AI TOOLS FOR DESIGN AND OPERATION OF DISTRIBUTED SPACECRAFT MISSIONS

A Dissertation

by

PAU GARCIA BUZZI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Daniel Selva
Committee Members,	Daniele Mortari
	Dileep Kalathil
	Nancy J. Currie-Gregg
Head of Department,	Srinivas Rao Vadali

August 2021

Major Subject: Aerospace Engineering

ABSTRACT

With the recent advances in satellite miniaturization, communication and information technologies, there has been a paradigm shift in space exploration missions over the last few decades. This paradigm shift involves the transition from monolithic architectures formed by just one big satellite to a concept of a sensor web for space exploration consisting of heterogeneous sensors hosted on a variety of platforms including space, air and ground assets. These multiple entities share information in real time and make coordinated autonomous decisions to maximize system performance and/or scientific value. In this context, this thesis uses AI and machine learning techniques to overcome two big challenges found in the design and operation of Distributed Spacecraft Missions (DSMs): (1) The combinatorial explosion of feasible Earth observing constellations when not constraining the satellite orbits to symmetrical configurations, such as the Walker pattern. (2) The constant monitoring and ground operations required for node buffer management in Delay Tolerant Networks (DTN), which are governed by a set of standardized internet-like communications protocols robust to long delay and constant disruptions, and used in the communication between nodes in DSMs. The first challenge is approached by creating novel evolutionary formulations to explore large tradespaces of non-Walker hybrid satellite constellations with diversity of orbital parameters. Finally, the second challenge is addressed with the use of deep reinforcement learning, to automate the on-board decision making process in certain aspects of memory buffer management in DTN nodes, with the ultimate goal of optimizing network performance and reducing operational costs.

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor Prof. Daniel Selva for the guidance and support over the last 5 years working together. Dani will always be my role model and I will forever be thankful for everything he has taught me. In our one-on-one weekly meetings, I learned everything I know today about designing space systems and how to do research. I also want to thank Dani for the countless hours revising my papers and for making me a better writer. Dani not only gave me good scientific advise, but also was there to listen and provide solutions to my life problems. Dani made me a much better engineer and researcher, that's for sure, but he also showed me the best ways to be a good team leader. He also offered me the opportunity to connect with NASA through my research and helped me get two summer internships inside the agency: one in Goddard Spaceflight Center and another one in the Jet Propulsion Laboratory (even though this last one had to be done online because of the COVID-19 pandemic). These were both unforgettable experiences and truly a dream come true. Thank you for this and more, Dani, I really hope we can work together again in the future.

I would also like to thank all the members in my committee. Prof. Mortari, Prof. Currie-Gregg and Prof. Kalathil, thank you so much for taking the time to attend all the committee meetings and for providing me with insightful ideas to better shape my research.

I also need to thank Marc Sanchez Net from JPL for being such an amazing co-PI in the RL-DTN project and for offering the opportunity to be a Summer 2020 intern in the laboratory. I really enjoyed the research we carried out together and learned a lot from you about space communications and delay tolerant networking.

I also want to thank all my current and former lab mates Nozomi, Nathan, Harris, Samalis, Antoni, Prachi, Filipe, Roshan, Alan, Gabe, Ben S, Ben G, Ada-Rhodes and Ashish.

Thank you all for the time we spent together inside and outside the lab and for listening and critiquing my work during the group meetings. I want to especially thank Samalis (and Markus) for those bike rides together, Antoni for always offering help and for being a good roommate, and Gabe for sharing the best PhD memories with me in College Park and College Station.

During my Master's degree in Ithaca, I made very special friends who kept supporting me all throughout the PhD. Thank you Coach, Greg, Cisco and My for making every day better. Thank you to my friends from Barcelona, especially Xavi and Joan, for keeping in touch with me on a weekly or even daily basis.

I also want to thank my girlfriend Sarah for her love and support. Thanks to you and the little kitten, the last year of my PhD has been the best one (even on a global pandemic!). You have given me a very much needed emotional stability and showed me what feeling truly loved means. I am looking forward to many more laughs and adventures together. I will support you and give you the best advice I can during the last year and a half of your PhD.

Finally, I am beyond grateful for my parents. They have always supported me all throughout this journey, even though pursuing graduate studies in the US entailed being far from each other. Thank you for always being there for me and for offering me emotional support when I really needed it. Thank you for always giving me everything I needed without even asking for anything in return.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Professors Daniel Selva (advisor), Daniele Mortari, and Nancy J. Currie-Gregg of the Department of Aerospace Engineering and Professor Dileep Kalathil of the Department of Electrical and Computer Engineering.

All the work conducted for the dissertation was completed by the student independently, under the guidance and supervision of Prof. Daniel Selva.

Funding Sources

The work presented in Chapter 2 was funded by NASA's Goddard Space Flight Center under research grant agreement number 80NSSC18K1640, to develop a tool called Tradespace Analysis Tool for Constellations using Machine Learning (TAT-C ML).

The research presented in Chapter 3 was carried out (in part) at the Jet Propulsion Laboratory (JPL), California Institute of Technology, under a contract with NASA and funded through JPL's Strategic University Research Partnerships program.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Motivation

In the last few decades, with the recent advances in satellite miniaturization [1], communication and information technologies [2], there has been a paradigm shift in space exploration [3, 4]. This paradigm shift involves the transition from monolithic architectures formed by just one big satellite to a concept of a sensor web for space exploration consisting of multiple and generally heterogeneous sensors hosted on a variety of platforms including space, air and ground assets [5]. These multiple entities share information in real time and make coordinated autonomous decisions to collaboratively solve a set of tasks and maximize scientific return. The ultimate goal of deploying a set of smaller distributed entities collaborating together is to generate a more complete data product than the one provided by just a massive single satellite. Therefore, a sensor web can be defined as a coordinated observation infrastructure composed of a distributed collection of resources that can collectively behave as a single, autonomous, taskable, dynamically adaptive and re-configurable observing system that provides data via a set of service-oriented interfaces [5]. Multiple space agencies such as the National Aeronautics and Space Administration (NASA) and the European Space Agency (ESA) have sponsored numerous projects whose main objective is to contribute to the concept of sensor web for Earth Observation (EO) with several applications[6, 7, 8, 9]: land-use change, ecosystem dynamics, disaster monitoring, agriculture and sustainability, biodiversity, advanced weather forecasting, climate monitoring, and public health.

Distributed Spacecraft Missions (DSM) that fly multiple simpler and cheaper satellites to provide increased capabilities such as better temporal, spatial, and angular sampling are gaining a lot of popularity [10, 11] in this new concept of sensor web for EO. The recent popularization of standards such as the CubeSat specifications [12], the development of smaller

and inexpensive payloads and satellite bus components, and the continuous increase of more economical launch options [13, 14, 15, 16] are creating new opportunities for the real implementation of DSMs. A few examples of recent DSM proposed by national space organizations, industry and academia are the following:

- NASA-funded Cyclone Global Navigation Satellite System (CYGNSS) [17], which uses a constellation of 8 micro satellites deployed by a single launch vehicle into a 500km altitude orbit. The mission objective is to better understand the interactions between the sea and the air near the core of tropical cyclones. Previous monolithic missions, such as ASCAT and QuikSCAT, have a mean revisit time of over a day in the tropical regions. With a fleet of 8 satelites, CYGNSS decreases the mean revisit time coverage metric to 4 hours, allowing the system to not only predict a storm's evolution, but also to accurately model storm cyclogenesis and intensification.

- The Time-Resolved Observations of Precipitation structure and storm Intensity with a Constellation of SmallSats (TROPICS) mission [18], which will be launched within the next year. This NASA Earth Venture Instrument mission aims to monitor the thermodynamics of the troposphere and the precipitation structure for storm systems over the tropical regions. To do so, TROPICS will use a fleet of several LEO dual spinning 3U-CubeSats, each hosting a payload consisting of a 12-channel high-performance millimeter-wave radiometer that provides different measurements such as temperature and water vapor profiles, imagery for precipitation quantification, and cloud ice measurements. The TROPICS mission is designed to achieve median revisit times of 60 minutes or less in the tropical regions.

