there are at most $N_ST(N_S-1)/2 \sim N_S^2T$ links), the maximum number of memory links is given by $N_S(T-1)$, the maximum number of processing links is given by N_S, while the maximum number of fictitious links is given by N_ST. It follows that $|\mathcal{E}| \sim N_S^2T$. Finally, the two loops in Lines 8-11 repeats for the maximum path length determined by Dijkstra's algorithm and the maximum deadline, which can be assumed to be equal to T. However, these are lower order contributes. For this reason, the complexity of Alg.3 can be easily calculated to be $\mathcal{O}\left(N_TN_ST\left(N_S+\log_2\left(N_ST\right)\right)\right)$, since for N_T times Alg.4 is applied, followed by a Dijkstra's algorithm execution.

3.3.3 Numerical results

In this section, I numerically evaluate how a strategy to jointly place processing and route information within a satellite constellation dedicated to Earth Observation can lower the energy consumption on ground station due to data processing. The proposed performance investigation is related to the evaluation of the Ground Station Energy Saving, defined as the percentage difference between the energy consumption on ground stations when all services are processed on ground and the energy consumption on ground stations when some services are processed on-board of satellites. The amount of services processed in orbit depends on the resource allocation and processing placement strategy applied. In particular, I first evaluate the energy saving obtained by means of an optimal strategy, and I compare it to the one obtained by leveraging the GSESH algorithm. In fact, it would be impractical to solve the optimization problem in a real orbital scenario, thus, I first validate the GSESH algorithm with respect to the optimal solution in a simpler, but not trivial, scenario, in such a way that the heuristic can be then leveraged to evaluate the effectiveness of the proposed solution in a real orbital case. In this latter case study, performance are compared to two benchmarks[11], the first providing for processing always happening on ground (Always Ground solution, AG), the second providing for processing to happen either on the source satellite or on ground (Always First or Ground solution, FoG). In particular, in the FoG solution all satellites have processing capacity on-board, but they can leverage it only to process images they acquired when flying over a region of interest. In other words, satellites cannot offload processing of images they acquired to another satellite, and if they have not enough resources to process data, they shall offload it to the ground station. In my solution, instead, a service processing can be done by any satellite of the constellation, even different from the one originating it, or by a ground station (i.e., if a satellite does not have enough resources to accomplish the task processing, it can offload it to another satellite having more resources, or, in the worst case, to the ground station).

The following parameters will be the same for all the analysis. The time cycle duration will be set to $\tau = 10$ min, as a compromise value between the granularity of the dynamic topology representation and the complexity of running the algorithm. In concordance with what happens on Sentinel-2, I consider all services having

preprocessing size[102]:

$$f_0^h = \bar{f}_0 = 160 \text{ Mb}, \ \forall h \in [0, \dots, N_T - 1] \tag{3.26}$$

while I set postprocessing size:

$$f_1^h = \bar{f}_1 = 16\text{Mb}, \ \forall h \in [0, \dots, N_T - 1] \tag{3.27}$$

The number of CPU cycles to process a service will be considered to be $n_{cyc} = 737.5$ cycles/bit, regardless of the service and processing node[11]. On ground stations, since I assume that datacenters have enough processing capacity to process EO data, I will consider unlimited processing capacity, thanks to an unlimited number of CPUs, each having a clock frequency $\nu_{GS} = 4.5$ GHz (a common value for CPUs operating in ground computers), while I first assume that only one CPU is available on board of satellites (due to limitations on available space on-board), each with a clock frequency:

$$\nu_{Sat} = \alpha \cdot \hat{\nu}_{Sat} \tag{3.28}$$

where:

$$\hat{\nu}_{Sat} = \bar{f}_0 \cdot n_{cyc}/\tau \tag{3.29}$$

stands for the frequency needed on board of satellites to process a task during a time cycle. Thus, by setting α, the maximum number of services which can be completely processed by a satellite in a time cycle is set. In particular, the α parameter influences both the processing capacity available on satellites:

$$\Gamma_{Sat} = \nu_{Sat}/n_{cyc} = \alpha \cdot \bar{f}_0/\tau \tag{3.30}$$

and the unit processing energy cost on satellites:

$$\gamma_p^i = k \cdot n_{cyc} \cdot \nu_i^2 = k \cdot n_{cyc}^3 \cdot (\bar{f}_0/\tau)^2 \cdot \alpha^2 \tag{3.31}$$

where k represents the processor effective capacitance coefficient, and it will be set equal to 10^{-27}[11]. Please notice that, given this energy consumption model, by increasing α, the number of services which can be processed in a time cycle linearly increases, but at the same time the energy consumption to process each service increases quadratically. While I assume that ground stations have unlimited available energy, satellites nodes have a limited amount of energy available on-board to be consumed for OEC-related operations (i.e., processing and transmission of data). In particular, I assume that satellites have an amount of energy:

$$\varepsilon_i = \varepsilon_{Sat} = \nabla \cdot \bar{P} \cdot \tau, \ \forall i \in [0, \dots, N_S - 2] \tag{3.32}$$

available at the beginning of each time cycle, whose value is dependent on the ∇ parameter indicating the percentage of the maximum power $\bar{P}$ available on board of a Sentinel-2 satellite[102]. This assumption is justified by the fact that the power available on board of satellites is always higher than the amount needed by the satellite (both for bus devices and payload instruments), then an amount of this extra power (which is always generated, regardless of its actual consumption) can be used for OEC operations. For example, in the case of Sentinel-2 satellites, at the

end of life each satellite has a power available equal to $\bar{P} = 1700$ W, but only 1250 W are actually consumed: thus, up to around 25% of available power can be still used to support OEC. Moving to transmission links, following [97], I will consider a transmission power $P_t = 10$ W, antenna gain $G = 27$ dBi, carrier frequency $\nu = 26$ GHz, system noise temperature $T_s = 290$ K. I also consider a data rate of $R_{ISL} = 500$ Mbps for ISLs and $R_{dl} = 520$ Mbps in case of downlink to the ground station, as in the case of Sentinel-2 satellites[102]. Finally, memory on each satellite is limited to $M_i = 100$ Gb, $\forall i \in [0, \ldots, N_S - 2]$ to support the worst case load of data in the service generation scenario described further on, while I will consider ground stations having unlimited storage capacity, since I assume to have datacenters with enough capacity to handle all the data amount produced by the constellation. Energy cost to store data in memory will be considered equal to zero.

Topology and services

I consider the same satellite constellation introduced in Subsection 2.5.2, made of 24 satellites distributed over 6 circular orbits with altitude 712.84 km and inclination 98.24 deg. I also consider the same three ground stations described in Subsection 2.5.2. The propagation of the position of satellites and ground stations is obtained by using a modified version of the Python tool presented in Subsection 2.5.2, allowing for the translation of the dynamical topology into the graph representation introduced in Section 3.2. In particular, a topology having $N_S = 25$ physical nodes (24 satellites and 1 virtual ground station) and $T = 287$ time cycles is obtained. However, although this complete graph is considered as real orbital scenario evaluated by means of the GSESH algorithm, because of the NP-completeness of the proposed optimization problem, its solution is restricted on a case providing for $T = 4$ time cycles.

Service generation is obtained by using the same Python tool presented in Subsection 2.5.2. In particular, each satellite generates a task for each second of flying over the region of interest. Again, while comparing the optimal solution to the GSESH algorithm one, I will consider that services are generated when satellites flies over Italy during four time cycles, since this leads to a total number of 724 services, thus, to a simplified, yet not trivial, scenario. Instead, in the real orbital scenario, I consider satellites to generate services while flying over Australia and Mexico during an entire repeat cycle, leading to a total of 111483 services.

Results

The first analysis I propose is a comparison between the energy saving on ground stations obtained by solving the optimization problem and by applying the GSESH algorithm. Results are shown in Fig.3.2 and 3.3 for $\nabla = 0.1\%$ (i.e., $\varepsilon_{Sat}/\tau = 1.7$ W, or, equivalently, $\varepsilon_{Sat} = 1020$ J) and $\nabla = 0.2\%$ (i.e., $\varepsilon_{Sat}/\tau = 3.4$ W, or, equivalently, $\varepsilon_{Sat} = 2040$ J), respectively, and $\alpha \in [0, \ldots, 16]$. The available energy values have been chosen to evaluate the behaviour of the optimal solution when a small amount of energy is available on-board with respect to the number of services to be processed. In this first analysis, the number of services is small with respect to a real orbital case, but it is high enough to consider a simple, yet not trivial scenario. It can be noticed that the energy saving curve obtained by applying the proposed heuristic

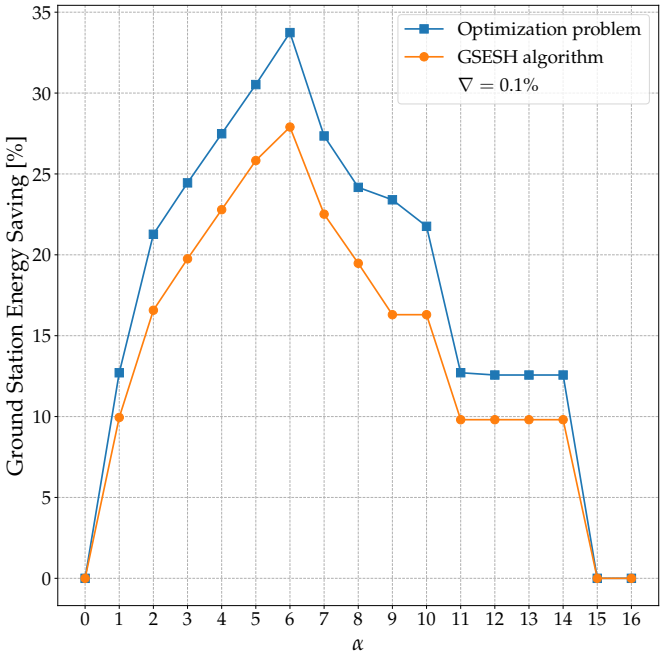

Figure 3.2. Ground Station Energy Saving obtained by solving the optimization problem and by applying the proposed GSESH algorithm, varying the number α of services processable on a satellite in a time cycle and fixing $\nabla = 0.1\%$ to limit the available energy for OEC operations to the 0.1% of the energy available on Sentinel-2 (1020 J).

follows the same trend of the optimal one, differing from the optimal values not more than 7.10% when $\nabla = 0.1\%$. However, this difference between the two curves even drop to zero with a small increase in the amount of energy available on board, i.e., when $\nabla = 0.2\%$. It is interesting to comment the particular behaviour both the curves follow. First, it can be noticed that there is an interval of α values where the energy saving percentage increases, until it reaches a maximum and then it starts decreasing, finally reaching the same value it would be obtained by processing all data on ground. This is due to the fact that, by increasing α, there is a linear increase in the number of tasks which can be processed, but since the on-board CPU clock frequency also increases linearly with α and the processing energy consumption increases quadratically with the clock frequency, the energy consumption for processing a task increases quadratically with α. For this reason, there is a first interval of α where the energy consumption on ground stations decreases because the number of services each satellite can process increases. Notice that the slope of the energy saving curves changes because of the energy margin limiting the number of services that can be handled by satellites. However, after a certain value of α (for example, in the case shown in Fig.3.2, after α =6) even though a satellite has an increased computational capacity, the energy required to process a task is such that the satellite has not enough energy to process all the α services it would be able to elaborate, thus, the number of services processed on board decreases because of energy constraints. For instance, in case shown in Fig.3.2, the same performance for $\alpha = 4$ and $\alpha = 7$ are obtained. In fact, even though

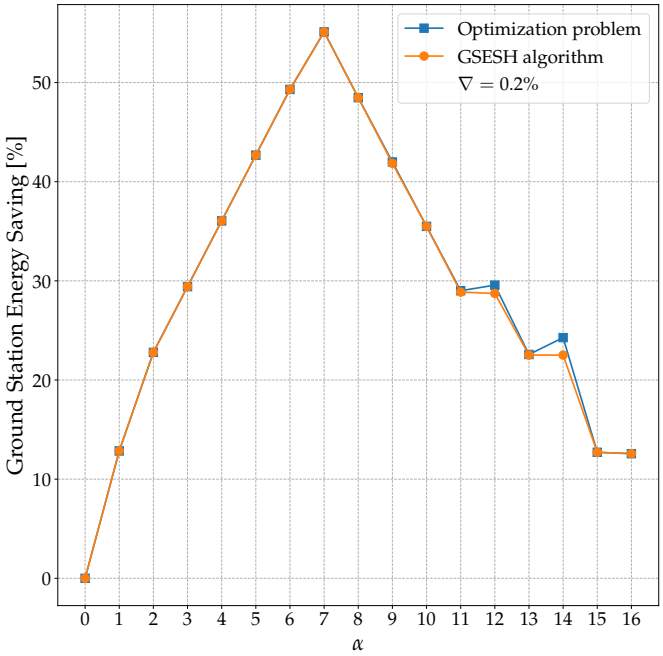

Figure 3.3. Ground Station Energy Saving obtained by solving the optimization problem and by applying the proposed GSESH algorithm, varying the number α of services processable on a satellite in a time cycle and fixing $\nabla = 0.2\%$ to limit the available energy for OEC operations to the 0.2% of the energy available on Sentinel-2 (2040 J).

when $\alpha = 7$ each satellite has enough computational capacity to process 7 services during a time cycle, due to the limited energy available on-board, it can only process around 4 services; thus, the same performance of a less powerful CPU having $\alpha = 4$ is obtained, since with $\alpha = 7$ there is a higher power consumption to elaborate every single task which leads to a higher consumption of the limited amount of energy available on board. Thus, due to constraints on the energy available on-board, after a certain value of α, the energy saving on ground stations decreases because a higher number of tasks are processed on ground, until the region of α values such that only one full service can be processed in space due to limited on-board energy, where the energy consumption on ground stations reaches a plateau. Finally, after a certain value of α (in the case of Fig.3.2, for $\alpha \geq 15$), the energy that would be consumed on-board to process even a single task is higher than the energy available on-board, thus, no service is processed on-board and tasks are all offloaded to the ground stations, thus, leading to no energy saving on ground.

After having validated the GSESH algorithm, I apply it to a complete orbital scenario to evaluate how the possibility to process data on board of satellites and to leverage the full constellation in processing, in such a way that if a satellite has not enough resource to process data can offload the elaboration to another satellite, allows to save energy on the ground station. Results reported in Fig.3.4 for $\alpha \in [0, \ldots, 50]$ and $\nabla = 1\%$ (i.e., $\varepsilon_{Sat}/\tau = 17$ W, or, equivalently, $\varepsilon_{Sat} = 10200$ J) show that when there is the possibility to process data on board (i.e., in case of GSESH Algorithm or FoG benchmark solution), energy is saved on ground stations,

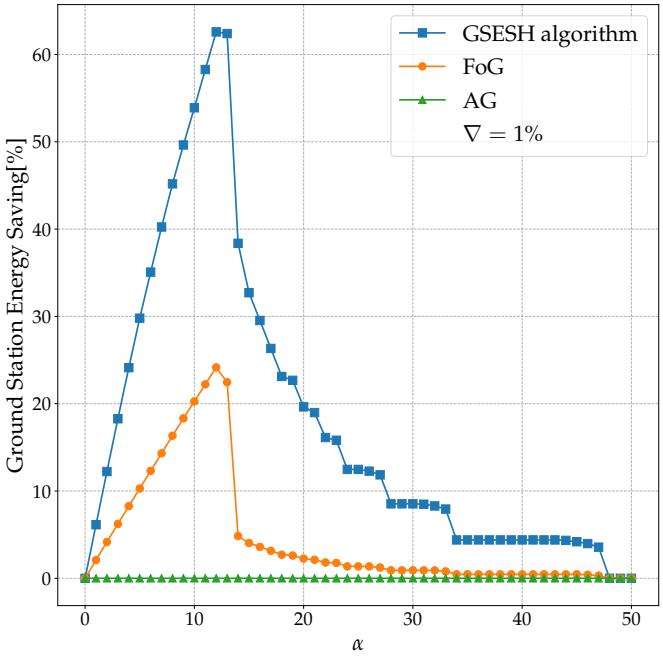

Figure 3.4. Ground Station Energy Saving obtained by applying the proposed GSESH algorithm and two benchmarks (FoG and AG), varying the number α of services processable on a satellite in a time cycle and fixing $\nabla = 1\%$ to limit the available energy for OEC operations to the 1% of the energy available on Sentinel-2 (10200 J).

and the ground station energy saving trend follows the same behaviour discussed before. In particular, the peak energy saving on ground station is reached for an increased value of α with respect to the case seen for the heuristic validation, since in this case there is an increased amount of energy available on board (in particular, $\nabla = 1\%$ with respect of $\nabla = 0.1\%$ of the previous analysis) to handle a higher number of services. Furthermore, in this complete case study orbital scenario it is possible to notice that in the region where ground station energy saving decreases because of limited on-board energy, there are some α value regions where the energy saving remains constant. This is because of the energy margin limiting the amount of data each satellite can handle. Finally, as expected energy saving on ground station is zero when the AG benchmark solution is applied.