- ESA QB50 [19]. This Earth Science mission consists of a fleet of 50 CubeSats (most of them launched from the Iternational Space Station and the others using the PSLV Indian Rocket) in a string-of-pearls configuration and carrying identical sensors. QB50

has the mission objective of studying the temporal and spatial variations of a number of key parameters in the lower thermosphere. This mission also studies the re-entry process by comparing predicted and actual CubeSat trajectories and orbital lifetimes.

- New Commercial ventures such as:

    1. Planet Labs [20, 21], whose goal is to provide medium-to-high resolution imaging of the entire planet on a daily basis and deliver Earth imagery and post processed data products to markets such as agriculture, defense, emergency response, and insurance.
    2. Capella Space [22], a company which uses satellites equipped with synthetic-aperture radars to create services of hourly, reliable, and persistent imagery of anywhere on the planet for the U.S. government and other commercial customers.
    3. OneWeb [23], which uses a constellation of high-frequency transmitting and receiving satellites, placed in circular Low Earth Orbit (LEO), at approximately 1,200 km of altitude, to provide global satellite Internet broadband services to people anywhere in the world.
    4. SES [24], a company based in Luxembourg that provides connectivity services to customers in markets including telecommunications, cloud computing, commercial air and shipping,holiday cruises, energy, mining and end users in remote locations. Through the O3b mPOWER constellation, a fleet of several high throughput satellites places in Medium Earth Orbit (MEO) at approximately 8,000 km of altitude, SES aims to provide low latency scalable-based communications.

The proliferation of DSMs for space exploration does not come free of challenges. Distributed configurations, despite providing higher science benefit and more complete data

products than monolithic ones, require a much more complicated system architecture design process [25, 26]. Indeed, transitioning from one to n satellites causes the combinatorial explosion of feasible constellation designs, and searching the optimal multi-satellite orbital configuration requires significantly higher computational resources. In fact, the number of possible constellation designs grows at an exponential rate with the number of satellites n forming the constellation, and the new tradespace contains in the order of millions –or even billions– of different feasible designs. To ease the search of constellation designs that satisfy mission requirements, a common approach is to restrict the constellation configuration to simple symmetrical patterns (e.g. Walker constellations) [27] that generally provide good performance (or science benefit/return) at a reasonable search expense. However, this simpler constellation designs could be surpass in both performance and cost by other more complex configurations that do not follow these simple patterns [28, 29]. The first big chapter of this thesis aims to use novel Evolutionary Algorithms (EA) formulations to explore large tradespaces of non-Walker hybrid satellite constellations with diversity of orbital parameters. Particularly, Multi-Objective Evolutionary Algorithms (MOEA) [30] are used since there generally exist multiple conflicting objectives to optimize. Indeed, attempting to formulate an equivalent single-objective optimization problem through, for instance a weighed average metric, is challenging because it is hard– and sometimes impossible–to know a priori the preferences between all these objectives. For instance, in the constellation design problem for Earth Observation missions, in addition to the trade-off between the cost and coverage performance, there may also be more precise trade-offs between different coverage figures of merit, such as percent coverage vs. revisit time, or mean vs. maximum revisit time [31, 32]. Furthermore, MOEAs offer the ability to deal with mixed-integer problems with highly nonlinear and non-convex objective functions, such as the numerical simulations needed to evaluate coverage performance for satellite constellations.

Another challenge encountered in DSMs is the persistent need of continuous ground op-

erations to ensure a correct system functioning, which translates into higher mission costs [33]. Therefore, significant effort over the past few years has gone into developing new on-board AI tools to automate some aspects of the decision making in DSM technologies [34, 35, 36]. One of the technologies that requires constant monitoring and management from ground are DSM network communications, which are subject to long delays, continuous disruptions and power constrains. To deal with all these challenges, Delay Tolerant Networking (DTN) protocols [37] were developed to complement other well-established communication protocols such as TCP and IP, which generally assume that all network links are stable and reliable. While the core protocols of DTN are well understood, management of this technology that enable Internet-like connectivity across network entities in the Solar System is still an area of active research. The second big chapter of this thesis aims to use AI to automate buffer monitoring and management of DTN nodes [38, 39]. Memory management is critical in space communications, since DTN protocols are based on a store-and-forward approach by which all intermediate nodes store their received data in the form of bundles in their buffers, until the next hop in the transmission path becomes available. This approach requires DTN nodes to have enough memory available at all times to avoid the loss of packets of information. For this purpose, there currently exists a need of developing intelligent on-board tools to assist in the management of large and complex DTNs, with the ultimate goal of optimizing network performance and reducing staffing and operational costs.

1.2 Background

1.2.1 Distributed Spacecraft Mission Tradespace Exploration

The design of most systems found in the real world is extremely complex and it involves trade-offs between different conflicting metrics or figures of merit [40]. A tradespace [41] is a set of different designs or architectures located in the space defined by 2 or more metrics.

The goal of tradespace exploration is to study the underlying relationship between these multiple metrics, understanding how changes in system design variables affect each of the metrics, and determining the best system architectures or designs. Tradespace exploration insights are used to inform feasibility studies, trade studies and what-if analyses during early phases of the design process (Pre-Phase A studies for Space Missions [42, 32]). Indeed, when designing complex systems with more than one objective, there is no unique optimal solution. Instead, there is a set of optimal designs or Pareto Front [43], whose size grows exponentially with the number of objectives under consideration. It is worthwhile to note that, generally, attempting to formulate a multi-attribute decision making problem a priori is challenging because it is hard, if not impossible, to know a priori the preferences between all these attributes or metrics. For example, when designing a weather forecasting mission, ideally we would like to run an Observing System Simulation Experiment for each possible design to see how different values of spatial resolution, temporal resolution, and accuracy combine to improve weather forecasting accuracy or a similar high-level parameter –but that is of course not possible. Thus, it is sometimes desirable to adopt a more human-driven process, in which the main trades and alternatives are discovered at the same time as decision maker preferences are elicited.

A common approach to perform tradespace exploration for DSMs involve the organization of concurrent design sessions [44, 45, 46, 47] which gather together a few experts in multiple disciplines or fields (generally an expert for each of the different satellite subsytems: Payload, ADCS, Communications, Thermal, Avionics, Power and Propulsion). During these sessions, a few candidate initial missions are defined and evaluated and, after a few design iterations, the expert designers come up with the best mission architecture that satisfies certain requirements and meets the stakeholder needs. During this collaborative design process, new knowledge is learned and used in future sessions. This expertise is manifested in the form of design heuristics and rules which, together with the use of effective communication, can help

the team moving faster towards the most interesting areas of the tradespace. This strategy, however, usually has the only objective of designing a mission that satisfies certain requirements and has the risk of missing other less intuitive unexplored areas of the tradespace that could potentially be more cost-efficient. Also, planning these concurrent design sessions is hard and expensive, specially due to the busy schedules from the experts, and the need of having a dedicated facility to promote this interactive design process.

There is another approach extensively used in the literature which does not require as many human resources, expert designers with years of experience in multiple disciplines, or a physical space to perform tradespace exploration for the design of new DSMs. This other approach consists of using Multi-Objective Evolutionary Algorithms (MOEA) and use a computer or cluster to explore this extensive tradespace of DSMs [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 52, 65]. Unlike single-objective optimization methods, MOEAs keep each objective separate and produce a Pareto front of optimal designs that helps later analyze the trade-offs between the different conflicting objectives under consideration. While this second approach requires little design expertise and has the ability of exploring millions of missions (including seemingly unpromising solutions through stochastic sampling), it does not come for free. Indeed, MOEAs for tradespace exploration can be computationally inefficient since they generally rely on evaluating many non-interesting designs before discovering a set high-quality system architectures.

1.2.1.1 Multi-objective Optimization Problems

The mathematical formulation of an optimization problem that involves more than a single objective [30] is the following:

$$\begin{aligned} \underset{\vec{x}\in X}{\text{minimize}} \quad & F(\vec{x}) \\ \text{subject to} \quad & g_i(\vec{x}) \leq b_i,\ i = 1, \ldots, P, \\ & h_j(\vec{x}) = b_j,\ j = 1, \ldots, Q, \end{aligned}$$

where $F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \ldots, f_m(\vec{x}))$ is a vector of size m which contains the m different objective functions to be optimized simultaneously, and $g_i(\vec{x})$ and $h_j(\vec{x})$ are the ith inequality and jth equality constraints, respectively. Finally, $X \subseteq \mathbb{R}^N$ corresponds to the problem decision space and the vector $\vec{x} \in X$ contains the N continuous or discrete design variables to be optimized.