I also propose an analysis aiming to study the impact of the amount of energy available on board of satellites on the percentage of ground station energy saving. In Fig.3.5, I show the energy consumed on ground stations by varying $\alpha \in [0, \ldots, 50]$ and $\nabla \in \{1\%, 5\%, 10\%, 15\%\}$, by applying the GSESH algorithm. It can be noticed that, by increasing the energy available on board, there is an extension of the region of α values where the ground station energy saving increases, and the peak ground station energy saving percentage also increases with ∇. In particular, it can be noticed that with only the 10% of energy available on Sentinel-2 to be dedicated to OEC operations, it is possible to obtain 99.95% of ground station energy saving when $\alpha = 27$, while with the 15% of energy available on Sentinel-2, the 100% ground station energy saving is reached when $\alpha = 28$. Please notice that drops in energy

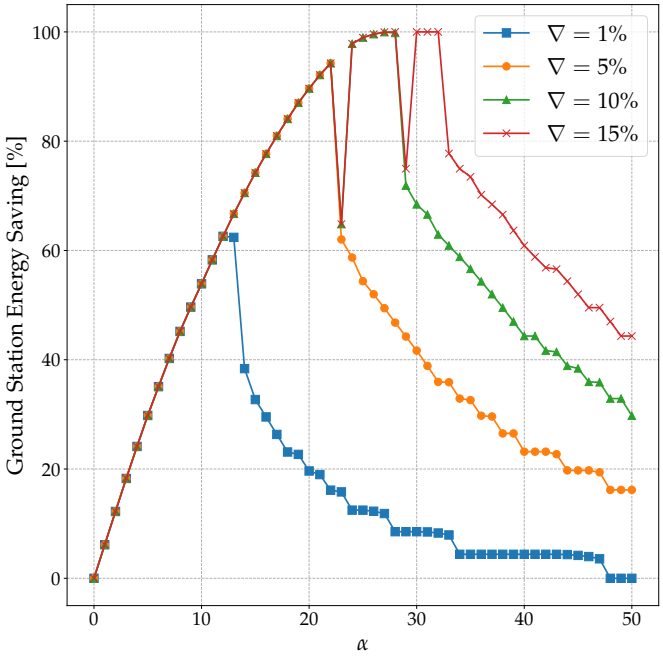

Figure 3.5. Ground Station Energy Saving obtained by applying the proposed GSESH algorithm by varying the number α of services processable on a satellite in a time cycle and the ∇ percentage of Sentinel-2 energy available for OEC operations.

saving which can be seen in case $\nabla = 10\%$ and $\nabla = 15\%$ are due to a particular interaction between the set of services and the empirically determined energy margin calculation, leading to a drop in services that can be managed by the satellites when $\alpha = 23$ and $\alpha = 29$ respectively.

Finally, I evaluate how the ground station energy saving is influenced by the number of cores available on board of satellites. In particular, by increasing the number of cores but leaving α fixed, there is a linear increase in the number of services which can be processed in a time cycle, and in this case the energy consumption increases linearly, too, since the CPU clock frequency of each single core remains the same, and consequently the energy demand of each core does not change. In particular, results reported in Fig.3.6, where I considered $\alpha \in [0, \ldots, 50]$ and $\nabla = 1\%$, show that by increasing the number of cores, the peak ground station energy saving increases, and the peak is also reached with a lower α value, i.e., with less powerful cores. This is due to the fact that, at the same α value, by increasing the number of cores, computational capacity still increases linearly, but the energy consumption also has a linear increase with the number of cores, since the α value remains the same. In other words, a dual-core architecture with each core having a CPU clock frequency such that $\alpha = 1$ would allow for the same computational capacity of a single-core architecture where the CPU clock frequency of the core is such that $\alpha = 2$, but the dual-core architecture would consume less energy with respect to the single-core solution, and this helps overcoming the limitation on energy available on board which would prevent the complete use of the available computational capacity. However, an increase in the number of cores would increase the on-board computer

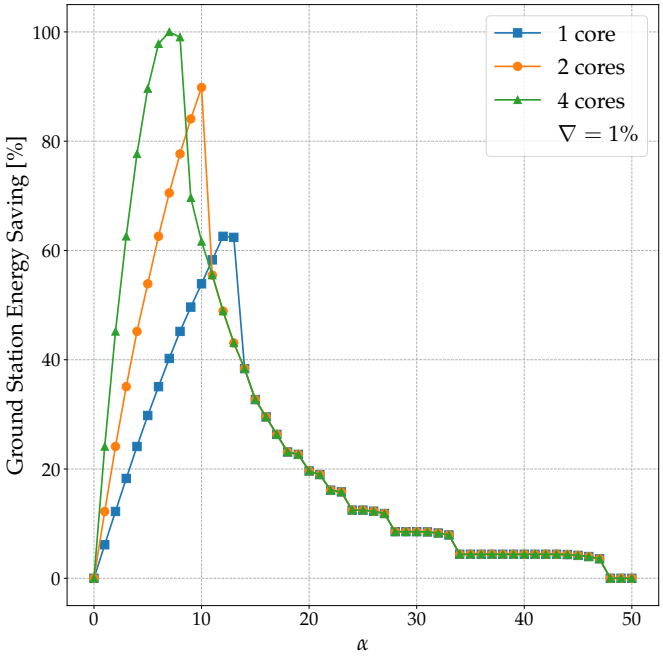

Figure 3.6. Ground Station Energy Saving obtained by applying the proposed GSESH Algorithm by varying the number α of services processable on a satellite in a time cycle and the number of cores available on board of satellites, with ∇ =1% of Sentinel-2 energy available for OEC.

complexity, and would require more space in the limited satellite volume. For this reason, I limited my analysis to a maximum of 4 cores.

3.4 Strategies with limitations on satellite battery Depth-of-Discharge

After having introduced strategies to maximize on-ground energy saving related to EO image processing without any optimization of the satellite operative life, in this section I propose an extension of the optimal strategy presented in Subsection 3.3.1 to jointly allocate resources and place processing to maximize the energy saving on ground stations while also minimizing the maximum on-board battery DoD to optimize satellite operative life. DoD optimization has been also proposed by authors of [12] as a solution to increase satellite operative life, but they did not take into account the joint maximization of ground station energy saving.

Differently from Section 3.3, I will consider that energy is provided by solar panels while a satellite is sunlit and by batteries during eclipse. The following problem will be defined on a network and service model similar to the one presented in Section 3.2. The only difference is in the fact that, for each i-th satellite, with $i \in [0, \ldots, N_S - 2]$, the amount of energy available from solar panels that can be used for OEC operations in a time cycle will be denoted by ε^i_{sp} (in J), while the amount of full-charge energy on batteries which can be used for OEC will be identified by ε^i_{batt} (in J). Instead, energy available on ground stations will be indicated with ε_{GS}

(in J). Finally, I introduce the parameter l_i^t, returning the value -1 if the i-th satellite is sunlit during the t-th time cycle, with $i \in [0, \dots, N_S - 2]$ and $t \in [0, \dots, T-1]$, or the time cycle at which eclipse started if the satellite is in eclipse.

3.4.1 Optimization Problem

Since I aim to optimize first ground station energy consumption and second battery DoD, the proposed problem is multi-objective. In this Subsection, I formulate and solve it by following the epsilon-constraint method[103], providing for formulating a single-objective problem maintaining only one of the original objectives as the new objective, and considering the remaining ones as constraints. This is possible, for example, when there exists an expected, or desired, optimal value for objectives to be considered as constraints. In this case, I will consider the energy consumption on ground stations as the objective of the proposed single-objective formulation, while I move the minimization of maximum DoD objective to the constraints, since the maximum DoD achievable can be seen as a project parameter. This allows to formulate the problem under discussion by adding the following constraint to the optimization problem introduced in Subsection 3.3.1:

$$\begin{aligned}\varepsilon_{i_s}^t = \delta_{i_s,N_S-1} \cdot \varepsilon_{GS} + (1 - \delta_{i_s,N_S-1}) \Bigg\{ \delta_{l_{i_s}^t,-1} \cdot \varepsilon_{sp}^{i_s} + \Big(1 - \delta_{l_{i_s}^t,-1}\Big) \Bigg[\varepsilon_{batt}^{i_s} + \\ - \Big(1 - \delta_{l_{i_s}^t,t}\Big) \sum_{h=0}^{N_T-1} \sum_{i_d=0}^{N_S-1} \sum_{p_s=0}^{1} \sum_{p_d=0}^{1} \sum_{\tilde{t}=l_{i_s}^t}^{t-1} y_{\{i_s,p_s\},\{i_d,p_d\}}^{\tilde{t},h} \; f_{p_s}^h \; \gamma_{i_s,p_s}^{i_d,p_d} \Bigg] \Bigg\}, \\ \forall i_s \in [0, \dots, N_S - 1], \; \forall t \in [0, \dots, T-1]\end{aligned} \tag{3.33}$$

Please notice that also in this case I leveraged the Kronecker's delta notation introduced in (2.17). Thanks to this constraint, it is possible to set the appropriate amount of available energy on each node (satellite or vGS) at the beginning of each time cycle. In particular, in case the energy on a satellite is being considered, if the satellite is sunlit during a certain time cycle, it has an energy amount equal to the extra energy provided by the solar panels during that time cycle (i.e., I assume that batteries are not used for OEC while sunlit) with respect to the energy request of both satellite payload and subsystems (i.e., energy that is generated but unused, and can be thus leveraged for OEC); instead, if the satellite is in eclipse and the eclipse takes several time cycles, I consider that the energy available during the first eclipse time cycle is equal to the maximum amount of energy stored in batteries that can be used for OEC, that can be expressed as function of the DoD parameter representing the maximum battery DoD related to OEC operations only. This also allows for the optimization of DoD (which influences the amount of energy from batteries that can be leveraged for OEC), following the epsilon-constraint method. Please notice that I consider batteries to be fully charged when eclipse begins. Instead, the energy available during following eclipse time cycles is equal to the energy available at the beginning of the eclipse, minus the energy allocated during the previous eclipse time cycles, in order to take into account the battery discharge.

Finally, it easily follows that also this new formulation is NP-complete, since it is analogous to the NP-complete problem stated in Subsection 3.3.1 with the addition of a further constraint.

3.4.2 Numerical Results

In order to numerically evaluate the optimization problem formulated in Subsection 3.4.1, I consider almost the same parameters introduced in Subsection 3.3.3. The only differences are the following:

- Given the NP-completeness of the proposed problem, in order to obtain significant results in a reasonable time, I consider a simple, yet not trivial, scenario. In particular, I focus on only two time cycles (i.e., $T = 2$). Furthermore, I consider that satellites generate images (i.e., services) during each second of flight over a region identified by a minimum latitude of 36.5° N, maximum latitude 40.1° N, minimum longitude 13.6° E, maximum longitude 16° E, for a total of 116 services, all generated during the first time cycle by sunlit satellites.
- In order to study the behaviour of the proposed strategy under energy limitations, given the small number of considered services, I set $\varepsilon^i_{sp} = \varepsilon_{sp}$ and $\varepsilon^i_{batt} = \varepsilon_{batt},\ \forall i \in [0, \ldots, N_S - 2]$, with $\varepsilon_{sp} = 600$ J and $\varepsilon_{batt} = 10 \cdot DoD \cdot \varepsilon_{sp}$. Thus, it is important to underline that, as it has been introduced in Subsection 3.4.1, the maximum amount of energy stored in batteries that can be used for OEC depends on the DoD parameter.
- I consider service delivery deadline to the ground as an analysis parameter, i.e., $f^h_d = \bar{f}_d,\ \forall h \in [0, \ldots, N_T - 1]$, with $\bar{f}_d \in \{0, 1\}$.

Results

In Fig. 3.7, I focus on the energy consumption on ground because of image processing, obtained by applying the optimal strategy for different maximum DoD values, considering service delivery to ground station deadline equal to zero time cycles. By increasing DoD, there is an increase in the energy available on satellites in eclipse, while maintaining the same energy on sunlit ones. It can be noticed that a higher DoD allows for reaching higher energy saving on ground when appropriate processing capacity is available on satellites. Furthermore, it can be interesting to note that, for $DoD > 10\%$, two maxima appear in the ground energy saving curve. This is due to the fact that for $DoD > 10\%$ the maximum energy available on board of satellites in eclipse during the entire eclipse time is higher than the amount available in a single time cycle on sunlit satellites. For this reason, as it can be seen in Fig. 3.8, where $DoD = 100\%$, after a certain α value ($\alpha = 5$ in the case shown), the number of services processed on sunlit satellites decreases because of the limited energy available from solar panels, while the number of services processed by satellites in eclipse continues to increase linearly with α because of the linear increase of computational capacity, which can be totally leveraged thanks to the higher energy provided by batteries with respect to solar panels. In particular, for $5 < \alpha < 8$ the increase in services processed by satellites in eclipse cannot compensate for the decrease in

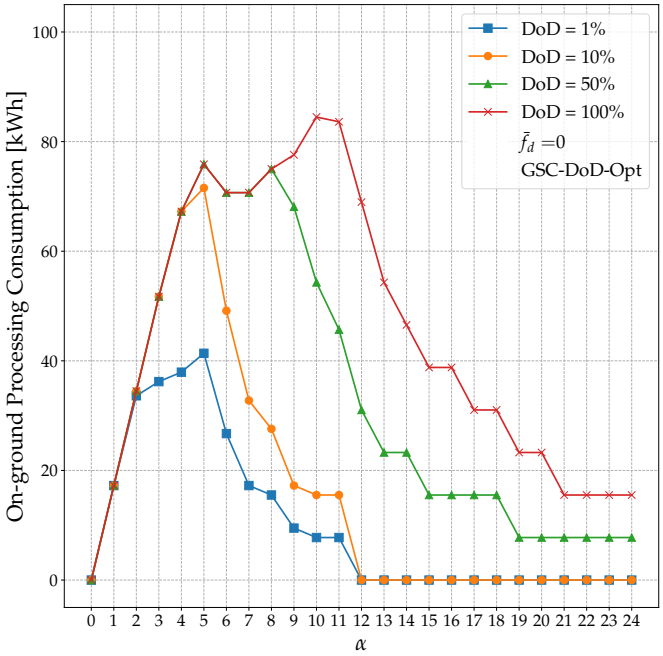

Figure 3.7. Energy saving on ground obtained by applying the optimal strategy for different DoD values, with service delivery to ground station deadline equal to zero time cycles.

images elaborated in sunlit satellites. For this reason, there is an overall decrease in services processed in-orbit, i.e., an increase in on-ground processing resulting into a decreased on-ground energy saving in this interval of α values. However, for $8 \leq \alpha \leq 10$, the increase in services processed by satellites in eclipse is higher than the decrease in number of services processed by sunlit satellites, thus, the number of images elaborated in-orbit starts increasing again, increasing the on-ground energy saving for data processing. This reaches a second maximum because of the fact that, after a certain α value, the on-board energy consumption for unit processing is such that also on satellites in eclipse it is no longer possible to process all services for which there would be processing capacity because of the limited energy available by batteries. Thus, there is a decrease in the number of services processed in eclipse, too, leading to an overall decrease in the number of services elaborated in-orbit which shall be thus processed on ground.

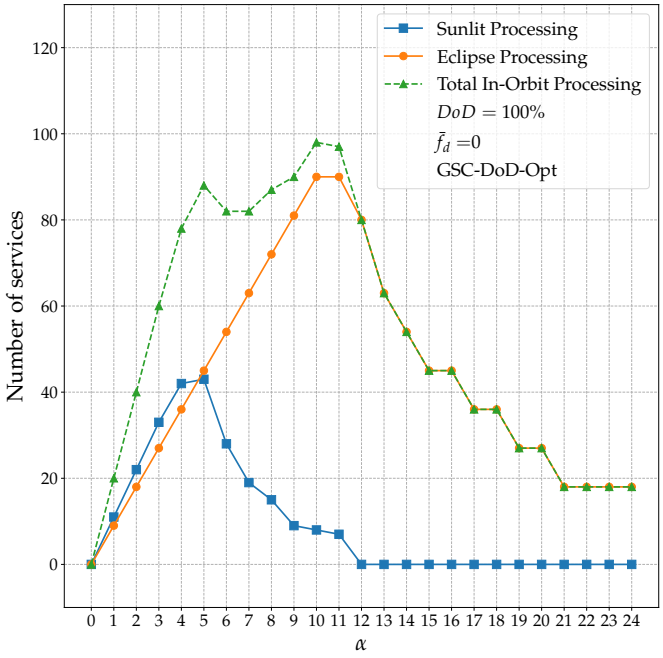

Figure 3.8. Number of services processed by sunlit satellites, by satellites in eclipse and total of services processed in orbit by applying the optimal strategy with $DoD = 100\%$ and $\bar{f}_d = 0$.