Multi-objective optimization can be used to solve non-trivial real world problems which involve optimizing multiple conflicting objectives, and where preferences among these objectives are indeterminable. Therefore, a utility function can not be defined and the different objectives need to be dealt with separately. A few examples are the following:

- DSM design, which involves finding the best system architectures (constellation orbital parameters, on-board instruments, bus characteristics, and launch vehicle), which satisfy many nonlinear constraints involving packaging space, power requirements, communication link budgets and thermal control, while optimizing several conflicting objectives including the maximization of performance and the minimization of other metrics such as cost and risk.

- Car design, which consists of finding the best vehicle configurations (frame, wheels and tires, motor, etc.) that minimize cost and fuel consumption while maximizing comfort and horse power, and satisfying certain emission levels constraints.

- Portfolio optimization, which a part from maximizing expected value of the return, it also seeks to minimize the variance of the return, which is commonly associated with

risk. There are many other finance and economics applications that can benefit from efficient algorithms capable of solving multi-objective problems.

The main objective of multi-objective optimization is to present the designers a diverse set of architectures, known as Pareto Front, that optimally trade the different objectives under consideration, allowing the designers to develop their own preferences based on the presented solutions and their trade-offs. In fact, for any non-trivial multi-objective problem (and by non-trivial we mean a problem with 2 or more conflicting objectives) there is no single solution that simultaneously optimizes all objectives. In that case, the Pareto optimal solutions are said to be non-dominated and they can be numerous, especially when the number of objectives increases. Solutions are non-dominated if none of the objective functions can be improved in value without degrading some of the other solution objective values. Mathematically, given two solutions with objective vectors F and G, F is said to dominate G if and only if $\forall k \in \{1, 2, ..., m\} F_k \leq G_k$ and $\exists l \in \{1, 2, ..., m\}$ such that $F_l < G_l$. Without additional designer preference information, all non-dominated solutions are considered equally good and need to be presented to the decision-maker as Pareto optimal solutions. Nevertheless, exactly determining the truly optimal set of solutions (or true Pareto Front) is very challenging, so the ultimate goal when solving a multi-objective problem is to obtain a set of solutions whose Pareto Front approximates to the maximum extent possible the true Pareto Front.

1.2.1.2 Multi-objective Evolutionary Algorithms

Evolutionary Algorithms (EA) are one particular kind of heuristic and meta-heuristic optimization algorithms. They are widely used in many scientific and engineering communities for their flexibility and their ability to deal with the two following issues:

1. Complex objective functions and constraints. Many real-world optimization problems are very complex [66, 32] and involve large numerical simulations to model

highly non-linear, non-differentiable and non-convex objective functions and constraints. Therefore, other gradient-based optimization methods can not be applied unless some assumptions are made, such as the use of approximation methods to simplify (e.g. linearization, surrogate modeling) these complex objective functions. EAs can handle such complicated physical models with unfavorable mathematical properties.

2. Multiple objectives. EAs can assist in solving problems involving the simultaneous optimization of several conflicting objectives by means of Multi-objective Evolutionary Algorithms (MOEA) [30].

Evolutionary algorithms essence lies in natural evolution. The first step involves defining a *chromosome*, which encapsulates the design variables (or the DNA of the system being optimized) and corresponds to an instance or solution of the system architecture. A set of chromosomes/solutions creates the population, which continuously evolves throughout the optimization process. Throughout this process, the genetic information of several chromosomes is crossed/recombined by means of *operators*, which mimic asexual and sexual reproduction, to create new child solutions or offspring that enter the population. In order to converge to a high quality population, EAs use selection operators to choose which solutions should remain and which should be eliminated from the population. A high quality population is not only formed by solutions that offer good objective values, but also maintain diversity between solutions to prevent the algorithm from getting stuck in a local optima. EAs can effectively solve either unconstrained and constrained problems, and both soft and hard *constraints* can be defined.

Despite being very flexible and versatile, EAs can be computationally inefficient [67]. Since they use stochastic sampling to explore the tradespace of feasible solutions, it usually requires many function evaluations to identify high-quality solutions. Choosing the right evolutionary formulation (i.e., objectives, chromosomes, operators and constraints) can help

increase the algorithm's efficiency, since the problem formulation is at least as critical in driving performance as the details of the optimization algorithm used itself. The second chapter of this thesis will focus on creating new evolutionary formulations to tackle the exploration of asymmetrical hybrid constellations for Earth observation DSMs.

1.2.1.3 The constellation design problem for Earth observing DSMs[*]

The satellite constellation design problem for Earth Observation (EO) DSMs is very complex and involves several orbital interrelated design variables and multiple conflicting objectives [32]; On the one hand, the decision variables generally correspond to the 6 orbital parameters –altitude, inclination, eccentricity, longitude of the ascending node, argument of perigee and mean/true anomaly– for each of the n satellites forming the constellation, leading to a total number of $6n$ optimization variables. For Earth Observing constellation, however, elliptical orbits are usually not considered, thus reducing the number of orbital parameters for each satellite to 4 (since eccentricity is equal to 0 for circular orbit and the argument of perigee is undetermined). To the orbital parameters, some formulations add other decision variables to the decision vector, such as instrument/payload or launch vehicle selection when a few different options are available to the designer; On the other hand, the orbit and constellation design process for Earth observation missions involves trades between several conflicting objectives such as coverage performance, cost, robustness, and mission lifetime. Among all these objectives, one of the most important ones is coverage performance, which can be assessed through multiple coverage metrics. Coverage metrics quantify how well the constellation "covers" the surface of the Earth with its observations [31]. Coverage metrics are usually calculated on a grid of points on the surface of the Earth, by propagating the different spacecraft that compose the constellation for a certain simulation

[*]This section is reprinted with permission from "Assessment of constellation designs for earth observation: Application to the TROPICS mission" by Pau Garcia Buzzi, Daniel Selva, Nozomi Hitomi and William J. Blackwell, 2019. Acta Astronautica, Volume 161, pp. 166-182 Copyright 2019 by IAA. Published by Elsevier Ltd

time T. During the propagation, the access time intervals in which coverage grid points are seen by any of the satellites are computed, when considering the field of view and imaging concept of the sensors, and any viewing geometry constraints, such as those on incidence angle. When ordering this list of time intervals, $(ts_{ki,n}, te_{ki,n})$ indicates the start and end time respectively for the nth access of satellite i to coverage grid point k. Often, the satellite(s) that perform(s) the access are irrelevant for the calculations, and thus the subindex i can be ignored. For every point on the grid k, a coverage gap is the interval of time between the end of an access n between the point k and any satellite in the constellation, and the start of the next access $n+1$: $tg_{k,n} = ts_{k,n+1} - te_{k,n}$. All coverage metrics are calculated from statistics of the access and gap intervals. These statistics are calculated for each point in the coverage grid, but can be aggregated (e.g., averaged out) for all points at a given latitude (e.g., to obtain a chart of average revisit time vs latitude), or for all points in the coverage definition (to obtain, for example, a single average revisit time number for the constellation to use as objectives in optimization problems).

Because there are many different ways to aggregate information from points in the coverage grid, and many statistics that can be calculated, many different coverage metrics have been defined. Moreover, it is unclear a priori which metrics are best for a given mission. The following are some of the most widely used metrics in the literature [31, 62, 68] – all of them are defined for a single point k on the grid:

- **Descriptive statistics**. Minimum, maximum, median, mean, variance and different percentiles of access and/or gap interval duration for point k. The most important ones are: (1) mean coverage gap, also known as mean revisit time, which is the average length of the gap intervals for point k, and the most common metric used in coverage analysis by far; and (2) maximum revisit time, also known as maximum gap time, which corresponds to the longest gap interval for point k, and is also popular as it provides worst-case information.

- **Percent Coverage (PC)**. Total time during which point k is accessed by at least one satellite in the constellation divided by the total simulation time:

$$PC_k = 1 - \frac{\sum_n tg_{k,n}}{T}$$

where T is the simulation time and $tg_{k,n}$ is the nth gap time for point k in the coverage grid, as defined above.