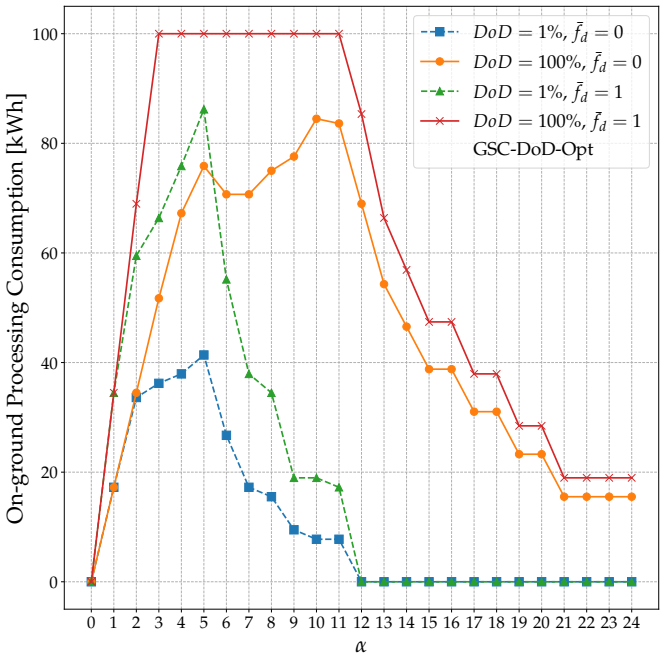

Figure 3.9. Energy saving on ground obtained by applying the optimal strategy for different DoD and $\bar{f}_d$ values.

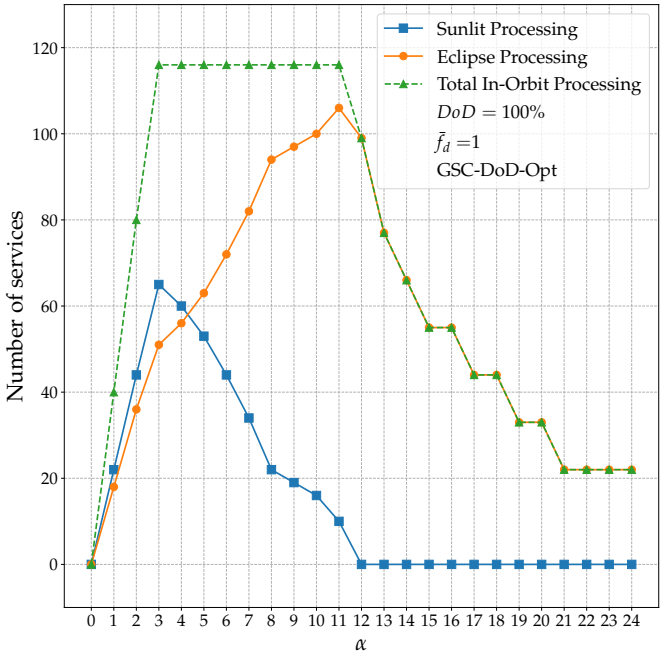

Figure 3.10. Number of services processed by sunlit satellites, by satellites in eclipse and total of services processed in orbit by applying the optimal strategy with $DoD = 100\%$ and $\bar{f}_d = 1$.

Finally, in Fig. 3.9 I report the energy saving on ground obtained by applying the proposed optimal strategy for different DoD and service delivery to ground station deadline values. In particular, by doubling the deadline (i.e., by setting $\bar{f}_d = 1$ instead of $\bar{f}_d = 0$), it is possible to obtain higher energy saving on ground, since the number of time cycles during which services can be processed in orbit is double, thus, the number of services that can be processed by satellites even with the same energy constraints (i.e., by fixing ε_{sp} and DoD) is potentially doubled. It can be interesting to further discuss what happens in case $DoD = 100\%$ and $\bar{f}_d = 1$, where the on-ground energy saving curve is constantly equal to 100% for $3 \leq \alpha \leq 11$. By looking at Fig. 3.10, it is possible to notice that, for these α values, even though the number of services processed by sunlit satellites decreases because of the energy limitation previously discussed, the higher energy provided by the battery and the availability of two time cycles to process data allow satellites in eclipse to execute all tasks that cannot be elaborated by sunlit satellites, thus, zeroing the energy consumption for processing on ground stations since all services are elaborated directly in-orbit.

3.5 Conclusions

In this chapter, I evaluated how the combined application of both satellite networks thanks to ISLs and edge computing capabilities on board of satellites can be fruitfully leveraged by EO satellite constellations to save energy on ground station by elaborating images directly on board of satellites instead of on ground. This allows

for better leveraging an amount of unused energy which is pre-allocated on board of satellites, instead of requiring extra energy on ground. First, I focused on only optimizing the on-ground energy saving, formalizing and solving an optimization problem which however resulted to be NP-complete. For this reason, in this context I also proposed the Ground Station Energy Saving Heuristic (GSESH) Algorithm to be applied to real orbital scenario. I showed that even with a small amount of energy available on board of satellites it is possible to obtain a substantial energy saving on ground stations. In particular, I have verified that the GSESH algorithm allows for an energy saving in the ground station up to 40% higher than the one achieved with the benchmark solution in a real scenario. However, although intuitively one might assume that the higher the computational capacity on board the satellites, the higher the ground station energy saving (since more data are processed in-orbit), I showed that the limitation on energy available on board of satellites is such that there is a CPU clock frequency value for which the maximum ground station energy saving is achieved, above which this saving starts decreasing because even though there is a higher computational capacity, the energy to be used to process every single task also increases and the small amount of energy available on board is such that there is not enough energy to leverage the full available computational capacity. This behaviour has been also confirmed by proposing and solving a more complex optimization problem, aiming at both maximizing on-ground energy saving and optimizing on-board battery DoD. In this case, obtained results show that the optimal value of the processing capacity to be installed on-board of satellites has to be chosen taking into account the amount of energy provided by solar panels while sunlit and by batteries during eclipse, as well as the maximum DoD reachable and the service delivery to the ground deadline time.

Chapter 4

Support to in-orbit Distributed Learning

One of the key enabling solutions to in-orbit extract information from Earth Observation images is given by deep learning techniques. However, the accuracy of these algorithms is strictly related to the availability of large datasets of satellite images for training purposes. Basic deep learning training schemes would provide for the transfer of all the acquired images to the training node (i.e., a ground station or a specific satellite), where model training is executed. However, this would require a high amount of bandwidth to transfer a high amount of data in a short visibility time. Instead, federated learning can be fruitfully leveraged in this scenario, since this technique provides for each satellite to train a local model only with its own dataset, and then to share its trained model with a central server, which receives models trained by the different satellites and aggregates them into a new global model which is finally shared with all the satellites, and this repeats until convergence is reached. Such strategy has been deeply investigated in literature[17]. In particular, in case of scarce communication opportunities with the central parameter server, asynchronous schemes[19] can be leveraged. However, in this case countermeasures to prevent model staleness negative impact on the convergence speed have to be taken into consideration[20]. Instead, in case of availability of intra-orbital ISLs, synchronous federated learning strategies can be leveraged[18]. However, it is important to underline that communication with a ground station (or, in general, with a node acting as a central parameter server) is limited by short visibility time. Consequently, local model gathering and consequent global model transmission to all satellites may need a long time because of limited communication windows, negatively impacting the time needed to reach model convergence, which is strictly related to the completion of these model sharing rounds. For this reason, I propose a communication strategy to support a completely distributed learning technique to train a deep learning model in-orbit, by leveraging the fact that satellites may form a network thanks to the potential availability of ISLs within and between orbital planes. My proposal is different from a federated learning approach since the presented strategy does not rely on a central parameter server, but each satellite receives all the information needed to calculate an updated global model by itself. Numerical results show that distributed learning outperforms federated learning in number of learning rounds

completed in the unit time by increasing the number of satellites, of orbital planes, of model parameters, of ground stations and by increasing the time needed to accomplish local learning. This behaviour allows for reaching validation accuracy convergence in a shorter time, as it has been verified on a land coverage classification task based on the EuroSAT dataset.

4.1 Network Modeling

Differently from models discussed in Sections 2.2 and 3.2, in this chapter I will not translate the orbital motion into a graph embedding the time-varying properties of the system. Instead, these properties will be evaluated in the continuous time domain. Parameters introduced in this section are summarized in Tab.4.1.

I consider to have a constellation of N_{Sat} satellites, equally distributed over N_{op} orbital planes. In this scenario, I will assume each orbit to be circular, with altitude h_p, inclination i_p and right ascension of the ascending node Ω_p, for $p \in [0, \ldots, N_{op}-1]$. I assume that the i-th satellite, with $i \in [0, \ldots, N_{Sat}-1]$, appertains to the orbital plane $p_i = \lfloor i \cdot N_{op}/N_{Sat} \rfloor$ and occupies the position $\mathbf{r}_i(t)$ at time t in the ECI reference frame. Satellite motion repeats with a period depending on the altitude h_p of the orbit it occupies given by:

$$T_p = 2\pi\sqrt{\frac{(R_E + h_p)^3}{\mu_E}},\ \forall p \in [0, \ldots, N_{op}-1]; \tag{4.1}$$

where R_E is the Earth's radius (assuming, without loss of generality, a perfectly spherical Earth) and μ_E is the Earth's gravitational constant.

The distance between the i-th and j-th satellite at time t, with $i, j \in [0, \ldots, N_{Sat}-1], i \neq j$, can be defined by means of the following expression:

$$d(i, j, t) = |\mathbf{r}_i(t) - \mathbf{r}_j(t)| \tag{4.2}$$

Similarly to Subsections 2.2 and 3.2, the distance between two satellites at time t allows to determine whether an ISL is available or not by verifying the following expression:

$$d(i, j, t) \leq d_{max}(i, j) \tag{4.3}$$

where $d_{max}(i, j)$ represents the maximum distance between the i-th and j-th satellite at which communication is possible. Again, under the assumption of Additive White Gaussian Noise channel, this can be calculated as:

$$d_{max}(i, j) = \frac{G\ c}{4\pi\nu_{tx}}\sqrt{\frac{P}{\left(2^{\frac{R_{i,j}}{B}} - 1\right) k_B T_s B}} \tag{4.4}$$

where G is the antenna gain, c is the speed of light, ν_{tx} is the carrier frequency, P is the transmission power, $R_{i,j}$ is the transmission data rate between the satellites, B is the bandwidth, k_B is the Boltzmann's constant, T_s is the system noise temperature. However, it can be useful to distinguish between two types of ISLs which can be potentially available: intra-orbital and inter-orbital. In particular, the former type

identifies ISLs between couples of satellites appertaining to the same orbital plane, while the latter identifies ISLs between satellites on different orbital planes. This distinction allows me to introduce a condition different from (4.3) to check for intra-orbital ISLs availability, independently from the time t. In fact, since all satellites on a same circular orbit move with the same angular velocity, their relative distance does not change in time. For this reason, intra-orbital ISLs are available if and only if the following two conditions are verified for any couple of adjacent satellites in the orbit:

$$\begin{gathered}\begin{cases}2\left(R_E+h_p\right)\sin\left(\frac{2\pi N_{op}}{N_{Sat}}\right)\leq d_{max}(i,j)\\ \left(R_E+h_p\right)\cos\left(\frac{2\pi N_{op}}{N_{Sat}}\right)>R_E\end{cases},\\ \forall i\in[0,\ldots,\frac{N_{Sat}}{N_{op}}-1],\ j=(i+1)\ mod\ \frac{N_{Sat}}{N_{op}}\end{gathered}\tag{4.5}$$

where the first condition ensures that the line of sight distance between two adjacent nodes is smaller than the maximum distance at which communication is possible, as defined in (4.4), while the second condition ensures that the line of sight does not intersect the Earth. If these are satisfied for any couple of satellites in the orbit, intra-orbital ISLs are always active, otherwise, they are always unavailable. Please notice that I am assuming intra-orbital ISLs to be potentially available only between a satellite and its adjacent nodes in the orbit.

As far as ground segment is concerned, I consider N_{GS} ground stations on the Earth enabled to receive data from the constellation. Each ground station moves with the Earth, thus, its position in time $\mathbf{r}_{GS_g}(t)$, with $g\in[0,\ldots,N_{GS}]$, changes periodically with a period equal to the sidereal day, $T_{GS}=86164$ s. Thus, the entire system constituted by all satellites and ground stations is periodic, with a period $T=lcm\left(T_{GS},T_0,\ldots,T_{N_{op}-1}\right)$. I assume that communications between the i-th satellite and the g-th ground station are possible when the elevation angle of the satellite with respect to the ground station (i.e., the angle between the tangent plane to the Earth surface containing the ground station and the vector Earth center-satellite vector) is higher than a minimum elevation El_{min}. This translates in the following condition:

$$\frac{\pi}{2}-\arccos\left(\frac{\mathbf{r}_{GS_g}(t)\cdot\mathbf{r}_i(t)}{|\mathbf{r}_{GS_g}(t)||\mathbf{r}_i(t)|}\right)\geq El_{min},\ \text{for}\ i\in[0,\ldots,N_{Sat}-1],\ g\in[0,\ldots,N_{GS}-1]\tag{4.6}$$

where $\mathbf{r}_{GS_g}(t)\cdot\mathbf{r}_i(t)$ represents the scalar product between the position vector of the ground station and of the satellite, respectively, and El_{min} is in radiant. It is important to underline that, in case a satellite is able to communicate with more than a ground station at a time, I assume that it is connected with only one of them.

Table 4.1. Network model parameters

Set or parameter	Description
N_{sat}	number of satellites
N_{op}	number of orbital planes
h_p	orbit altitude
i_p	orbit inclination
Ω_p	orbit right ascension of the ascending node
i	index in $[0, \ldots, N_{sat} - 1]$ representing a satellite
p_i	orbital plane occupied by the i-th satellite, with $i \in [0, \ldots, N_{sat} - 1]$
$\mathbf{r}_i(t)$	position vector of the i-th satellite, with $i \in [0, \ldots, N_{sat} - 1]$
h_p	altitude of satellites on the p-th orbital plane, with $p \in [0, \ldots, N_{op} - 1]$
T_p	period of motion of satellites on the p-th orbital plane, with $p \in [0, \ldots, N_{op} - 1]$
R_E	Earth's radius
μ_E	Earth's gravitational constant
G	antenna gain
c	speed of light
ν_{tx}	carrier frequency
P	transmission power
$R_{i,j}$	transmission data rate between i-th and j-th satellite, with $i, j \in [0, \ldots, N_{sat} - 1],\ i \neq j$
B	transmission bandwidth
k_B	Boltzmann's constant
T_s	system noise temperature
N_{GS}	number of ground stations
g	index in $[0, \ldots, N_{GS}]$ representing a ground station
$\mathbf{r}_{GS_g}(t)$	position vector of the g-th ground station, with $g \in [0, \ldots, N_{GS} - 1]$
T_{GS}	ground station rotation period, i.e., Earth's sidereal day
T	repeat cycle
El_{min}	minimum elevation angle

4.2 Distributed Learning Communication Strategy

In this section, I introduce my distributed learning-based proposal by first describing how each satellite can autonomously calculate the updated global model and locally train its own local model in Subsection 4.2.1. Finally, the communication strategy supporting the presented distributed learning approach is discussed in Subsection 4.2.2. Parameters introduced in this section are summarized in Tab.4.2.

4.2.1 Local Training Phase

I propose a purely distributed learning solution where satellites exchange their own models among themselves, and each satellite calculates the updated global

Table 4.2. Application model parameters

Set or parameter	Description
N_r	number of distributed learning rounds
N_{ep}	number of local learning epochs
D_i	dataset of the i-th satellite, with $i \in [0, \ldots, N_{sat} - 1]$
$\|D_i\|$	number of samples in the dataset of the i-th satellite, with $i \in [0, \ldots, N_{sat} - 1]$
$\mathbf{w}_i^{r,e}$	local model on the i-th satellite, at the e-th local training epoch of r-th distributed learning round, with $i \in [0, \ldots, N_{sat} - 1]$, $r \in [0, \ldots, N_r]$, $e \in [0, \ldots, N_{ep}]$
$\mathbf{w}_G^r$	global model at the beginning of r-th distributed learning round, with $r \in [0, \ldots, N_r]$
F_i	local loss function on the i-th satellite, with $i \in [0, \ldots, N_{sat} - 1]$

model by its own, once it has received local models from all the other satellites. This approach shares the way local models are aggregated in a global one with the synchronous FL scheme proposed in [18], in turn based on Federated Average (FedAvg) algorithm[104], but my proposal differs in the fact that in this strategy there is no central aggregation node, while each satellite is able to calculate the updated global model by its own. In particular, I suppose that learning happens in rounds. During each round, first satellites exchange their locally trained models in such a way each satellite has the models of all the other satellites. Then, each satellite computes the global model which will be the starting point of the training phase on a local dataset. After each satellite has finished to locally train the model, a new round begins with the sharing of the locally trained models. This procedure goes on until convergence is reached, or after a maximum number N_r of rounds.

Thus, I assume that each i-th satellite, with $i \in [0, \ldots, N_{Sat} - 1]$, has its own dataset D_i, containing a number of samples $|D_i|$, and a local model represented by the vector $\mathbf{w}_i^{r,e}$, containing the local values for weights and biases at the e-th learning epoch of the r-th distributed learning round, with $r \in [0, \ldots, N_r]$, $e \in [0, \ldots, N_{ep}]$ and N_{ep} representing the maximum number of local learning epochs, i.e., the maximum number of times the local model goes through updates over the local dataset during the round. As previously stated, each r-th round, with $r \in [1, \ldots, N_r]$, starts with each i-th satellite sharing its most updated version of locally trained model, i.e., the local model after N_{ep} epochs at the end of the previous round, denoted with $\mathbf{w}_i^{r-1,N_{ep}}$, with all the other satellites. I assume that $\mathbf{w}_i^{0,N_{ep}}$ is the local model on the i-th satellite before any distributed learning round occurs. Once each satellite has received the local models of all the remaining ones, the global model can be locally obtained as:

$$\mathbf{w}_G^r = \sum_{i=0}^{N_{sat}-1} \frac{|D_i|}{\sum_{i=0}^{N_{sat}-1} |D_i|} \mathbf{w}_i^{r-1,N_{ep}} \tag{4.7}$$

and the local model on the i-th satellite during the r-th round before any local learning epoch (i.e., $e = 0$) is set as:

$$\mathbf{w}_i^{r,0} = \mathbf{w}_G^r, \ \forall i \in [0, \ldots, N_{sat} - 1] \tag{4.8}$$

Finally, each i-th satellite trains its local model by applying the gradient descent technique considering only the local dataset for N_{ep} epochs. After local training, the local model will be:

$$\mathbf{w}_i^{r,N_{ep}} = \mathbf{w}_i^{r,0} - \eta \sum_{k=0}^{N_{ep}-1} \nabla F_i \left(\mathbf{w}_i^{r,k} \right), \ \forall i \in [0, \ldots, N_{sat} - 1] \tag{4.9}$$

with η representing the learning rate, and ∇F_i the gradient operator applied on the function F_i representing a local loss function, i.e., a loss function evaluated only on the samples of the i-th satellite dataset.

4.2.2 Model weight distribution phase

Let me discuss in more detail how the distributed learning strategy introduced in Subsection 4.2.1 can be integrated into the orbital environment by means of an *ad-hoc* communication solution. The strategy proposed in this chapter is formalized in Alg.5. The inputs needed are the number of satellites N_{sat}, maximum number of distributed learning rounds N_r, maximum number of local gradient descent epochs N_{ep}, duration of local learning phase τ_l, model size w_s (in Mb, please notice that I assume that the number of weights and biases is the same on each satellite, only their values changes, thus, the model size is the same on each node), data rate between each couple of i-th and j-th nodes $R_{i,j}$, with $i, j \in [0, \ldots, N_{sat}]$ and event list E. In particular, E is an ordered list where events are ordered by their occurrence time, with the earliest event occupying the first position, i.e., being the $E[0]$ element of the list. Each event $\varepsilon \in E$ is characterized by an event type denoted by ε.type, whose value is *'link-on'* if the event represents the fact that two nodes reached a relative position such that they are close enough to allow data transmission, it is equal to *'link-off'* if the event represents two nodes reaching a relative position such that their distance is not enough to enable communication, or it is equal to *'transfer_completed'* to indicate that data transmission between two nodes is completed. Furthermore, each ε event is also associated with a list of involved nodes $\varepsilon.N$ and a time at which the event happens $\varepsilon.t$. Finally, in case the event is *'link-on'* or *'link-off'*, the event is also associated to a property $\varepsilon.LOT$ indicating the time at which the link will be unavailable in case the event type is *'link-on'*, or the time at which the link became available in case the event type is *'link-off'*; instead, in case the event type is *'transfer_completed'*, the event is associated to a property $\varepsilon.W$ representing the models memorized on both the nodes involved in the data transfer after the transmission ends. At the beginning, the event list contains only *'link-on'* and *'link-off'* events that can be obtained by orbital mechanics, propagating node positions in time and evaluating their distances as discussed in Section 4.1. In particular, I evaluate these events only during a repeat cycle T, since because of the periodicity of both satellite and Earth motion, what happens in a repeat cycle will be repeated the same in the following ones.

Algorithm 5: Distributed Learning Strategy

Input: E, N_{sat}, N_r, N_{ep}, τ_l, $R_{i,j}$ $\forall i, j \in [0, \ldots, N_{sat} - 1]$

Initialize: $r \leftarrow 1$, $k \leftarrow False$, $\Lambda_{i,j} \leftarrow 0$, $M_i \leftarrow \{\mathbf{w}_i^{0,N_{ep}}\}$, $\forall i, j \in [0, \ldots, N_S - 1]$;
while $E \neq \emptyset \wedge r \neq N_r$ **do**
  $\varepsilon \leftarrow E[0]$ *//Extract earliest event*;
  $E \leftarrow E - \{\varepsilon\}$ *//Remove the extracted event from the event list*;
  $t \leftarrow \varepsilon.t$ *//Extract the event time*;
  $i \leftarrow \varepsilon.N[0]$, $j \leftarrow \varepsilon.N[1]$ *//Extract linked nodes*;
  if *ε.type is 'link-on'* **then**
    $\Lambda_{i,j} \leftarrow \varepsilon.LOT$ *//Set the link-off time*;
    Add Transfer Complete Event(E, i, j, M_i, M_j, $R_{i,j}$, $\Lambda_{i,j}$) *//Apply Alg.6*;
  else if *ε.type is 'transfer_completed'* **then**
    $M_i \leftarrow M_i \cup \{\varepsilon.W\}$, $M_j \leftarrow M_j \cup \{\varepsilon.W\}$ *//Update the list of received models on both the sharing nodes*;
    for $n \in [0, \ldots, N_{sat} - 1]$ **do**
      if $|M_n| == N_{sat}$ **then**
        $k \leftarrow True$ *//At least satellite has received all local models*;
      else
        $k \leftarrow False$ *//At least a satellite has not received all local models, yet*;
        break;
      end
    end
    for $n \in \{i, j\}$ **do**
      for $m \in [0, \ldots, N_{sat} - 1]$ **do**
        Add Transfer Complete Event(E, i, j, M_n, M_m, $R_{n,m}$, $\Lambda_{n,m}$) *//Apply Alg.6*;
      end
    end
  else if *ε.type is 'link-off'* **then**
    $\Lambda_{i,j} \leftarrow 0$ *//Set the link as unavailable*;
  end
  //If all satellites have received local models from the remaining ones, start local learning phase;
  if k **then**
    Update global model $\mathbf{w}_{GS}^r$ by means of eq.4.7;
    Apply N_{ep} epochs of gradient descent on each satellite in a time τ_l, until obtaining $\mathbf{w}_i^{r,N_{ep}}, \forall i \in [0, \ldots, N_{sat} - 1]$ by means of eq.4.9;
    //After local learning phase ends, a new round can start;
    for $i \in [0, \ldots, N_{sat} - 1]$ **do**
      $M_i \leftarrow \{\mathbf{w}_i^{r,N_{ep}}\}$;
      for $j \in [0, \ldots, N_{sat} - 1]$ **do**
        $\Lambda_{i,j} \leftarrow 0$;
      end
    end
    Update Event List(E, $t + \tau_l$) *//Apply Alg.7*;
    $r \leftarrow r + 1$;
  end
end

Output: r

Moving to the discussion of the different algorithm steps described in Alg.5, in Line 1 I initialize:

- the current round number r;
- a boolean auxiliary variable k indicating whether the local model distribution phase is completed or not;
- an auxiliary variable $\Lambda_{i,j}$ whose value is zero if at the current event time no link is available between the i-th and j-th node, with $i, j \in [0, \dots, N_{sat} - 1]$, and is different from zero when the link is available, with the specific value indicating the time at which link will become unavailable;
- an auxiliary list M_i for each i-th node, with $i \in [0, \dots, N_{sat} - 1]$, containing local models currently kept in memory on the i-th node.

In particular, before the algorithm starts, I assume no link is already available, since no link-on event has been considered, yet, and I assume each node has in its memory its own local model only, since no data transfer happened, yet. After auxiliary variable initialization, next steps will be repeated until event list is not empty and the maximum round number is not reached. In particular, first the earliest event in time ε is extracted and removed from the event list (Lines 3-4). Then, auxiliary variables t, i, j are set to be equal to the event time, and to the first and second nodes involved in the event, respectively (Lines 5-6). Following steps depend on the event type. In case of *'link-on'* event, since a new link is available, $\Lambda_{i,j}$ is set to be equal to the link-off time (Line 8). Furthermore, I assume that as soon as a link becomes available, the two connected nodes try to share the models in their memories following Alg.6, which will be discussed in detail further on. In case model transmissions are possible, a transfer completed event is added to the event list (Line 9). Instead, in case of *'transfer_completed'* event, first the lists of received models on both nodes involved in the exchange is updated (Line 11) and then memories on each satellite in the constellation are inspected to check whether each of them stores the local models of all the other satellites or not (Lines (12-19). In case all satellites have all the local models, auxiliary variable k is set to *True*. Furthermore, I assume that as soon as a node has received local models, it tries to share them with other nodes, again leveraging Alg.6 (Lines 20-24). Finally, if local model distribution phase is completed, global model can be updated (Line 30) and local learning phase can begin (Line 31). After its end, a new round can begin. For this reason, $\Lambda_{i,j}$ is reinitialized to be zero for each couple of nodes, and memory M_i on each i-th node only contains the local model obtained after N_{ep} of local gradient descent epochs (Lines 33-38). However, before a new round can start, event list shall be updated to contain only events happening after the local training phase is ended, following Alg.7 which will be discussed further on (Line 39).

Algorithm 6: Add Transfer Complete Event

Input: E, w_s, i, j, t, M_i, M_j, $R_{i,j}$, $\Lambda_{i,j}$

1 $n_w \leftarrow |M_i| + |M_j| - 2|M_i \cap M_j|$ *//Number of local models to be exchanged*;
2 $\tau_t \leftarrow w_s \cdot n_w / R_{i,j}$ *//Calculate the transmission time*;
3 $\tau_p \leftarrow 2 \cdot d(i,j,t)/c$ *//Calculate the propagation time*;
4 **if** $n_w \neq 0 \ \wedge t + \tau_t + \tau_p \leq \Lambda_{i,j}$ **then**
5 $\quad \tilde{\varepsilon}$.type $\leftarrow$ 'transfer_completed' *//Set new event type as transfer completed*;
6 $\quad \tilde{\varepsilon}.t \leftarrow t + \tau_t + \tau_p$ *//Time of the new transfer completed event*;
7 $\quad \tilde{\varepsilon}.N \leftarrow [i,j]$ *//Nodes involved in the new transfer completed event*;
8 $\quad \tilde{\varepsilon}.W \leftarrow M_i \cup M_j$ *//Weights on each involved node after new transfer completed event*;
9 $\quad E \leftarrow E \cup \{\tilde{\varepsilon}\}$ *//Add the new transfer completed event in the event list*;
10 **end**

Output: E

Let me now discuss steps of Alg.6. As introduced before, it evaluates if it is possible to complete a data transfer between two nodes and, in positive case, it adds a transfer completed event in the event list. In particular, I consider that in the communication between the i-th and j-th node, i sends to j all models that are in M_i but not in M_j, and vice versa. For this reason, first the number of models to be exchanged is evaluated by means of the expression in Line 1, as well as the transmission and propagation delay (Lines 2-3). Then, if the number of models to be exchanged is different from zero and the link is still available when the potential transmission is completed (Line 4), a *'transfer_completed'* event is added to the event list, having the time at which transfer finishes as event occurrence time, i and j as involved nodes, and the union of M_i and M_j as the models memorized on both i and j after transfer is completed (Lines 5-8). Finally, the new event is added to the event list (Line 9).

As far as Alg.7 is concerned, it allows for updating the event list to include only events happening after the end of local model training phase happening at time $\tilde{t}$, corresponding to the beginning of a new round. In particular, a new empty event list is initialized (Line 1). To update the event list, the following steps are applied for each event in the event list. First, the earliest event in time is extracted and removed from the event list, then t is set to be equal to the occurrence time of the extracted event (Lines 3-5). If the event occurs before the beginning of a new round (Line 6), it can be included in the updated list only if its type is *'link-on'* and if the link remains active after the beginning of the new round (Lines 7-8): in this case, a new *'link-on'* event is added to the updated event list, having occurrence time equal to the beginning time of the new round, and same nodes and link-off time of the extracted event. Instead, in case the occurrence time of the extracted event is higher than the time at which the new round begins, the extracted event can be included in the new event list if it is a *'link-on'* or *'link-off'* event. In particular, in case it is a *'link-on'* event, it is included in the new event list as it is, while in case it is a *'link-off'* event, an event having the same event type, occurrence time and nodes (Lines 20-22) is added in case the occurrence time is different from the new round beginning time. However, the new event will have an associated link-on time given by the maximum between the link-on time of the extracted event and the beginning

of the new round (Line 23). This is due to the fact that the associated *link-on* event may occur before the beginning of the new round, but as previously discussed, if a link becomes available before the new round and it is still available when the round begins, the associated *'link-on'* event will have a occurrence time equal to the new round begin time. Furthermore, if the link-on event associated to the link-off one just added to the new event list is not in $\tilde{E}$, it is added to the list (Lines 26-32). Finally, it can be noticed that no *'transfer_completed'* event appertaining to the previous event list is included in the new one, since transfers in the previous list are referred to the previous round.