- **Mean Response Time**. Response time is defined as the time from when a random request is received to observe a point k until the constellation can actually observe it. Note that response time is a function of time. If at a given time t, the point k is being accessed by the constellation, i.e., $ts_{k,n} \leq t \leq te_{k,n}$ for some n, then the response time at that time is zero ($R_k(t) = 0$). If the point k is in a coverage gap at time t, i.e., $te_{k,n} \leq t \leq ts_{k,n+1}$ for some n, then the response time is the time until the end of that gap, i.e., until the point is accessed again: $R_k(t) = ts_{k,n+1} - t$. Thus, mean response time is defined as the time average of the response time.

$$R_k = \frac{1}{T}\int R_k(t)dt = \frac{\sum_n tg_{k,n}^2}{2T}$$

- **Time Average Gap**. For a given point of the grid k, it corresponds to the time average of the mean gap duration, which is also a function of time. This FOM is very similar to mean response time because the function being averaged (integrated) is the length of the current gap at every time instant $G_k(t)$, which is 0 in the case of the point being accessed at time t and equal to $tg_{k,n}$ otherwise. Time average gap can be obtained by multiplying the mean response time by a factor of 2, since the area under the curve we are integrating now corresponds to sum of the area of the rectangles whose area is

twice the one of the triangles on the mean response time calculation.

$$G_k = \frac{1}{T}\int G_k(t)dt = \frac{\sum_n tg_{k,n}^2}{T}$$

One issue with some of the above metrics is that they can be biased or misleading for gap duration distributions with many short gaps and a few very long gaps – typical of string-of-pearls constellations, which are popular constellation designs for CubeSats, since they are advantageous in terms of launch cost. In most applications, many successive short gaps during coverage periods do not compensate for a few very long gaps between coverage periods. Thus, using a simple mean or median of all gap durations as a FOM in these cases may lead to overly optimistic results, especially when the number of satellite grows. In Figure 1.1, two Cumulative Distribution Functions (CDF) of the revisit times for 2 very different constellations are shown. In constellation 1, 6 satellites are put into the same orbital plane, whereas in constellation 2, these 6 satellites are distributed in 6 different planes equally spaced in Right Ascension of the Ascending Node (RAAN). We can observe that only looking at mean revisit times, both constellations would appear to be very similar. However, for constellation 1, 90% of gaps are less or equal to 16 mins, and the only few very long gaps of nearly 1000 mins bias their mean statistics towards the ones corresponding to constellation 2, which has fewer very short gaps but they are all shorter than 156 minutes.

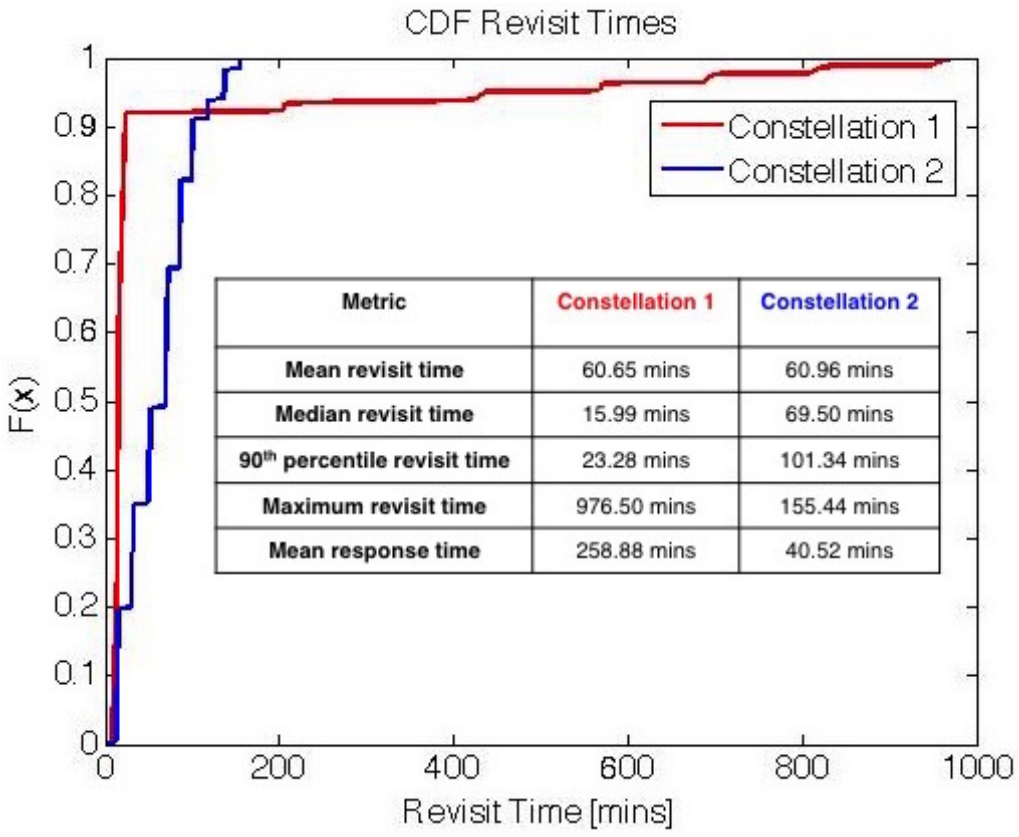
For this reason, another metric was proposed in a TROPICS coverage study that had not been described in the literature, which the team called Continuous High Revisit Coverage (CHRC) [32]. This metric is defined as the percentage of time where point k is either in an access, or in a gap shorter than a threshold gap duration t_{hold}.

$$CHRC_k = 1 - \frac{\sum_n (tg_{k,n} \geq t_{hold})}{T}$$

For some applications, gaps shorter than some threshold may not be important. For

instance, if the satellite data is to be assimilated in a weather model that has a time step of 1 hour, gaps shorter than 1 hour may not affect the output of the model, so one could argue that they should be ignored. This new metric ignores short gaps and thus can better account for the importance of those long gaps left, for example, by string-of-pearls constellations.

Such coverage metrics need to be traded against cost and risk during Pre-Phase A and Phase A studies to determine the number of satellites required and their orbital characteristics. In addition, there are other important considerations that must be taken into account in that decision, including the deployment strategy for the constellation [69, 70, 71,



Metric	Constellation 1	Constellation 2
Mean revisit time	60.65 mins	60.96 mins
Median revisit time	15.99 mins	69.50 mins
90th percentile revisit time	23.28 mins	101.34 mins
Maximum revisit time	976.50 mins	155.44 mins
Mean response time	258.88 mins	40.52 mins

Figure 1.1: Comparison of the coverage metrics for 2 constellations. Constellation 1 (6 satellites distributed in 1 plane) has a lot of very short gaps and only a few very long gaps. Constellation 2 (6 satellites distributed in 6 planes) has fewer very short gaps and their duration is more balanced with no very long gaps.

72, 73, 74, 75, 76, 77], the robustness of the constellation to satellite and launch failures [78, 79, 80, 81, 82, 83, 84], as well as mission lifetime and deorbiting [85].

1.2.2 Distributed Spacecraft Mission Operations

The transition from monolithic to distributed spacecraft missions not only added more complexity to the mission's design/development phase, but also to its operational phase. Despite offering better scientific value, maintaining a synchronized fleet of spacecraft in orbit requires more ground and/or software resources than operating a single-satellite architecture. In fact, [11] proposes a new taxonomy for DSMs in which one of the characteristics defined is the level of autonomy, which ranges from *Ground-based controlled mission execution* to *Full Autonomy* missions, including *Semi-Autonomy* systems in between which represent a hybrid combination of autonomous systems and ground control. Depending on the level of autonomy of a DSM, a lot of resources have to be allocated to either ground personnel (for missions whose operation/execution relies on decisions made on the ground in a centralized manner) or to the design of sophisticated on-board autonomous systems which take care of most of the decision making present during the operational phase of the mission. While the former is easier to implement in the short term, the latter could help decrease the amount of resources dedicated to ground workers (e.g. money spent on salaries) in the long term. Indeed, autonomy is a critical need for DSMs, since the cost associated with applying conventional approaches for command and control does not scale well with the continuously increasing number of satellites in the constellation [36].