Algorithm 7: Update Event List

Input: E, $\tilde{t}$
1 Initialize: $\tilde{E} \leftarrow \emptyset$;
2 **while** $E \neq \emptyset$ **do**
3 $\quad \varepsilon \leftarrow E[0]$ *//Extract earliest event*;
4 $\quad E \leftarrow E - \{\varepsilon\}$ *//Remove the extracted event from the event list*;
5 $\quad t \leftarrow \varepsilon.t$ *//Extract the event time*;
6 $\quad$ **if** $t < \tilde{t}$ **then**
7 $\quad\quad$ **if** $\varepsilon.type$ *is 'link-on'* **then**
8 $\quad\quad\quad$ **if** $\varepsilon.LOT > \tilde{t}$ **then**
9 $\quad\quad\quad\quad \tilde{\varepsilon}$.type $\leftarrow$ 'link-on' *//Set event type as link-on*;
10 $\quad\quad\quad\quad \tilde{\varepsilon}.t \leftarrow \tilde{t}$ *//Set event time as the beginning of new round*;
11 $\quad\quad\quad\quad \tilde{\varepsilon}.N \leftarrow \varepsilon.N$ *//Nodes involved remains the same*;
12 $\quad\quad\quad\quad \tilde{\varepsilon}.LOT \leftarrow \varepsilon.LOT$ *//Link-off time remains the same*;
13 $\quad\quad\quad\quad \tilde{E} \leftarrow \tilde{E} \cup \{\tilde{\varepsilon}\}$ *//Add the modified event in the event list*;
14 $\quad\quad\quad$ **end**
15 $\quad$ **else**
16 $\quad\quad$ **if** $\varepsilon.type$ *is 'link-on'* **then**
17 $\quad\quad\quad \tilde{E} \leftarrow E \cup \{\varepsilon\}$ *//Add the current event in the event list*;
18 $\quad\quad$ **else if** $\varepsilon.type$ *is 'link-off'* **then**
19 $\quad\quad\quad$ **if** $t \neq \tilde{t}$ **then**
20 $\quad\quad\quad\quad \tilde{\varepsilon}$.type $\leftarrow$ 'link-off' *//Event type remains link-off*;
21 $\quad\quad\quad\quad \tilde{\varepsilon}.t \leftarrow \varepsilon.t$ *//Event time remains the same*;
22 $\quad\quad\quad\quad \tilde{\varepsilon}.N \leftarrow \varepsilon.N$ *//Nodes involved remains the same*;
23 $\quad\quad\quad\quad \tilde{\varepsilon}.LOT \leftarrow \max\{\tilde{t}, \varepsilon.LOT\}$ *//Link-on time is the maximum between current event link-on time and new round start time*;
24 $\quad\quad\quad\quad \tilde{E} \leftarrow \tilde{E} \cup \{\tilde{\varepsilon}\}$ *//Add the modified event in the event list*;
25 $\quad\quad\quad\quad$ *//Find the associated link-on event*;
26 $\quad\quad\quad\quad \hat{\varepsilon}$.type $\leftarrow$ 'link-on';
27 $\quad\quad\quad\quad \hat{\varepsilon}.t \leftarrow \tilde{\varepsilon}.LOT$;
28 $\quad\quad\quad\quad \hat{\varepsilon}.N \leftarrow \tilde{\varepsilon}.N$;
29 $\quad\quad\quad\quad \hat{\varepsilon}.LOT \leftarrow \tilde{\varepsilon}.t$;
30 $\quad\quad\quad\quad$ **if** $\hat{\varepsilon} \notin \tilde{E}$ **then**
31 $\quad\quad\quad\quad\quad \tilde{E} \leftarrow \tilde{E} \cup \{\hat{\varepsilon}\}$ *//Add the associate link-on event in the event list*;
32 $\quad\quad\quad\quad$ **end**
33 $\quad\quad\quad$ **end**
34 $\quad\quad$ **end**
35 $\quad$ **end**
36 **end**
Output: $\tilde{E}$

As far as the complexity of the proposed strategy is concerned, it is possible to notice that it is mainly related to the length of the event list, to the maximum number of rounds and to the complexity of adding a *'transfer_completed'* event to the list. In particular, by implementing the event list as an heap queue, adding an element to the queue has a complexity $\mathcal{O}(\log n)$, where n is the length of the event list. By looking at the loop starting in Line 2, it can be noticed that an upper bound to the number of repetitions of this loop is given by the completion of N_r distributed learning rounds. In particular, in order to complete a round it is necessary that each satellite has received local models from all the other satellites. In the worst case, this is accomplished by means of direct communications between each couple of satellites, leading to $N_{sat}(N_{sat}-1)/2$ *'transfer_completed'* events for each round. For this reason, the complexity of the strategy can be given by $\mathcal{O}\left(N_r N_{sat}^2 \log n\right)$. However, in the proposed strategy the event list length may increase during each cycle of the loop, due to the inclusion of new *transfer_completed* events. An upper bound to the event list length can be estimated by considering that, in the worst case of direct communications between each couple of satellites, the list shall include all the *transfer_completed* events necessary to accomplish N_r distributed learning rounds discussed before; furthermore, since local model transfer between each couple of satellites can be completed only if there is an active link between them, a number of $N_{sat}(N_{sat}-1)/2$ *'link-on'* (and, potentially, *'link-off'*) events shall be also included in the list. For this reason, an upper bound to the event list length to guarantee N_r distributed learning rounds can be given by $N_r(N_{sat}(N_{sat}-1)/2)^2$. Consequently, the complexity of the proposed strategy can be expressed by $\mathcal{O}\left(N_r N_{sat}^2 \log\left(N_r N_{sat}^4\right)\right)$.

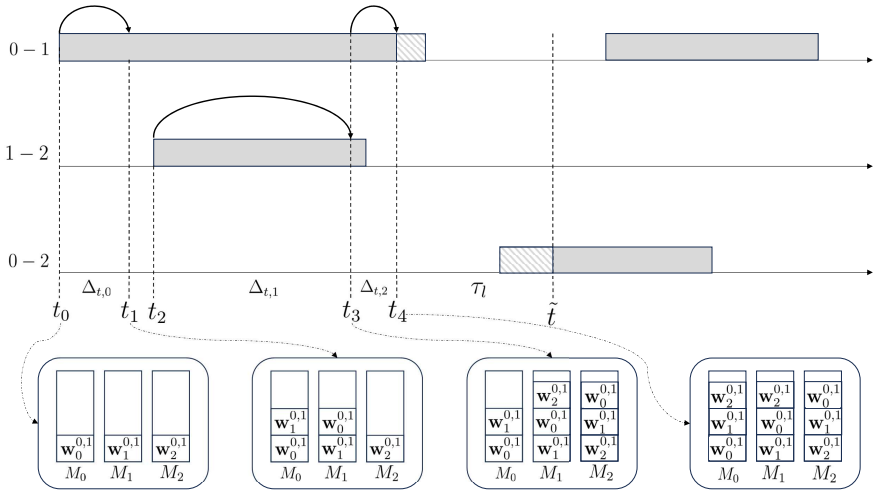

Figure 4.1. Example of the proposed strategy with 3 satellites. Grey horizontal bars indicate the availability of a link between a couple of satellites in time, thus, their limits represent link-on and link-off events. Transfer completed events are represented by curved arrow ends. Dashed bars indicates events that are not included in the new round event list. At the bottom, memories of nodes at t_0, t_1, t_3 and t_4 are shown.

In order to better clarify the presented communication strategy, let me introduce the example shown in Fig.4.1. In particular, I consider a constellation of three satellites, i.e., $N_{sat} = 3$. Initial event list represented by *link-on* and *link-off* events is known because of orbital mechanics, and I assume these events to be represented by the initial and final extremes of the colored horizontal bars in Fig.4.1, respectively. At time t_0, each node has only its own local model in memory, that I assume to have been trained on local dataset, as a matter of example, for $N_{ep} = 1$ epoch, i.e., $M_i = \{\mathbf{w}_i^{0,1}\},\ \forall i \in [0, \ldots, 2]$. However, at t_0 a link between nodes 0 and 1 becomes available, thus, the two nodes tries to share the models in their memories, i.e., node 0 tries to send its model to node 1, and vice versa. Assuming this information transfer needs a time $\Delta_{t,0}$ such that from t_0 to $t_1 = t_0 + \Delta_{t,0}$ the link between 0 and 1 is always active, a transfer completed event occurs at t_1. Furthermore, at t_1 memories are such that $M_0 = M_1 = \{\mathbf{w}_0^{0,1}, \mathbf{w}_1^{0,1}\}$ and $M_3 = \{\mathbf{w}_3^{0,1}\}$. Both nodes 0 and 1 will try to share their updated model lists with other nodes at t_1, but since no link is available, no further transfer happens. Next event occurs at t_2, when a link between nodes 1 and 2 becomes available. Thus, node 1 tries to share the models it has in its memory, i.e., both $\mathbf{w}_0^{0,1}$ and $\mathbf{w}_1^{0,1}$ with 2, and vice-versa. Again, assuming this information transfer needs a time $\Delta_{t,1}$ such that the link between nodes 1 and 2 remains active from t_2 to $t_3 = t_2 + \Delta_{t,1}$, transfer is possible and a transfer completed event happens at t_3. Thus, at this time $M_0 = \{\mathbf{w}_0^{0,1}, \mathbf{w}_1^{0,1}\}$ and $M_1 = M_2 = \{\mathbf{w}_0^{0,1}, \mathbf{w}_1^{0,1}, \mathbf{w}_2^{0,1}\}$ and both nodes 1 and 2 try to share their updated model lists with other nodes. In particular, since at t_3 a link between nodes 0 and 1 is active and will be still available during the full time span $\Delta_{t,2}$ needed to transmit $\mathbf{w}_2^{0,1}$ (the only local model being in node 1 memory but not in node 0 one), a transmission completed event occurs at $t_4 = t_3 + \Delta_{t,2}$. Since at this time memories are given by $M_0 = M_1 = M_2 = \{\mathbf{w}_0^{0,1}, \mathbf{w}_1^{0,1}, \mathbf{w}_2^{0,1}\}$, the local model distribution phase is over and the local learning phase begins. At time $\tilde{t} = t_4 + \tau_l$, a new round can start, repeating the same steps just described. Please notice that the new round event list will not contain all events happening between t_4 and $\tilde{t}$ (dashed regions in Fig.4.1). In particular, link-on events happening in this time span are moved to $\tilde{t}$ if the link is still available at the beginning of the new round, as in the case of the link between nodes 0 and 2 in the example shown in Fig.4.1.

4.3 Numerical results

In this section, I evaluate the performance of the proposed distributed learning solution with respect to federated learning-based one [18]. In fact, as previously discussed, both solutions allow to reduce the bandwidth and the energy consumption on ground with respect to a centralized learning solution where datasets acquired by satellites have to be transferred to a ground stations where the model is trained. However, a performance evaluation of the distributed learning and federated learning-based solutions should take into account the actual need of this application, i.e., to allow for reaching validation accuracy convergence as fast as possible. For this reason, the two strategies should be compared not in terms of bandwidth usage (they both already allow to lower the high bandwidth need associated to centralized learning), but in terms of how efficiently they leverage the available bandwidth,

given the application need. In particular, the number of completed learning rounds that can be accomplished in a repeat cycle may be a more insightful indicator, since the higher the number of completed round is, the higher the convergence speed is, thus, the better available bandwidth is leveraged to satisfy the application need. For this reason, I first evaluate the performance of the proposed distributed learning solution with respect to federated learning-based one by comparing the number of completed learning rounds that can be accomplished in a repeat cycle, and then I focus on the validation accuracy convergence time in case a deep learning satellite image classification model is trained on the EuroSAT[105] dataset. It is important to better clarify that, in case of distributed learning-based strategy, a learning round is made of a first inter-satellite model sharing phase, lasting until each satellite has received models of the remaining ones, followed by a local learning phase; instead, in case of federated learning-based strategies, a round provides for a first phase during which all satellites receive the updated global model from the ground, then local learning phase starts and, finally, local models are transmitted to the ground. In the proposed analysis, I will consider two schemes in the distributed learning strategy. In the first one, hereafter named "DL w/ GS", I assume that ground stations can contribute to model distribution as relay nodes, i.e., when a satellite A flies over a ground station, it shares the models it has in its memory with the ground, in such a way that when another satellite B flies over a ground station, it can receive the models memorized in A directly from the ground, without having to wait to communicate with A. Please notice that ground stations only act as models relay, and they do not have to receive all local models from satellites to calculate the global model, as it happens in federated learning solutions. Furthermore, I assume that ground stations are all interconnected in such a way that communicating with one of them is equivalent to communicate with all of them. For this reason, if a ground station receives models from a satellite, these are immediately available on any other ground station, too. Instead, the second distributed learning strategy considered (named "DL w/o GS") does not provide for involving ground stations as relay nodes, and satellites can share models only by means of in-orbit communications. As far as federated learning-based schemes considered as benchmarks are concerned, I consider a strategy leveraging both intra-orbital and inter-orbital ISLs ("FL w/ all ISLs"), a strategy where no inter-orbital communication is possible because of lack of inter-orbital ISLs ("FL w/o inter-orbital ISLs"), and a strategy providing for satellites having no ISLs ("FL w/o ISLs"), i.e., satellites cannot communicate with each other. All the federated learning solutions provide for having ground stations as parameter servers which, at each round, receive all the local models and aggregate them into a single global model which is transmitted to all satellites at the beginning of the following round. Furthermore, in this proposal I will assume that no aggregation of a partial number of local models happens on nodes neither in case of distributed learning-based solutions nor in federated learning-based ones.

Proposed analysis will be based on the following parameters: the number of satellites N_{sat}, the number of orbital planes N_{op}, the number of model parameters (i.e., weights and biases) N_{mp}, the duration of the local learning phase τ_l, the number of ground stations N_{GS} and their location. Please notice that, since model parameters are usually expressed in *float32* format, it easy to obtain the model size in bit given N_{mp}, since $w_s = 32 \cdot N_{mp}$ bit. The values of each parameter will

be specified for each of the following analysis. Instead, the remaining parameters will be the same for each analysis. In particular, I will assume to have a Walker constellation, with circular orbits having altitude $h_p = 712.84$ km, inclination 98.24 deg, repeat cycle $T = 2$ sidereal days, transmission data rate on both ISLs and links to the ground stations $R_{ISL} = R_{GS} = 200$ Mbps, transmission power $P = 10$ W, antenna gain $G = 34.31$ dBi, transmission frequency $\nu_{tx} = 26$ GHz, system noise temperature $T_s = 290$ K, bandwidth $B = 500$ MHz, minimum elevation angle $El_{min} = 5$ deg.

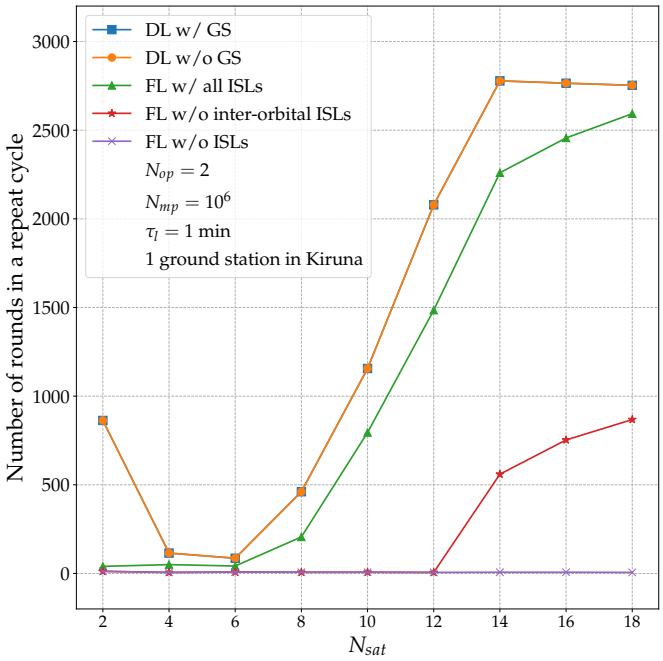

Figure 4.2. Number of completed rounds in a repeat cycle for different distributed learning-based and federated learning-based strategies, obtained by varying the number of satellites and by fixing the number of orbital planes $N_{op} = 2$, the number of model parameters $N_{mp} = 10^6$, the local learning time $\tau = 1$ min, and placing a single ground station in Kiruna (Sweden).

First analysis focuses on the impact of the number of satellites in an orbital plane on the number of completed rounds in a repeat cycle. For this analysis, I consider the number of satellites to be $N_{sat} \in [2, 4, \ldots, 18]$, equally distributed over $N_{op} = 2$ orbital planes, and I consider to have only a ground station placed in Kiruna (Sweden), a typical choice for orbits with the chosen inclination, a local learning time on satellites $\tau_l = 1$ min, and model having a number of parameters $N_{mp} = 10^6$. Results in Fig.4.2 show that any distributed learning-based strategy outperforms any federated learning-based strategy, regardless the number of satellites. However, it can be noticed that the number of rounds in a repeat cycle when a distributed learning-based strategy is applied first decreases with the number of satellites, then it starts increasing, and finally it decreases again. This is due to the fact that by increasing the number of satellites, the number of models that each satellite has to receive to calculate the global model increases, thus, the number of models to

be shared increases, too. At the same time, by increasing the number of satellites without changing the number of orbital planes, each orbit has an increased number of satellites, and this increases the communication possibilities within satellite couples. However, for $2 < N_{sat} \leq 6$, the increase in communication opportunities is not enough to allow the sharing of an increased number of models in the same time span as in case $N_{sat} = 2$, thus, the number of rounds in a repeat cycle decreases. Instead, for $N_{sat} > 6$, the increased number of communications between satellites is such that, even though the number of models to be shared increases, a complete sharing is achievable in a smaller time than in case $N_{sat} = 6$, thus, the number of rounds in a repeat cycle increases. In particular, for $N_{sat} \geq 10$, in the same amount of time it is even possible to share an increased number of models with respect to the case $N_{sat} = 2$, thus, a higher number of round in a repeat cycle is completed with respect to the ones accomplished when $N_{sat} = 2$. However, for $N_{sat} > 14$, the number of rounds completed in a repeat cycle starts decreasing again. This is due to the fact that, as it can be easily verified by means of expressions in (4.5), the intra-plane ISLs become always active for $N_{sat} \geq 14$, while they are never active when $N_{sat} < 14$. Thus, for $N_{sat} \geq 14$, as soon as a satellite receives models, it transmits them to satellites in the same orbital plane, with a delay only depending on the amount of models to be transferred and on the propagation time, without having to wait for an intra-plane ISLs to become available. Thus, the model sharing is completed shortly after any couple of satellites appertaining to different orbital planes is able to communicate. However, the maximum number of completed rounds is reached exactly when $N_{sat} = 14$, since for a higher number of satellites, even though inter-plane communications happen slightly earlier because of the increased number of satellites and, consequently, of the inter-plane communication opportunities, this does not compensate for the increase in the amount of data to be exchanged because of the increased number of models to be shared. Thus, for $N_{sat} > 14$, there is a mild decrease in the number of completed rounds in a repeat cycle. Still on the subject of distributed learning-based strategies behavior, it can be also noticed that "DL w/ GS" and "DL w/o GS" schemes lead to the same values of the number of completed rounds in a repeat cycle. This is due to the relative position of the satellites and the chosen ground station, such that when a satellite flies over the ground station, it already has in its memory the models available on ground because it has received them earlier directly from other satellites. This effect will be present in all analysis where a single ground station in Kiruna is considered, and it will be further discussed in the analysis related to the number of ground stations. Moving to federated learning-based strategies, it can be noticed that the best performance is obtained when all ISLs are leveraged, but the number of completed rounds in a repeat cycle is smaller than the one obtained in case of distributed learning strategies, because in federated learning schemes first all models have to reach the ground, where the global model is centrally determined, and then the ground station has to uplink the updated global model to all satellites. This also explains why, by increasing the number of satellites, there is an overall increase in the number of completed rounds in a repeat cycle, since the higher the number of satellites is, the higher the number of communication opportunities with the ground is. Finally, it can be noticed that "FL w/o inter-orbital ISLs" and "FL w/o ISLs" have the same behavior for $N_{sat} \leq 12$, while the latter leads to an increased number of completed

rounds in a repeat cycle for $N_{sat} > 12$. This is due to the fact that, as discussed before, for $N_{sat} \leq 12$, no intra-orbital ISL is available, while they are always available for $N_{sat} \geq 14$. Obviously, the worst performance in terms of completed rounds is obtained with the "FL w/o ISLs" solution, since by means of this strategy it is necessary to wait for all satellites to fly over the ground to transmit their models, and then, after the global model has been aggregated on the ground, it is necessary to wait again for all satellites to fly over the ground to receive it.