In the last decades there has been a lot of research in developing new technologies and methodologies to enable the integrated navigation, communication and control of DSMs [34]. NASA Goddard Space Flight Center is putting a lot of effort into the technology development for intelligent and collaborative muti-satellite constellations, including early mission design tools, high-speed low-cost small satellite communications, autonomous guidance and control systems, ground "Big Data" data processing and on-board intelligent systems [6]. Similarly, the Bay Area Environmental Research Institute at NASA Ames Research Center

is leading a project to develop a tool called D-SHIELD to enable logistical decisions in DSM with the use of heuristic intelligence [35]. This tool consists of an intelligent scheduler that can be run from ground or on-board of the spacecraft, an observable science simulator and an operations tradespace analyzer, and has the ultimate goal of helping schedule payload operations of large constellations.

1.2.2.1 Satellite communications in space

Space communications, like any other sort of communication process, relies on three fundamental pieces: a transmitter, a channel and a receiver. The transmitter encodes a message or packet of information, modulates the encoded data on a certain band of electromagnetic frequency and sends the encoded modulated message through bursts of radio signals to an antenna. By means of electromagnetic waves, the message travels through the deep space channel towards the receiver. The receiver's antenna acquires the electromagnetic waves, demodulates the signal and, finally, decodes and recovers the transmitter's message. The NASA's Space Communications and Navigation (SCaN) program [86] enables the data exchange with space, including the communication with astronauts aboard the International Space Station (ISS), rovers exploring the surface of Mars, or the Artemis missions dedicated to landing "the first woman and the next man" on the lunar south pole region by 2024. NASA has an extensive network of antennas, called the Deep Space Network (DSN) [87], spread around Earth to communicate with satellites in orbit. These ground station antennas range from the small very high frequency antennas that provide backup communications to the ISS to massive antennas that can communicate with very far deep space exploration missions. Network engineers carefully schedule communications between ground stations and operating missions, ensuring that antennas are ready to receive data as communication with flying satellites can be established. In addition to direct-to-Earth communications, many NASA missions rely on relay satellites in order to download data to Earth. That being said, com-

municating to, from and within space is not an easy task and represents an endeavor full of challenges, which are listed below [88, 89]:

- Extremely long distances. In telecommunications, the free-space path loss is the attenuation of radio energy between two antennas that results from the obstacle-free, line-of-sight path through free space. Indeed, the ratio between the power received and the power transmitted is inversely proportional to the square of the communication path length. Thus, the extraordinarily large distances existing between deep space communication network nodes make path losses to be exceptionally large, posing a challenge very difficult to overcome.

- Power constraints. One of the main objectives when designing a satellite is minimizing cost which, in space, translates into designing a low mass system since the larger the satellite is the more expensive putting it into orbit is going to be. Therefore, all the bus components need to be small, including transceivers and antennas. Consequently, another challenge encountered in space communications is the limiting transmitting power and the small gain offered by satellite transmitters and antennas, respectively.

- Expensive bandwidth. Bandwidth is the frequency range occupied by a modulated carrier signal. Higher bandwidths can carry more bits of information per time unit, thus allowing spacecraft to transmit data faster. The technology used back in the Apollo radios only allowed to send grainy black and white video from the Moon. With the advances in technology, which among other things use a much wider frequency spectrum, the upcoming optical terminal on the Artemis II mission will be able to send 4K, ultra-high definition video from lunar orbit. Bandwidth in space is, however, very expensive and the frequency spectrum is strictly regulated by agencies such as the Federal Communications Commission (FCC). Therefore, communications in space are generally bandwidth constrained, making high-speed data communications sometimes impossi-

ble.

- Connectivity disruptions. Spacecraft move in space at very high speeds. For instance, the ISS, which orbits the Earth at an altitude of 400km, travels at 7.66 km/s (which translates into 27,724 kilometres per hour). This large node mobility causes constant line-of-sight losses between network entities in space, leading to a high intermittent communications. Relays can sometimes be a solution to deal with communication interruptions. For example, NASA deployed the Tracking and Data Relay Satellites (TDRS) at three smartly chosen regions above Earth, offering global coverage and near-continuous communications between LEO missions and the ground. Operating satellites, rather than waiting for line-of-sight and establish communication with a ground station, they can relay data to TDRS anytime. In addition to NASA's TDRS, Mars 2020 Perseverance rover will also send data through orbiters or relays around Mars, which will after forward the data to Earth.

- High latency (or long delays). Communications don't occur instantaneously. In fact they are bound by the speed of light (approximately 300,000 km/s), which corresponds to the universal speed limit. For communications between the ground and a satellite close to Earth, this time delay (or latency) is almost negligible. In deep space, however, distances are very long and time delays become significant (in the order of minutes or even hours). For instance, when Mars and Earth are at their greatest distance, the communication delay is around 24 minutes. Therefore, when astronauts eventually land on Mars, they will need to wait 24 minutes for their messages to reach mission control and vice versa.

- Noise and Interference. As electronic waves travel over long distances or through the atmosphere, the quality of the data deteriorates and noise is added to the original message [90]. Additionally, radiation from other missions, the Sun, or other celestial

bodies can also interfere with the quality of communications.

Currently, space communications rely primarily on radio waves, but significant effort is being put into developing ways to communicate with infrared lasers, which will offer missions higher data rates than ever before [91]. Big space agencies and private companies are developing novel technologies and capabilities that address the real-world challenges of space communications, while empowering science and exploration missions with robust communications services.

1.2.2.2 Delay Tolerant Networking

Delay Tolerant Networks (DTN) [37] offer the possibility to communicate between nodes in a network where constant connectivity between entities cannot be guaranteed due to loss of line-of-sight, long delays, or other considerations. Other well-established communication protocols such as TCP and IP generally assume that the underlying network links are stable and reliable. Similarly, large amounts of overhead are acceptable since transmission rates are large and bandwidth is cheap to provision. Unfortunately, neither of these characteristics is typically encountered in space, particularly for deep space exploration, where large amounts of data need to be returned to Earth over vast distances and power-constrained links. Therefore, the DTN protocol stack was originally developed to deal with connectivity disruptions and long delays, both of which lead to non-existence of continuous end-to-end paths between source and destination. To cope with these interruptions, DTNs use a store-and-forward approach by which all intermediate nodes store their received data in the form of bundles in their buffers, until the next hop in the transmission path becomes available. Moreover, to increase the Quality of Service (QoS) and probability of successful bundle delivery, DTN protocols can also send duplicates of the same piece of information through multiple paths, at the price of increasing the amount of energy, bandwidth and memory allocated per transmitted data unit.

Despite being a promising technology that can extend networking into connectivity challenged environments, there are still specific technology issues which need to be credibly addressed before DTNs can be reliabily deployed in space. Indeed, reaching high levels of network performance or QoS in DTNs is more challenging than in other more stable networks due to the low bandwidth, high interference, fading, lack of availability of new frequency spectrum and node mobility, which leads to highly unstable links [92]. Some of the network performance metrics that describe the level of QoS include indicators such as delivery ratio, packet drop, message overhead, latency among others. To help improve the performance metrics of this key enabling technology for the communication in future space exploration missions, the DTN community has investigated the development of new management solutions to deal with undesirable issues in DTNs such as congestion, selfishness, fairness, queuing delay and jitter.

1.2.3 Reinforcement Learning

1.2.3.1 What is RL?

RL interprets the world as a Markov Decision Process (MDP), with its corresponding set of states $x \in X$, set of actions $a \in A$ and reward function $R(x, a)$. The key difference between classic dynamic programming and model-free RL is that the system dynamics or the transition probability matrix from a state x to a state x' given a certain action a, $P\left(x' \mid x, a\right)$, are unknown in RL and, therefore, they need to be learned from experience and observations (i.e., sequences of states, actions and rewards). The goal of RL is to find the best control policy $\pi^*(x)$ that specifies the action to take in each state of the system. This optimal policy is defined as the one that maximizes the value function $V_\pi(x)$ or, equivalently, the one that maximizes the optimal Q-Value function $Q^*(x, a)$:

$$\pi^*(x) = \underset{\pi}{\operatorname{argmax}} V_\pi(x) = \underset{a}{\operatorname{argmax}} Q^*(x, a). \tag{1.1}$$

Given a policy π, the value function $V_\pi(x)$ corresponds to the expected cumulative discounted reward obtained by starting from state x and following the policy π until the end of the simulation. Mathematically, it is computed in the following way:

$$V_\pi(x) = \mathbb{E}\left[R\left(x_0, a_0\right) + \gamma R\left(x_1, a_1\right) + \gamma^2 R\left(x_2, a_2\right) + \cdots + \gamma^t R\left(x_t, a_t\right) + \cdots | x_0 = x, a_t \sim \pi\left(\cdot \mid x_t\right)\right]$$
$$= R(x, \pi(x)) + \gamma \sum P(y \mid x, \pi(x)) V_\pi(y), \tag{1.2}$$

where γ is commonly known as the discount factor and takes values from 0 to 1.