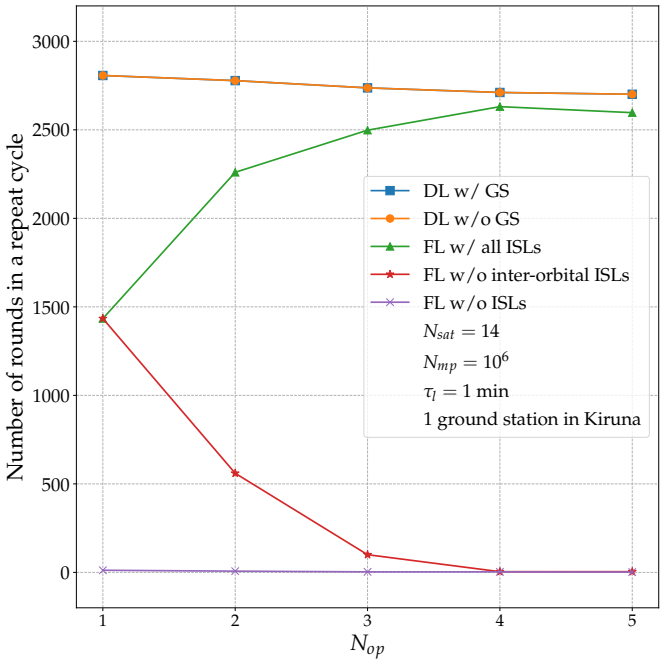

Figure 4.3. Number of completed rounds in a repeat cycle for different distributed learning-based and federated learning-based strategies, obtained by varying the number of orbital planes N_{op}, by considering 7 satellites on each orbital planes, $N_{mp} = 10^6$ model parameters, local learning time $\tau = 1$ min, and placing a single ground station in Kiruna (Sweden).

In Fig.4.3, I analyse the impact of the number of orbital planes, chosen to be $N_{op} \in [1, \ldots, 5]$, on the number of rounds completed in a repeat cycle when the different strategies previously introduced are applied. In particular, in this analysis I consider again a number of model parameters $N_{mp} = 10^6$, a learning time $\tau_l = 1$ min and a single ground station placed in Kiruna. Instead, the number of satellites will be equal to $N_{sat} = 7N_{op}$, in such a way that each orbital plane has 7 satellites and intra-orbital ISLs are always active, as previously discussed. Again, distributed learning-based solutions outperform the federated learning-based ones. However, it can be noticed that by increasing the number of orbital planes, the number of completed rounds decreases in case distributed learning-based strategies are applied, while it increases up to a maximum in case "FL w/ all ISLs" is considered. The behavior of the distributed learning solutions is due to the fact that, even though by increasing the number of orbital planes the communication opportunities among satellites on different orbital planes are more numerous, again this increase is not

enough to allow for the sharing of an increased number of models to conclude a learning round in the same time span, leading to an overall decrease in number of completed rounds in a time cycle. However, this decrease becomes milder by increasing the number of orbital planes, since this reduces the distances between couples of satellites on different orbits, increasing the number of communication opportunities until this is high enough to allow for the sharing of an increased number of models in almost the same time span. Instead, by looking at "FL w/ all ISLs" solution, it is possible to notice that the number of completed rounds increases for $N_{op} \leq 4$, and decreases when $N_{op} > 4$. This is due to the fact that, by increasing the number of orbital planes, there are both an increase in the communication opportunities among satellites appertaining to different orbital planes and in the communication opportunities among satellites and ground. This leads to the increase in number of completed rounds for $N_{op} \leq 4$. However, when $N_{op} = 4$, any couple of satellites can communicate, regardless of the occupied orbital plane, because intra-orbital ISLs are always active and the position of orbital planes is such that at any time there is at least a satellite of an orbital plane being able to communicate with a satellite of any other plane. Furthermore, for this number of orbital planes, at any time there is at least a satellite being able to communicate with the ground. It follows that any satellite in the constellation can communicate with the ground station at any time. Obviously, this property will be still valid by increasing the number of orbital planes, and the number of communication links between orbital planes will even increase with N_{op}. However, for $N_{op} > 4$, a higher number of models has to be shared with the ground because of the increased number of satellites, and this requires a higher amount of time because of the higher amount of data to be transmitted, leading to a decrease in number of completed rounds in a time cycle with respect to the case $N_{op} = 4$, since the increased number of communication opportunities is not high enough to allow for the increased number of models to be transmitted in the same time span. Finally, in case no inter-orbital ISL is leveraged, like in "FL w/o inter-orbital ISLs" and "FL w/o ISLs" solutions, there is no advantage in increasing the number of orbital planes, and the number of completed rounds in a repeat cycle decreases when N_{op} increases because of the increased number of satellites having to communicate with the ground to complete a learning round.

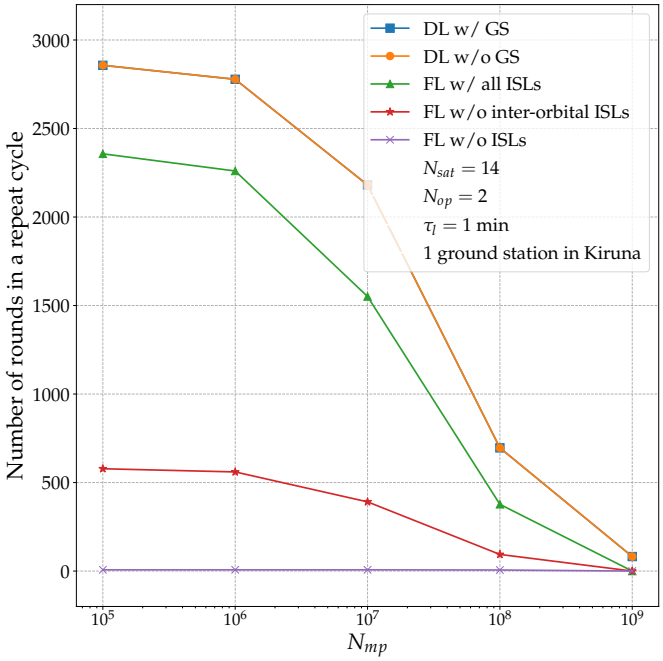

Figure 4.4. Number of completed rounds in a repeat cycle for different distributed learning-based and federated learning-based strategies, obtained by varying the number of model parameters and by fixing the number of satellites $N_{sat} = 14$, the number of orbital planes $N_{op} = 2$, the local learning time $\tau = 1$ min, and placing a single ground station in Kiruna (Sweden).

I also analyse the impact of the number of model parameters on the number of completed learning rounds in a repeat cycle. In particular, I considered $N_{mp} \in \{10^5, 10^6, 10^7, 10^8, 10^9\}$, I fixed the number of satellites $N_{sat} = 14$, distributed over $N_{op} = 2$ orbital planes, I assumed a local learning time $\tau_l = 1$ min and a single ground station placed in Kiruna. Please notice that, even though the local learning time is actually dependent on the number of model parameters, it also depends on the available computational capacity. For this reason, I left the local learning time as an analysis parameter that will be investigated further on. Results shown in Fig.4.4 allow to conclude that distributed learning-based strategies outperform the federated learning-based ones for any value of the number of model parameters. However, the number of completed rounds decreases when N_{mp} increases. This is due to the fact that, by increasing the number of parameters, there is an increase in the amount of data to be transmitted, and, consequently, in the time needed to accomplish the data transfer, making each learning round longer. However, in case of distributed learning, since it is not necessary to wait to first transfer all local models to the ground and then receive the updated global model from the Earth, there is an overall shorter duration of model distribution phase, which allows for completing a round in a shorter time and, consequently, to have a higher number of completed rounds in a repeat cycle.

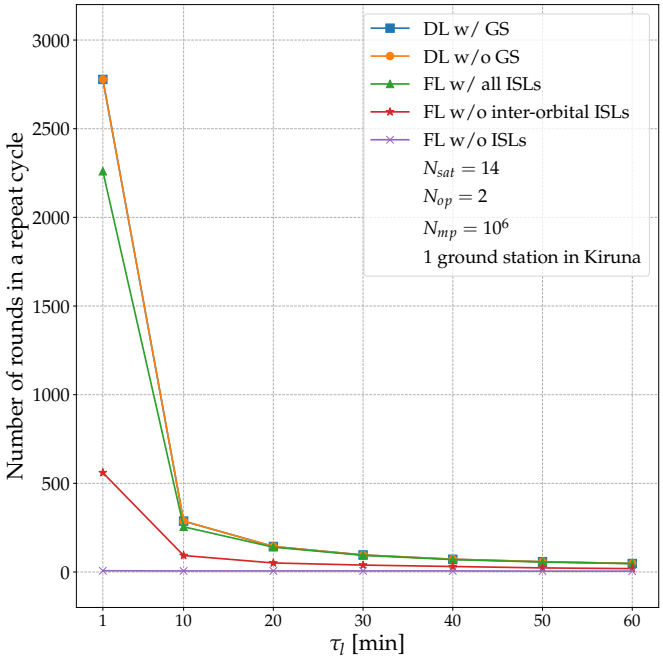

Figure 4.5. Number of completed rounds in a repeat cycle for different distributed learning-based and federated learning-based strategies, obtained by varying the learning time and by fixing the number of satellites $N_{sat} = 14$, the number of orbital planes $N_{op} = 2$, the number of model parameters $N_{mp} = 10^6$, and placing a single ground station in Kiruna (Sweden).

Instead, in Fig.4.5 I investigate the impact of local learning time on the number of completed rounds in a repeat cycle, considering $\tau_l \in \{1, 10, 20, 30, 40, 50, 60\}$ min, a number of model parameters $N_{mp} = 10^6$ and, again, $N_{sat} = 14$ satellites distributed over $N_{op} = 2$ orbital planes and a single ground station in Kiruna. It can be noticed that, even though distributed learning-based solutions outperform federated learning-based ones for small τ_l values, the difference in the number of completed rounds in a repeat cycle becomes marginal as τ_l increases. This is a consequence of the fact that, by increasing τ_l, the contribution to the time to complete a learning round of the local learning time increases with respect to the model distribution time. In particular, in the overall duration of a learning round, the more τ_l increases, the more negligible model distribution time is: for this reason, since the chosen strategy only impacts on the model distribution phase duration, the differences between the considered strategies in the number of completed rounds decrease as τ_l increases.

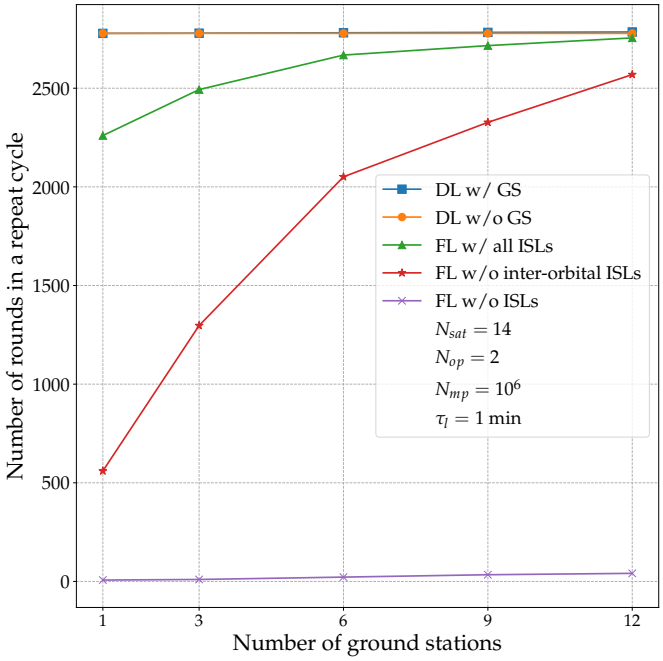

Figure 4.6. Number of completed rounds in a repeat cycle for different distributed learning-based and federated learning-based strategies, obtained by varying the number of ground stations and by fixing the number of satellites $N_{sat} = 14$, the number of orbital planes $N_{op} = 2$, the number of model parameters $N_{mp} = 10^6$, and the local learning time $\tau = 1$ min.

Results in Fig.4.6 give insight on the impact of the number of ground stations on the number of completed learning rounds in a repeat cycle for the different considered strategies. In particular, I set the number of satellites $N_{sat} = 14$, the number of orbital planes $N_{op} = 2$, the number of model parameters $N_{mp} = 10^6$, the local learning time $\tau_l = 1$ min, and I consider an increasing number of ground stations, mainly provided by Amazon Web Services[99], grouped as described in the last numerical analysis commented in Subsection 2.5.2.

From Fig.4.6 it is possible to notice that all solutions providing for leveraging ground stations, i.e., "DL w/ GS" and the three federated learning-based strategies, improves in terms of number of completed rounds in a repeat cycle when the number of ground stations increases, as a consequence of the fact that there are more communication possibilities with ground stations when their number increases. Furthermore, it is important to underline that since the ground stations are considered to be interconnected, as soon as a model is available on one of them, it will be immediately available on each of them, facilitating the model sharing within the constellation, since a model can be transferred between two satellites not being able to directly communicate but simultaneously flying over two different ground stations. This reflects in the fact that the "DL w/ GS" strategy allows for completing a slightly higher number of rounds in a repeat cycle than the "DL w/o GS", and the "FL w/ all ISLs" solution achieves almost the same performance as the distributed learning-based ones for a high number of ground stations.

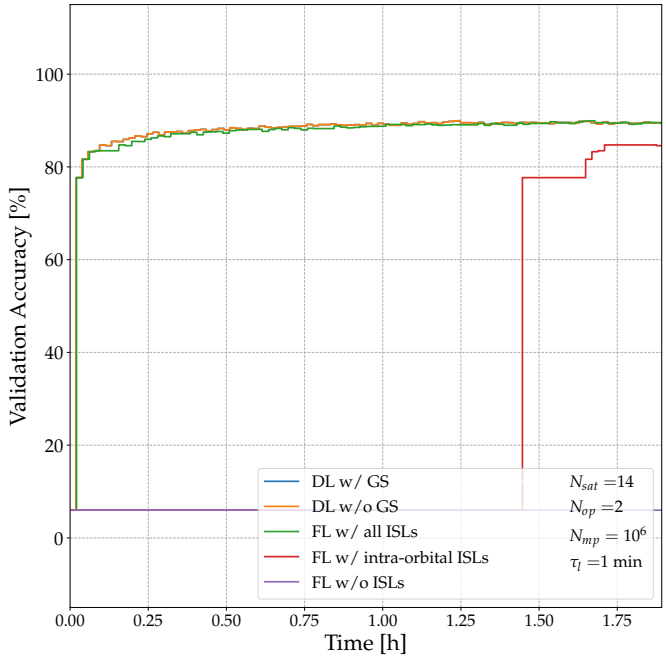

Figure 4.7. Validation accuracy in training a VGG16-based satellite image classification model on the EuroSAT dataset by applying different distributed learning-based and federated learning-based strategies, obtained by setting the number of satellites $N_{sat} = 14$, the number of orbital planes $N_{op} = 2$, the number of model parameters $N_{mp} = 10^6$, the local learning time $\tau = 1$ min and a ground station in Kiruna (Sweden).