On the other hand, the $Q_\pi(x, a)$ state-action value function is defined as the expected cumulative discounted reward by starting from state x, taking action a, and then following the policy π until the end of the simulation. It is computed in the following way:

$$Q_\pi(x, a) = \mathbb{E}\left[R\left(x_0, a_0\right) + \gamma R\left(x_1, \pi\left(x_1\right)\right) + \gamma^2 R\left(x_2, \pi\left(x_2\right)\right) + \cdots \mid x_0 = x, a_0 = a\right]. \tag{1.3}$$

Value and Q-value are related in the following way:

$$V_\pi(x) = Q_\pi(x, \pi(x)). \tag{1.4}$$

Therefore, the optimal value function $V^*(x)$ can be obtained from the optimal Q-value function $Q^*(x, a)$ in the following way:

$$V^*(x) = \max_a Q^*(x, a), \tag{1.5}$$

where $Q^*(x, a)$ is defined as:

$$Q^*(x, a) = R(x, a) + \gamma \mathbb{E}\left[V^*(y) | x, a\right]. \tag{1.6}$$

1.2.3.2 Q-Learning

Probably the most common RL algorithm is Q-learning. Using the Bellman optimality equation and the Bellman operator properties (fixed point, monotonicity, contraction mapping), together with the convergence proofs of Temporal Difference (TD) learning, Q-learning guarantees under certain circumstances that the agent can learn the Q-value function to eventually obtain the optimal policy. The problem is now reduced to estimating accurate values of the optimal Q-function for all state/action pairs. This is done in an iterative learning process (Q-learning) according to the following rule:

$$Q_{t+1}(x,a) = Q_t(x,a) + \alpha \left[R_t(x,a) + \gamma \max_{a'} Q_t\left(x,a'\right) - Q_t(x,a) \right]. \tag{1.7}$$

The main idea behind this iterative process is to find the TD between the predicted Q-value and its current estimated value. The parameter α corresponds to the learning rate and it is used to determine the impact of new information on the existing Q-value. The learning rate must satisfy Robbins-Monro conditions to guarantee the convergence of the Q-learning algorithm. There are many extensions to the Q-learning algorithm such as SARSA (online Q-learning algorithm) or Deep Q-learning. A thorough literature review of all available RL algorithms can be found in [93].

1.2.3.3 Deep Q-Learning

For relatively simple problems in which state and action spaces are small, the value and Q-value functions and policies can be represented in tabular form. For these simple problems, the Q-learning algorithm can efficiently learn and find the optimal policy. However, for environments or systems with a larger number of states and actions, Q-learning may not be able to obtain the optimal policy. Furthermore, memory might also become a limiting issue when storing a Q-table with one entry for each state-action pair, due to combinatorial explosion. Consequently, Deep Q-Learning (DQL) emerged [94] to solve this problem

using supervised learning techniques (in particular Deep Neural Networks) to approximate the Q-value function. In other words, a neural network trained to approximate the optimal Q-value function $Q^*(x, a)$ will take the current state $x \in X$ as an input and will output the probability of choosing an action $a \in A$ to maximize cumulative expected reward. With the DQL algorithm, the memory issue of storing a Q-table with all state-action pairs is solved, since the only memory required to store a NN is the one needed to store its weights, and there are many fewer neurons than state-action pairs. The DQL algorithm with Experience Replay and Fixed Target Q-Network shown in Algorithm 1 is the one used in this thesis.

Algorithm 1: Deep Q-Learning with experience Replay and Target Network

Initialize replay buffer E to its size N;

Initialize the Q-Network action-value function Q with random weights w;

Initialize the target Q-network action-value function $\hat{Q}$ with random weights $w' = w$;

for *episode=0,* M **do**

  for *t=1,* T **do**

    With probability ϵ select a random action a_t, otherwise select $a_t = \arg\max Q\left(s_t, a_t, w\right)$;

    Perform action a_t in the simulator and observe reward r_t and the new state s_{t+1};

    Store transition or new experience (s_t, a_t, r_t, s_{t+1}) in the replay memory E;

    Select a random minibatch of experiences of size L (s_i, a_i, r_i, s_{i+1}) from E;

    Update the weights w of the Q-Network Q by using stochastic gradient descent: $\left(r_i + \gamma \max_{a_{i+1}} \mathcal{Q}\left(s_{i+1}, a_{i+1}; \boldsymbol{w'}\right) - \mathcal{Q}\left(s_i, a_i; \boldsymbol{w}\right)\right)^2$;

    Every C steps reset $\hat{Q} = Q$;

  end

end

The experience replay mechanism [95] breaks the strong temporal correlation between observations by mixing more and less recent experiences for the Q-Network weights w updates. This is crucial since many popular Stochastic Gradient Descend (SGD) algorithms make the assumption of independent and identically distributed (i.i.d.) samples. Moreover, it uses rare experiences for more than just a single update to avoid rapid forgetting of these rare experiences which could be useful later on in the training process. On the other hand, a target Q-Network $\hat{Q}$ is included because in supervised learning, the target does not depend on the optimized parameters w. In Q-Learning, the target itself depends on the weights w of

the Q-Network, leading to oscillations and instabilities since the target moves as the weights w are updated. A solution to stabilize the algorithm is presented in [96], in which a separate target network is used and kept unchanged for multiple w updates. We refer the reader to [97] for a deeper dive into the details of other state of the art DQL algorithms.

1.3 General Problem Statement

There exist 2 ways to tackle the tradespace exploration of Earth Observation DSM constellation designs: through organizing concurrent design sessions that gather field experts who enumerate and evaluate a few interesting designs, or by means of multi-objective optimization. The former generally requires a lot of time and economic resources to just evaluate a few possible mission architectures, and the latter either constrains the constellation configuration to simple and symmetrical patterns (e.g. Walker constellation, Flower constellations, etc.) or fixes some orbital parameters (e.g. choosing a common altitude or inclination for all satellites forming the constellation) to significantly reduce the original size of the tradespace. This is due to the long numerical simulations to assess the coverage performance of a single design, which can be in the order of minutes when using high-fidelity propagators that include not only the physics of the two-body problem, but also other perturbations such as Earth harmonics, solar radiation pressure, third body interactions or atmospheric drag. For that reason, the use of optimization (and in particular MOEAs) is needed when solving the constellation design problem due to its ability to converge to good regions of the tradespace without having to run a full factorial enumeration and evaluation of designs which, indeed, is almost always intractable. Despite MOEAs having a lot of potential for the exploration of large tradespaces, the current evolutionary formulations that are found in the literature are not well suited for DSMs since, as just mentioned, they usually restrict the search to certain regions of the tradespace containing simple configurations. Consequently, these available formulations do not offer enough exploration of new, less intuitive and more innovative

designs. Therefore, there is the need of developing more sophisticated and efficient evolutionary formulations to be used during the early development phase of DSMs that have the capacity of finding the best constellation designs, including more complicated asymmetric multi-satellite configurations, in a relatively shorter amount of time.