Finally, I evaluate the time of convergence of the validation accuracy when the different strategies are applied. For this analysis I consider to have $N_{sat} = 14$ satellites distributed over $N_{op} = 2$ orbital planes, a local learning time $\tau_l = 1$ min and a single ground station placed in Kiruna. I also set the maximum number of learning rounds to be $N_r = 100$ and the number of local learning epochs $N_{ep} = 1$. I consider a land cover classification task based on the EuroSAT dataset[105], made of 27000 64x64 images, taken by Sentinel-2. Images are classified with respect to 10 classes (AnnualCrop, Forest, HerbaceousVegetation, Highway, Industrial, Pasture, PermanentCrop, Residential, River, SeaLake), depending on the represented scene. I am aware of the fact that training a classification model requires labeled data, and this may be not the case when considering images acquired from satellites to be used for training without previously transmitting them to the ground. However, since the focus of this chapter is on the communication strategy underlying the learning algorithm, I only want to provide some insight on the performance of the communication scheme by focusing on a general machine learning task, as it happens also in other works [17, 18, 19, 20, 106], since this insight may be also extended to more sophisticated machine learning techniques, like self-supervised learning, which are beyond the scope of this chapter. Furthermore, it is important to underline that, given the model to be trained, the chosen strategy does not influence the accuracy obtained when training converges, but only the time needed to reach validation convergence. Following [107], I consider VGG16[108] as classification model, pre-trained on ImageNet dataset, to which a regular densely-connected neural network

of 2048 units with ReLU activation function, a dropout layer with 0.2 drop rate, and a regular densely-connected neural network of 10 units with softmax activation function are added. I will assume that only these added layers will be trained, thus, the number of model parameters to be exchanged will be equal to $N_{mp} = 4216842$. Differently from [107], I assume that the model is not trained on a central node, but each satellite trains its own local model during local training phase. In particular, each satellite will have a different dataset, since satellites fly over different areas. However, after some orbits, each satellite will have flown over a high amount of different areas, thus, I suppose that in datasets of each satellite there are samples for all classes, but the distribution of samples on each orbital plane with respect to the different classes is different. In particular, I randomly split the initial dataset to separate a 20% of samples for validation. For each image in the training set, I generate a randomly rotated version and a noisy version of it, and I add the two new images to the training set for augmentation purposes. In order to obtain the training sets associated to each satellite, for each class I split the samples between the two orbital planes in random proportion, and the samples for each class associated to each orbital plane are equally and randomly associated to each satellite on the orbital plane. This also allows to obtain non-IID training sets on the satellites. I thus evaluate the accuracy of the global model (i.e., of the model obtained by aggregating the locally trained models) on the validation set for each learning round. Since I know how long each learning round lasts when the different strategies are applied, it is easy to report the validation accuracy in time, as shown in Fig.4.7. I also evaluate the time to converge as the time at which validation accuracy reaches a value that is not improved in the next 10 learning rounds. Values of the time to converge are summarized in Tab.4.3.

Table 4.3. Time to converge

Strategy	Time to converge
DL w/ GS	1.25
DL w/o GS	1.25
FL w/ all ISLs	1.68
FL w/ intra-orbital ISLs	4.13
FL w/o ISLs	558 (23.3 days)

From presented results, it can be noticed that by using strategies providing for the use of inter-orbital ISLs, like in case of distributed learning-based and "FL w/ all ISLs" schemes, validation accuracy converges much faster than in case of strategies where inter-orbital ISLs are not used. This is due to the fact that, as previously discussed, by leveraging inter-orbital ISLs there is an increase in the communication opportunities among satellites. Furthermore, distributed learning-based strategies, thanks to a reduced duration of the model distribution phase, allows to reach convergence in a shorter time than any federated learning-based solution.

4.4 Conclusions

In this chapter, I proposed and evaluated a distributed learning solution in the context of EO constellations with satellites forming a network by means of ISLs.

Similarly to federated learning, the proposed solution allows to train ML models directly in-orbit, without having to share a dataset of acquired images with a central node on ground where a centralized model training is accomplished. In this way, it is possible to obtain a reduction of both bandwidth usage and energy consumption on ground stations. The proposed solution differs from the federated learning one in the fact that there is no central node which has to receive the local models to aggregate them in an updated version of the global model, since I assume that satellites share local models with each other until each satellite has received the local models of the others, in order to locally calculate the updated global model. Numerical results show that distributed learning outperforms federated learning in number of learning rounds completed in the unit time by increasing the number of satellites, of orbital planes, of model parameters, of ground stations and by increasing the time needed to accomplish local learning. These results provide evidence of a more efficient bandwidth usage in case distributed learning-bases solutions are applied, since a higher number of learning rounds allows to satisfy the need for reaching accuracy convergence as fast as possible. This is also confirmed by results related to the time needed to reach validation accuracy convergence, evaluated on the training of a deep learning-based land coverage classification model by means of the EuroSAT dataset.

Conclusions and Future Developments

In this thesis, I designed and investigated strategies to allocate transmission, storage and processing resources in a constellation dedicated to EO where satellites form a network by means of intra-orbital and inter-orbital ISLs. Differently from works available in literature, I proposed to leverage this network to enable task offloading within the constellation, in such a way that if a satellite has not enough resources to elaborate images, this processing can be executed by another satellite. In particular, the presented strategies aimed to minimize the constellation total operating cost, to minimize the energy amount consumed by ground stations to process EO images, or to support in-orbit training of deep learning models by means of distributed learning techniques.

As far as operating cost minimization is concerned, I first introduced and solved an optimization problem to jointly allocate processing and communication resources to reach this goal. Furthermore, since the optimal problem resulted to be NP-hard, I also proposed two heuristics, which have been validated against optimal results and applied in a real orbital scenario, proving that they outperform benchmark solutions where task processing can happen only either on the acquiring satellite or the ground station. Furthermore, even though these strategies aim to minimize costs instead of delay, they also outperform state-of-the-art solutions in delivering information to the ground in a shorter time. An interesting extension of the proposed strategies would be to consider that an image processing service may be made of different tasks to be executed in a predetermined order, with each task being potentially assigned to different satellites within the constellation, in order to complete the service processing in a shared way. It is also important to underline that the proposed strategies are centralized. In particular, they provide for a ground control station knowing the positions of satellites in advance by means of orbital mechanics, and, consequently, the instants during which each ISL is active and a region of interest is flown over are supposed to be known *a priori*. This allows the central ground control station to define where each image has to be processed and the routes to be followed during a repeat cycle beforehand. Thus, the resource allocation decision is uplinked to satellites executing it. However, this operation scheme may be not robust to link failures, since these cannot be provided in advance. Furthermore, in case of a change in the regions of interest over which images have to be acquired, the full resource allocation decision has to be recalculated. In order to overcome this potential issues, it may be interesting to extend the proposed strategy in a

distributed and dynamical sense, providing for resource allocation decisions to be made in time by the single satellites, which collaborate in such a way that a local decision made by a single satellite contributes to reach a global optimum.

Moving to the minimization of ground station energy consumption related to EO image processing, I proposed and investigated two optimal strategies to leverage at most in-orbit processing, avoiding for image elaboration to be executed on-ground. It is important to underline that different resource allocation decisions may lead to the same optimal amount of on-ground energy saving. In particular, in the first optimal strategy I only considered the maximization of energy saving on ground stations, without taking into account any optimization of the on-board batteries to preserve satellite operative life. In this case, since the proposed formulation resulted to be NP-complete, I also introduced a heuristic-based strategy, which has been validated against the optimal scheme and applied to a real orbital scenario, showing considerable improvements with respect to benchmark solutions. Instead, a second optimization problem aimed to jointly maximize the on-ground energy saving and minimize the on-board batteries depth-of-discharge to optimize the satellite operative life. Results related to both strategies also provided useful insights on how the processing capacity (i.e., CPU clock frequency) available on-board of satellites should be chosen to achieve optimal performance. In particular, results show that these are not obtained by installing CPUs having the highest clock frequency possible, since energy consumption quadratically increases with the CPU frequency and limitations on energy available on-board would prevent the full leverage of the available computational capacity. A first extension of the proposed strategy would be, again, to consider that a service may be made of more than a single task. For this reason, strategies to in-orbit elaborate most of these tasks to save energy on ground stations may be proposed. Similarly to the strategies aiming to minimize operating cost, also these strategies are centralized and may be not robust to link failures or not scalable whenever frequent changes in regions to be monitored happen. For this reason, an interesting future development of this research would be, again, the extension of the proposed strategies in a distributed and dynamical manner. Furthermore, the heuristic proposed in case no limitations on battery usage are considered provided for the use of an empirically defined energy margin to be left on nodes to prevent resource shortage due to the greedy behaviour of the proposed algorithm. However, it may be interesting to investigate further ways to estimate this margin, or to design strategies that do not have to rely on it. Finally, since the optimization problem proposed to jointly optimize on-ground energy saving and in-orbit battery depth-of-discharge is also NP-complete, a near-optimal strategy to obtain a resource allocation decision in reasonable time in a real orbital scenario may be proposed.

Regarding the support to in-orbit distributed learning, since such solution allows to reduce both bandwidth usage and ground stations energy consumption with respect to centralized learning schemes, I proposed a communication strategy where satellite networks made of intra-orbital and inter-orbital ISLs are leveraged to allow for satellites exchanging their locally trained models in such a way each satellite has the possibility to calculate a global model on its own, without having to lean on a central aggregating node as it happens in federated learning strategies. Distributed learning schemes enabled by this communication solution show to

outperform federated learning-based ones in completing a higher number of learning rounds in a repeat cycle, and this allows for reaching model convergence in a shorter time, as it has been confirmed by evaluating a distributed training of a deep learning satellite image classification model on the EuroSAT dataset. The proposed strategy provides for each satellite to share all the received models with other satellites, without any partial aggregation of the received local models in a new single model. Since this could lead to a further saving in bandwidth and, consequently, to the completion of a higher number of learning rounds in a shorter time, allowing to further reduce the time needed to reach model convergence, it may be interesting to extend the proposed communication scheme to embed both an optimal strategy enabling satellites to aggregate received weights and biases in partial model to be shared, and an appropriate management of the received aggregated models to correctly evaluate the global one.

Bibliography

[1] B. Denby and B. Lucia, "Orbital edge computing: Nanosatellite constellations as a new class of computer system," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 939–954. [Online]. Available: https://doi.org/10.1145/3373376.3378473

[2] G. Giuffrida, L. Fanucci, G. Meoni, M. Batič, L. Buckley, A. Dunne, C. van Dijk, M. Esposito, J. Hefele, N. Vercruyssen *et al.*, "The ϕ-sat-1 mission: the first on-board deep neural network demonstrator for satellite earth observation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–14, 2021.

[3] G. Mateo-Garcia, J. Veitch-Michaelis, L. Smith, S. V. Oprea, G. Schumann, Y. Gal, A. G. Baydin, and D. Backes, "Towards global flood mapping onboard low cost satellites with machine learning," *Scientific reports*, vol. 11, no. 1, pp. 1–12, 2021.

[4] C. Hao, X. Ming, and P. Zhibo, "Satellite-based computing networks with federated learning," *IEEE Wireless Communications*, vol. 9, pp. 78–84, 2022.

[5] R. Nasrin, M. Bho, D. Armin, and P. Petar, "On-board federated learning for dense leo constellations," in *ICC 2022 - IEEE International Conference on Communications*. IEEE, 2022, pp. 1–6.

[6] Y. Wang, J. Che, N. Wang, L. Liu, N. Wu, X. Zhong, and X. Han, "Load-balancing method for leo satellite edge-computing networks based on the maximum flow of virtual links," *IEEE Access*, vol. 10, pp. 100 584–100 593, 2022.

[7] K. Taeyeoun, K. Jeongho, and C. Jihwan, "Satellite edge computing architecture and network slice scheduling for iot support," *IEEE Internet of Things journal*, vol. 9, no. 16, pp. 14 938–14 951, 2022.

[8] J. Min, Z. Liang, W. Jann, G. Qing, and G. Xuemai, "Joint computing and communication resource allocation for edge computing towards huge leo networks," *China Communications Magazine*, vol. 8, pp. 73–84, 2022.

[9] Y. Xiumei and H. Bo, "Cost-efficient task offloading for satellite edge computing systems," in *IEEE International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2022, pp. 1–6.

[10] C. Lei, F. Gang, S. Yao, L. Mengjie, and Q. Shuang, "Dynamic computation offloading in satellite edge computing," in *ICC 2022 - IEEE International Conference on Communications*. IEEE, 2022, pp. 1–6.

[11] M. M. Gost, I. Leyva-Mayorga, A. Pérez-Neira, M. Á. Vázquez, B. Soret, and M. Moretti, "Edge computing and communication for energy-efficient earth surveillance with leo satellites," in *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2022, pp. 556–561.

[12] Q. Li, S. Wang, X. Ma, A. Zhou, and F. Yang, "Towards sustainable satellite edge computing," in *2021 IEEE International Conference on Edge Computing (EDGE)*. IEEE, 2021, pp. 1–8.

[13] R. Verduci, V. Romano, G. Brunetti, N. Yaghoobi Nia, A. Di Carlo, G. D'Angelo, and C. Ciminelli, "Solar energy in space applications: review and technology perspectives," *Advanced Energy Materials*, vol. 12, no. 29, p. 2200125, 2022.

[14] B. Mao, F. Tang, Y. Kawamoto, and N. Kato, "Ai models for green communications towards 6g," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 210–247, 2021.

[15] Z. Jia, M. Sheng, J. Li, D. Zhou, and Z. Han, "Vnf-based service provision in software defined leo satellite networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 6139–6153, 2021.

[16] Z. Jia, M. Sheng, J. Li, R. Liu, K. Guo, Y. Wang, D. Chen, and R. Ding, "Joint optimization of vnf deployment and routing in software defined satellite networks," in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2018, pp. 1–5.

[17] B. Matthiesen, N. Razmi, I. Leyva-Mayorga, A. Dekorsy, and P. Popovski, "Federated learning in satellite constellations," *IEEE Network*, 2023.

[18] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "On-board federated learning for dense leo constellations," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 4715–4720.

[19] ——, "Ground-assisted federated learning in leo satellite constellations," *IEEE Wireless Communications Letters*, vol. 11, no. 4, pp. 717–721, 2022.

[20] ——, "Scheduling for ground-assisted federated learning in leo satellite constellations," in *2022 30th European Signal Processing Conference (EUSIPCO)*. IEEE, 2022, pp. 1102–1106.

[21] SpaceX. Starlink Official Website. Available at https://www.starlink.com/ (accessed October 23, 2023).

[22] Amazon. Project Kuiper Official Website. Available at https://www.aboutamazon.com/news/tag/project-kuiper (accessed October 23, 2023).

[23] F. Rinaldi, H.-L. Maattanen, J. Torsner, S. Pizzi, S. Andreev, A. Iera, Y. Koucheryavy, and G. Araniti, "Non-Terrestrial Networks in 5G & Beyond: A Survey," *IEEE Access*, vol. 8, pp. 165 178–165 200, 2020.

[24] G. Araniti, A. Iera, S. Pizzi, and F. Rinaldi, "Toward 6g Non-Terrestrial Networks," *IEEE Network*, vol. 36, no. 1, pp. 113–120, 2021.

[25] J. Liu, Y. Shi, Z. M. Fadlullah, and N. Kato, "Space-Air-Ground Integrated Network: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2714–2741, 2018.

[26] P. Cassará, A. Gotta, M. Marchese, and F. Patrone, "Orbital Edge Offloading on Mega-LEO Satellite Constellations for Equal Access to Computing," *IEEE Communications Magazine*, vol. 60, no. 4, pp. 32–36, 2022.

[27] K. S. Basavaraju, N. Sravya, S. Lal, J. Nalini, C. S. Reddy, and F. Dell'Acqua, "Ucdnet: A deep learning model for urban change detection from bi-temporal multispectral sentinel-2 satellite images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–10, 2022.