Additionally, during the operational phase of DSMs, DTN protocols offer the possibility to communicate in the space environment thanks to their ability to deal with undesirable issues such as long delays, power limitations and continuous interruptions. However, one of the assumptions to ensure a proper functioning of these protocols rely on having enough buffer memory to store and forward information as packets arrive to the different nodes in the network. Currently, there are no on-board decision making tools to help monitor and manage buffers in DTN nodes. Instead, mission controls predicts the line-of-sight contact windows between the different network entities, including user terminals, orbital relays and Earth ground stations and plans and schedules accordingly the transmission of packets of information between the different nodes forming the mission. Thus, the persistent need of continuous ground operations in DTNs poses a problem to the operational phase of DSMs. To overcome this challenge and reduce cost of operations, there is the need of developing novel intelligent tools to help automate some aspects of the decision making in this key enabling technology for the communication of future deep space exploration missions technologies.

1.4 Approach and research goals

This thesis uses AI and machine learning techniques to overcome the two important challenges, which are found in the development and operational phases of DSMs, respectively. Specifically, we address each of the issues identified in the previous section as outlined in the following paragraphs:

- The combinatorial explosion of feasible Earth observing constellation designs when

exploring architectures that can follow more complicated patterns than other widely used symmetrical configurations, such as Walker constellations. This first challenge will be approached by creating novel evolutionary formulations to explore large tradespaces of non-Walker hybrid satellite constellations with diversity of orbital parameters. The idea behind this new formulations is that they not only allow to search through the space of base constellations –such as Walker formations, Sun-synchronous trains and string-of-pearls among others– but also combinations of multiple base constellations. By doing so, we open up the tradespace to cover a much broader and understudied set of constellation designs during Pre-Phase A and Phase A studies of Earth Observation space missions. The performance of the proposed formulations is evaluated in several case studies including the observation of symmetrical, asymmetrical, connected and disconnected regions of interest.

- The constant monitoring and ground operations required for the management of Delay Tolerant Networks (DTNs). This second challenge will be addressed by the use of deep reinforcement learning, to automate the on-board decision making process in certain aspects of node management in DTNs, with the ultimate goal of optimizing network performance and QoS while anticipating failures and avoiding memory overflows. More specifically, the proposed algorithm trains an intelligent agent that is in charge of deciding when to drop packets, when to change the data rate of the neighbor node links, when to re-route bundles to crosslinks, or when not to change any network parameter. The agent's goal is to maximize the bits received by the ground stations while minimizing the capacity allocated to all controlled links, and control the buffer utilization to avoid memory overflows. In order to assess the potential of using RL in DTN management in DSMs, the performance of the trained RL agent is benchmarked against other non-RL based policies.

1.5 Structure

The structure of this thesis is as follows:

Chapter 2 first provides a survey of the literature on existing evolutionary formulations to tackle the tradespace exploration of DSM designs for Earth Observation. After identifying the gaps in the literature, a new evolutionary formulation to tackle the exploration of hybrid constellation designs is proposed to fill the gaps. This new formulation is evaluated and benchmarked against other existing formulations by looking at its ability to attain high-quality solutions within fewer number of function evaluations and its capacity to create less intuitive and more innovative constellation designs.

Chapter 3 starts by providing the background of why memory management is fundamental to guarantee the correct operation of DTN protocols, as well as an extensive literature review of both buffer management strategies and reinforcement learning applications in communications and networking. Then, we introduce a novel AI methodology that utilizes reinforcement learning to automate the decision making process in the monitoring and management of buffer utilization in DTN nodes. The proposed methodology is tested in two different case studies including a lunar mission and an Earth observation mission. In both scenarios, the performance of the RL-based proposed methodology is benchmarked against other non-RL based policies including a policy based on rules designed by a DTN expert.

Chapter 4 provides a summary of the thesis and its contributions to the literature. Limitations of the proposed work are addressed and opportunities for future work are discussed.

2. EVOLUTIONARY FORMULATIONS FOR DESIGN OF HYBRID EARTH OBSERVING CONSTELLATIONS*

2.1 Introduction

Earth observation missions formed by tens, hundreds or even thousands of satellites are gaining popularity thanks to a big change in the manufacturing process and new technology advances, which enable the high volume and low-cost production of small satellites. As performance metrics requirements such as global coverage and revisit times keep becoming more and more restrictive, the need of increasing the number of satellites in the final architecture arises.

The constellation and orbit design for this type of missions is an underlying problem in nowadays research. It is a complicated problem mainly due to the fact that the number of possible architecture designs explodes with the number of satellites in the constellation. Even only considering symmetrical patterns such as Walker constellations, the number of design parameters grows very fast with constellation size. Therefore, this combinatorial explosion together with the highly nonlinear and computationally expensive numerical simulations required to evaluate these system designs (e.g. coverage performance, robustness, cost, etc.) make this design problem very challenging. In other words, the full factorial enumeration of architectures and their posterior evaluation is infeasible and, consequently, more sophisticated sampling strategies for tradespace exploration or the use of optimization techniques are needed to solve this problem.

As previously mentioned, restricting the constellation design space to simple geometries, such as Walker constellations, is a common practice found in the literature to make the con-

*Part of this chapter is reprinted with permission from "Evolutionary formulations for design of heterogeneous Earth observing constellations" by Pau Garcia Buzzi and Daniel Selva, 2020. 2020 IEEE Aerospace Conference, pp. 1-10, Copyright 2020 by IEEE.

stellation and orbit design of Distributed Spacecraft Missions (DSMs) more tractable. These constellation patterns often provide good coverage performance thanks to their symmetry properties. However, it is known that other constellation configurations that do not follow these symmetric patterns could offer better performance for certain applications, such as regional coverage missions or Earth observing missions aimed to track disjoint areas of interest of the globe.

While reducing the design space size by restricting the architecture configuration to a simple constellation pattern is a way to sub-optimally solve the orbit design problem, it is not the chosen approach in this thesis. Instead, evolutionary optimization methods are used to explore the constellation design space, allowing to search through simple base configurations such as Walker constellations, heterogeneous walker constellations with satellites at different altitudes and inclinations, satellite trains, string-of-pearls and ad-hoc constellations. The main contribution of this work is that the approach presented not only allows to explore the space of this base configurations, but also combinations of multiple simple constellations (e.g. A Walker constellation at 600km altitude and 30° inclination + a Sun Synchronous Orbit (SSO) train at 800km altitude). By doing so, we open up the tradespace to cover a much broader and never studied set of constellation designs during Pre-Phase A and Phase A studies of Earth Observation space missions.

Indeed, the methodology presented in this thesis was developed in the context of a NASA-funded project to develop a tool called Tradespace Analysis Tool for Constellations using Machine Learning (TAT-C ML). This tool is intended to be used during early stages of the design of Earth observation missions, and it is planned to be released open source within the next year. TAT-C ML was developed to analyze DSMs, which include heterogeneous instruments hosted on different satellites forming the constellation. This tool is able to evaluate a very large number of design alternatives in a relatively short time period, with the ultimate goal of informing feasibility studies, trade studies and what-if analyses. Its first version

(TAT-C) was born in 2014 and developed by a team at NASA's Goddard Space Flight Center. TAT-C contained an orbits module to propagate satellites and calculate coverage metrics as well as a cost and risk module. The main drawback was that the design search strategies were limited to an extensive brute-force enumeration of the design space. TAT-C ML was developed with the objective of searching through the design space in a much more efficient and intelligent manner, by combining genetic algorithms and machine learning techniques. In this thesis, the evolutionary formulations used by the Tradespace Search Executive (TSE) module in TAT-C ML are described. One challenge for the TSE was that there are different types of constellations to enumerate, which require different formulations in terms of the optimization, in which the number of variables in the chromosome depends on the type/s of constellation and the number of satellites. To address this problem, a variable-length formulation was developed and used to search the design space for any DSM that combines the five types of constellations considered: Delta Homogeneous Walker, Delta Heterogeneous Walker, Train, String-of-pearls and Ad-hoc constellations.

The remainder of this chapter is structured as follows: Section 2.2 contains a literature review of how evolutionary algorithms have been used to solve the constellation design problem, as well as a more thorough description of TAT-C ML. Section 2.3 presents the evolutionary formulations (chromosome and operators) developed to explore hybrid constellation designs. Sections 2.4 and 2.5 present two case studies used to test the performance and versatility of the evolutionary formulations proposed. Finally, Section 2.6 discusses the limitations of the study and opportunities for future work.