[28] N. E. Khalifa, M. Loey, and S. Mirjalili, "A comprehensive survey of recent trends in deep learning for digital images augmentation," *Artificial Intelligence Review*, pp. 1–27, 2022.

[29] X. Lin, S. Rommer, S. Euler, E. A. Yavuz, and R. S. Karlsson, "5g from space: An overview of 3gpp non-terrestrial networks," *IEEE Communications Standards Magazine*, vol. 5, no. 4, pp. 147–153, 2021.

[30] F. Rinaldi, S. Pizzi, A. Molinaro, A. Iera, and G. Araniti, "Cooperative resource allocation in integrated terrestrial/non-terrestrial 5g and beyond networks," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[31] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. M. Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani *et al.*, "Evolution of non-terrestrial networks from 5g to 6g: A survey," *IEEE communications surveys & tutorials*, 2022.

[32] M. Giordani and M. Zorzi, "Non-terrestrial networks in the 6g era: Challenges and opportunities," *IEEE Network*, vol. 35, no. 2, pp. 244–251, 2020.

[33] ——, "Satellite communication at millimeter waves: A key enabler of the 6g era," in *2020 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2020, pp. 383–388.

[34] Y. Zhang, Q. Wu, Z. Lai, and H. Li, "Enabling low-latency-capable satellite-ground topology for emerging leo satellite networks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1329–1338.

[35] Z. Lai, W. Liu, Q. Wu, H. Li, J. Xu, and J. Wu, "Spacertc: Unleashing the low-latency potential of mega-constellations for real-time communications," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1339–1348.

[36] European Space Agency - Connectivity and Secure Communications. ANChOR - Data-driven Network Controller and Orchestrator for Real-time Network Management. Available at https://connectivity.esa.int/projects/anchor (accessed October 23, 2023).

[37] F. Patrone, G. Bacci, A. Galli, P. Giardina, G. Landi, M. Luglio, M. Marchese, M. Quadrini, C. Roseti, G. Sperlì *et al.*, "Data-driven network orchestrator for 5g satellite-terrestrial integrated networks: the anchor project," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[38] A. Galli, P. Giardina, M. Guta, L. Lossi, A. Mancina, V. Moscato, F. Patrone, C. Roseti, S. P. Romano, G. Sperlí *et al.*, "Ai for zero-touch management of satellite networks in b5g and 6g infrastructures," in *First International Workshop on Artificial Intelligence in beyond 5G and 6G Wireless Networks*, 2022, pp. 1–9.

[39] V. Eramo, F. Valente, T. Catena, and F. G. Lavacca, "Proposal and investigation of a convolutional and lstm neural network for the cost-aware resource prediction in softwarized networks," *Future Internet*, vol. 13, no. 12, p. 316, 2021.

[40] V. Eramo, F. Valente, F. G. Lavacca, and T. Catena, "Ai-based resource prediction in network function virtualization architectures," in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2021, pp. 909–914.

[41] ——, "Cost-aware and ai-based resource prediction in softwarized networks," in *2021 AEIT International Annual Conference (AEIT)*. IEEE, 2021, pp. 1–4.

[42] S. Saafi, O. Vikhrova, G. Fodor, J. Hosek, and S. Andreev, "Ai-aided integrated terrestrial and non-terrestrial 6g solutions for sustainable maritime networking," *IEEE Network*, vol. 36, no. 3, pp. 183–190, 2022.

[43] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[44] C. Jiang and X. Zhu, "Reinforcement learning based capacity management in multi-layer satellite networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 7, pp. 4685–4699, 2020.

[45] X. Wang, Z. Dai, and Z. Xu, "Leo satellite network routing algorithm based on reinforcement learning," in *2021 IEEE 4th International Conference on Electronics Technology (ICET)*. IEEE, 2021, pp. 1105–1109.

[46] K.-C. Tsai, L. Fan, L.-C. Wang, R. Lent, and Z. Han, "Multi-commodity flow routing for large-scale leo satellite networks using deep reinforcement learning," in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2022, pp. 626–631.

[47] F. Tang, H. Hofner, N. Kato, K. Kaneko, Y. Yamashita, and M. Hangai, "A deep reinforcement learning-based dynamic traffic offloading in space-air-ground integrated networks (sagin)," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 276–289, 2022.

[48] T. K. Rodrigues and N. Kato, "Network slicing with centralized and distributed reinforcement learning for combined satellite/ground networks in a 6g environment," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 104–110, 2022.

[49] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE signal processing magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[50] P. Zhang, Y. Zhang, N. Kumar, and M. Guizani, "Dynamic sfc embedding algorithm assisted by federated learning in space-air-ground integrated network resource allocation scenario," *IEEE Internet of Things Journal*, 2022.

[51] H. Xu, S. Han, X. Li, and Z. Han, "Anomaly traffic detection based on communication-efficient federated learning in space-air-ground integration network," *IEEE Transactions on Wireless Communications*, no. 99, pp. 1–1, 2023.

[52] F. Tang, C. Wen, X. Chen, and N. Kato, "Federated learning for intelligent transmission with space-air-ground integrated network (sagin) toward 6g," *IEEE Network*, 2022.

[53] H. Chen, M. Xiao, and Z. Pang, "Satellite-based computing networks with federated learning," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 78–84, 2022.

[54] Y. Qiu, J. Niu, X. Zhu, K. Zhu, Y. Yao, B. Ren, and T. Ren, "Mobile Edge Computing in Space-Air-Ground Integrated Networks: Architectures, Key Technologies and Challenges," *Journal of Sensor and Actuator Networks*, vol. 11, no. 4, p. 57, 2022.

[55] Z. Zhang, W. Zhang, and F.-H. Tseng, "Satellite mobile edge computing: Improving qos of high-speed satellite-terrestrial networks using edge computing techniques," *IEEE network*, vol. 33, no. 1, pp. 70–76, 2019.

[56] F. Wang, D. Jiang, S. Qi, C. Qiao, and L. Shi, "A dynamic resource scheduling scheme in edge computing satellite networks," *Mobile Networks and Applications*, vol. 26, pp. 597–608, 2021.

[57] Y. Hu and W. Gong, "An on-orbit task-offloading strategy based on satellite edge computing," *Sensors*, vol. 23, no. 9, p. 4271, 2023.

[58] Y. Wang, J. Yang, X. Guo, and Z. Qu, "A game-theoretic approach to computation offloading in satellite edge computing," *IEEE Access*, vol. 8, pp. 12 510–12 520, 2019.

[59] Q. Li, S. Wang, X. Ma, Q. Sun, H. Wang, S. Cao, and F. Yang, "Service coverage for satellite edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 695–705, 2021.

[60] T. Kim, J. Kwak, and J. P. Choi, "Satellite edge computing architecture and network slice scheduling for iot support," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14 938–14 951, 2022.

[61] Y. Wang, J. Yang, X. Guo, and Z. Qu, "Satellite edge computing for the internet of things in aerospace," *Sensors*, vol. 19, no. 20, p. 4375, 2019.

[62] F. Chai, Q. Zhang, H. Yao, X. Xin, R. Gao, and M. Guizani, "Joint multi-task offloading and resource allocation for mobile edge computing systems in satellite iot," *IEEE Transactions on Vehicular Technology*, 2023.

[63] Z. Song, Y. Hao, Y. Liu, and X. Sun, "Energy-efficient multiaccess edge computing for terrestrial-satellite internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 14 202–14 218, 2021.

[64] S. Yu, X. Gong, Q. Shi, X. Wang, and X. Chen, "EC-SAGINs: Edge-Computing-Enhanced Space–Air–Ground-Integrated Networks for Internet of Vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5742–5754, 2022.

[65] Z. Jia, M. Sheng, J. Li, Y. Zhu, W. Bai, and Z. Han, "Virtual network functions orchestration in software defined leo small satellite networks," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[66] X. Gao, R. Liu, and A. Kaushik, "Virtual network function placement in satellite edge computing with a potential game approach," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1243–1259, 2022.

[67] D. Yang, Z. Wang, H. Ding, S. Zhu, S. Meng, and J. Duan, "Dynamic resource allocation of network function virtualization for space-air-ground integrated energy internet," in *2021 International Conference on Power System Technology (POWERCON)*. IEEE, 2021, pp. 1897–1903.

[68] L. Zhang, C. Yang, Y. Ouyang, T. Li, and A. Anpalagan, "Isfc: Intent-driven service function chaining for satellite networks," in *2022 27th Asia Pacific Conference on Communications (APCC)*. IEEE, 2022, pp. 544–549.

[69] M. R. Haque, S. C. Tan, Z. Yusoff, K. Nisar, C. K. Lee, B. Chowdhry, S. Ali, S. K. Memona, and R. Kaspin, "Sdn architecture for uavs and evs using satellite: A hypothetical model and new challenges for future," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2021, pp. 1–6.

[70] P. Kumar, S. Bhushan, D. Halder, and A. M. Baswade, "fybrrlink: Efficient qos-aware routing in sdn enabled future satellite networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2107–2118, 2021.

[71] A. Papa, T. De Cola, P. Vizarreta, M. He, C. Mas-Machuca, and W. Kellerer, "Design and evaluation of reconfigurable sdn leo constellations," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1432–1445, 2020.

[72] J. Guo, L. Yang, D. Rincón, S. Sallent, Q. Chen, and X. Liu, "Static placement and dynamic assignment of sdn controllers in leo satellite networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4975–4988, 2022.

[73] D. K. Luong, Y.-F. Hu, J.-P. Li, and M. Ali, "Metaheuristic approaches to the joint controller and gateway placement in 5g-satellite sdn networks," in *ICC 2020-2020 IEEE international conference on communications (ICC)*. IEEE, 2020, pp. 1–6.

[74] F. Mendoza, M. Minardi, S. Chatzinotas, L. Lei, and T. X. Vu, "An sdn based testbed for dynamic network slicing in satellite-terrestrial networks," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2021, pp. 36–41.

[75] Z. Lai, Q. Wu, H. Li, M. Lv, and J. Wu, "Orbitcast: Exploiting mega-constellations for low-latency earth observation," in *2021 IEEE 29th International Conference on Network Protocols (ICNP)*. IEEE, 2021, pp. 1–12.

[76] M. Lyu, Q. Wu, Z. Lai, H. Li, Y. Li, and J. Liu, "Falcon: Towards fast and scalable data delivery for emerging earth observation constellations," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 2023, pp. 1–10.

[77] H. Chen, X. Zhang, L. Wang, L. Xing, and W. Pedrycz, "Resource-constrained self-organized optimization for near-real-time offloading satellite earth observation big data," *Knowledge-Based Systems*, vol. 253, p. 109496, 2022.

[78] P. Wang, H. Li, B. Chen, and S. Zhang, "Enhancing earth observation throughput using inter-satellite communication," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 7990–8006, 2022.

[79] G. Li, "Online scheduling of distributed earth observation satellite system under rigid communication constraints," *Advances in Space Research*, vol. 65, no. 11, pp. 2475–2496, 2020.

[80] Z. Chang, Z. Zhou, L. Xing, and F. Yao, "Integrated scheduling problem for earth observation satellites based on three modeling frameworks: an adaptive bi-objective memetic algorithm," *Memetic Computing*, vol. 13, no. 2, pp. 203–226, 2021.

[81] M. Ghiglione and V. Serra, "Opportunities and challenges of ai on satellite processing units," in *Proceedings of the 19th ACM international conference on computing Frontiers*, 2022, pp. 221–224.

[82] ESA Phi-lab. Phi-sats programme. Available at https://philab.esa.int/flagship-programmes/phi-sats-programme/ (accessed October 23, 2023).

[83] G. Guerrisi, F. Del Frate, and G. Schiavon, "Artificial intelligence based on-board image compression for the ϕ-sat-2 mission," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.

[84] P. Xu, Q. Li, B. Zhang, F. Wu, K. Zhao, X. Du, C. Yang, and R. Zhong, "On-board real-time ship detection in hisea-1 sar images based on cfar and lightweight deep learning," *Remote Sensing*, vol. 13, no. 10, p. 1995, 2021.

[85] D. D. Langer, M. Orlandić, S. Bakken, R. Birkeland, J. L. Garrett, T. A. Johansen, and A. J. Sørensen, "Robust and reconfigurable on-board processing for a hyperspectral imaging small satellite," *Remote Sensing*, vol. 15, no. 15, p. 3756, 2023.

[86] M. P. Del Rosso, A. Sebastianelli, D. Spiller, P. P. Mathieu, and S. L. Ullo, "On-board volcanic eruption detection through cnns and satellite multispectral imagery," *Remote Sensing*, vol. 13, no. 17, p. 3479, 2021.

[87] J. Nalepa, M. Myller, M. Cwiek, L. Zak, T. Lakota, L. Tulczyjew, and M. Kawulok, "Towards on-board hyperspectral satellite image segmentation: Understanding robustness of deep learning through simulating acquisition conditions," *Remote sensing*, vol. 13, no. 8, p. 1532, 2021.

[88] A. S. Kucik and G. Meoni, "Investigating spiking neural networks for energy-efficient on-board ai applications. a case study in land cover and land use classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2020–2030.

[89] M. Xu, L. Chen, H. Shi, Z. Yang, J. Li, and T. Long, "Fpga-based implementation of ship detection for satellite on-board processing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 9733–9745, 2022.

[90] S. Wiehle, D. Günzel, and B. Tings, "Sar satellite on-board ship, wind, and sea state detection," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. IEEE, 2021, pp. 8289–8292.

[91] R. Pitonak, J. Mucha, L. Dobis, M. Javorka, and M. Marusin, "Cloudsatnet-1: Fpga-based hardware-accelerated quantized cnn for satellite on-board cloud coverage classification," *Remote Sensing*, vol. 14, no. 13, p. 3180, 2022.

[92] M. Elmahallawy and T. Luo, "Asyncfleo: Asynchronous federated learning for leo satellite constellations with high-altitude platforms," in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 5478–5487.

[93] ——, "Fedhap: Fast federated learning for leo constellations using collaborative haps," in *2022 14th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2022, pp. 888–893.

[94] F. Valente, V. Eramo, and F. G. Lavacca, "Optimal bandwidth and computing resource allocation in low earth orbit satellite constellation for earth observation applications," *Computer Networks*, p. 109849, 2023.

[95] F. Valente, F. G. Lavacca, and V. Eramo, "Proposal and Investigation of a Processing and Bandwidth Resource Allocation Strategy in LEO Satellite Networks for Earth Observation Applications," in *ICC 2023-IEEE International Conference on Communications*. IEEE, 2023.

[96] V. Eramo and F. G. Lavacca, "Proposal and investigation of a reconfiguration cost aware policy for resource allocation in multi-provider nfv infrastructures interconnected by elastic optical networks," *Journal of Lightwave Technology*, vol. 37, no. 16, pp. 4098–4114, 2019.

[97] A. J. Alqaraghuli, H. Abdellatif, and J. M. Jornet, "Performance analysis of a dual terahertz/ka band communication system for satellite mega-constellations," in *2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2021, pp. 316–322.

[98] ESA. Sentinel-2 Operations. Available at https://www.esa.int/Enabling_Support/Operations/Sentinel-2_operations (accessed October 23, 2023).

[99] Amazon. Amazon Web Services Ground Station Locations. Available at https://aws.amazon.com/ground-station/locations/ (accessed October 23, 2023).

[100] F. Valente, F. G. Lavacca, and V. Eramo, "A resource allocation strategy in earth observation orbital edge computing-enabled satellite networks to minimize ground station energy consumption," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2023, pp. 1–6.

[101] F. Valente, F. G. Lavacca, M. Polverini, and V. Eramo, "A resource allocation strategy in orbital edge computing earth observation satellite constellations to jointly save energy on ground and balance on-board energy consumption," in *Proceedings of the International astronautical congress, IAC*, 2023.

[102] ESA. Copernicus: Sentinel-2. Available at https://www.eoportal.org\/satellite-missions\/copernicus-sentinel-2 (accessed October 23, 2023).

[103] R. L. Becerra and C. A. C. Coello, "Solving hard multiobjective optimization problems using ε-constraint with cultured differential evolution," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2006, pp. 543–552.

[104] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[105] P. Helber, B. Bischke, A. Dengel, and D. Borth, "Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification," in *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, 2018, pp. 204–207.

[106] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "On-board federated learning for satellite clusters with inter-satellite links," *arXiv preprint arXiv:2307.08346*, 2023.

[107] N. Sonune. Land Cover Classification with EuroSAT Dataset. Available at https://www.kaggle.com/code/nilesh789/land-cover-classification-with-eurosat-dataset (accessed October 23, 2023).

[108] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.