2.2 Background and Literature Review

2.2.1 Evolutionary algorithms in constellation design

In the last few decades, genetic or evolutionary algorithms have been proven to be an effective way of solving mixed-integer problems with complex objective functions. Indeed,

the numerical simulations required to evaluate the coverage performance for satellite constellations (using available software such as STK, GMAT or Orekit) constitute highly non-linear and non-convex objective functions, for which traditional optimization methods such as gradient descend are ineffective. Consequently, evolutionary algorithms establish an adequate method for solving the constellation design problem for DSMs.

Since in constellation design there exist multiple conflicting objectives, such as the well-known trade-off between the cost of launching and operating a constellation and its coverage performance, multi-objective evolutionary algorithms (MOEAs) are commonly used. The preferences between these multiple objectives are problem-specific and are very hard to define a priori and, consequently, it is challenging to convert the multi-objective problem into a single-objective one by performing, for instance, a weighted average of the different relevant metrics.

The first papers found in the literature that utilize evolutionary algorithms to obtain and analyze the optimal Pareto front of constellation designs considering multiple conflicting objectives –such as coverage metrics like percent coverage and revisit time, or the total number of satellites in the constellation– are from the late 1990s and early 2000s [48, 49, 50]. Together with the latest advances in evolutionary optimization, Ferringer et al. [62] utilized probably the most popular genetic algorithm NSGA-II [98] to explore the trade-off between three very popular coverage metrics: area-weighted average revisit time (AWART) and maximum revisit time and ground sample distance (GSD). A posterior paper by Wang et al. [99] used an MOEA based also on the NSGA-II algorithm to optimize the design of a regional covreage reconnaissance satellite constellation. In the following years, MOEAs became more and more popular in the orbit and constellation design area of research, leading to a big amount of publications aimed to design optimal satellite constellations by means of genetic algorithms [51, 52, 53, 54, 55, 56, 57, 58, 59].

It is important to note that most of the works cited constrain the orbit design space to

simple and well-known symmetrical constellations, such as the Walker Delta [27], streets of coverage [100], Rosette [101] and Flower [102] patterns. As mentioned in the introduction, this allows to simplify the problem by controlling the combinatorial explosion of feasible constellation designs while preserving a certain level of coverage performance. For instance, [52] and [54] optimize the design of a small satellite constellation for continuous coverage and the spacecraft and orbit design for a re-configurable satellite constellation, respectively, by only considering Walker constellations. In these works only circular orbits are considered and the optimization variables essentially correspond to the number of planes, the number of satellites in each plane, and a common value of altitude and inclination for all the satellites forming the constellation. Along the same lines, [55] also solves the constellation design problem by introducing a two-step optimization method and restricting again the design space to Walker constellations. The Walker delta is also the chosen pattern in [56] and [57] for the optimal design, respectively, of a LEO satellite broadband network and of a constellation of CubeSats aimed to provide continuous coverage over Europe to respond to hypothetical emergencies. Xu et al. [58] optimizes a Global Navigation Satellite System (GNSS) radio occultation constellation using evolutionary algorithms considering two very specific cases of Flower constellations (2D-LFC and 3D-LFC).

Contrarily, other works found in the literature present more general formulations that do not constrain the design space to the above mentioned simple and symmetrical constellation geometries. However, these formulations often are way too general and prevent the genetic algorithm to find the optimal Pareto front in a reasonable amount of time. Moreover, these general formulations do not take into account the inevitable increase of operational costs in highly non-symmetrical heterogeneous constellations containing satellites whose orbital parameters –altitude, inclination, longitude of the ascending node (RAAN), eccentricity, argument of perigee and mean anomaly– are completely uncorrelated from one satellite to another. The general evolutionary formulations found in the literature are very diverse:

[48] focuses on the design of circular geosynchronous constellations (and therefore altitude and eccentricity are given fixed inputs to the problem) and considers each of the satellite's RAANs and mean anomalies as separate optimization variables together with extra variable corresponding to the inclination, shared by all the different satellites forming the constellation. While altitude is also a fixed given input parameter in [49], this work includes the study of elliptical orbits. To do so, a separate variable corresponding to the argument of perigee is added to the formulation presented in [48] for each satellite in the constellation, as well as an extra variable corresponding to the eccentricity, common in all satellite orbits. Another formulation presented in [62] fixes the total number of satellites in the constellation, and includes altitude as an optimization variable shared by all the satellites in the constellation. The RAANs and mean anomalies for all satellites are separate variables included in the chromosome and inclination is treated as a problem constraint. Finally, [51] and [53] break away from the assumption of rigidly symmetric orbits and search through a much broader design tradespace. While the former considers independent inclinations, RAANs and mean anomalies for each satellite and a semi-major axis common across all the satellites forming the architecture, the latter considers separate optimization variables for the six orbital elements of all the satellites in the constellation. The general approach presented in [53] searches for non-symmetric and circular LEO constellations with a fixed number of satellites.

Some of the general formulations mentioned in the previous paragraph allow for different values of total number of satellites in the explored constellation designs. This is done by means of flagging [48, 49, 99, 51], in which the different possible satellites in the constellations are turned "on" or "off" using Boolean variables. However, other works [103, 64, 28] introduce the use of variable-length chromosomes to solve two issues that make the search of a genetic algorithm more challenging: the low-locality and the redundancy of fixed-length chromosomes that use Boolean flags to include or exclude participating satellites in the constellation. Furthermore, the method presented in [64] is the least constrained

evolutionary formulation found in the constellation design literature because, not only uses a variable-length chromosome, but also optimizes altitude, inclination, eccentricity, argument of perigee, RAAN and mean anomaly independently for each of the satellites in the constellation.

To summarize, evolutionary algorithms have been extensively applied to constellation design for the last few decades. However, existing formulations are either too simple and reduce the design space to symmetrical constellation geometries that are not representative of the whole tradespace, or too complicated for the genetic algorithm to come up with reasonable designs in an acceptable computational time. The methodology proposed in this thesis aims to offer a good balance between these two issues by defining a new chromosome able to create hybrid constellations formed by multiple simpler constellations, which are presented in section 2.3. This method not only considerably opens up the tradespace with respect formulations that only consider Walker constellations by generating more heterogeneous designs, but also keeps the constellation asymmetry down to a certain level. Finally, the approach presented in this thesis allows to help find synergies between potentially new and already existing constellations – e.g. how a Walker constellation could improve the science return of the NASA A-Train, which could be though as an hybrid constellation formed by a Sun-synchronous train and a Walker Delta Constellation.

2.2.2 Tradespace Analysis Tool for Constellations using Machine Learning (TAT-C ML)

As previously mentioned in the introduction, TAT-C ML is a constellation design tool intended to be used during the early design stages of Earth Observing missions. TAT-C ML was developed with the purpose of efficiently exploring large tradespaces of constellation designs, with the ultimate goal of performing feasibility studies, trade studies and what-if analyses.

Figure 2.1 shows the modular architecture of the TAT-C ML. The main input to the tool is the tradespace search request (TSR) JSON file, which can be either generated manually or through a built-in Graphical User Interface (GUI). The TSR contains all the necessary information required by the Tradespace Search Executive (TSE) module to enumerate constellation designs, such as what type of constellations we seek to explore, satellite and instrument availability and orbital parameters bounds among others. In other words, the TSR contains information about the decision variables and constraints for the search, which are transformed by TSE into a set of decisions that formally define a design space to search over. Once this first step is completed, the TSE module is now in charge of generating new architecture designs or solutions, which are later evaluated by the different TAT-C ML modules (orbits and coverage, instrument, launch, maintenance, cost and risk, and value modules). These solution evaluation modules produce a set of outputs that are fed back to the TSE to help guide the search towards good regions of the tradespace. In order to do so, the TSE uses three different search strategies to come up with new constellation designs:

- An exhaustive full factorial enumeration, which is simple and complete but very computationally expensive and generally infeasible for very large design spaces.
- A generic multi-objective evolutionary algorithm (ϵ-MOEA), which is still relatively simple and flexible but may need several function evaluations to converge to the optimum.
- Knowledge driven optimization (KDO), which is more complex than the second search strategy and it is based on discovering domain dependent knowledge to help the optimization algorithm converge faster to the optimum.

TAT-C ML includes several different constellation types, which are explained in detail in section 2.3, but listed below for the sake of clarity:

- Homogeneous Walker constellations, which is a very well known constellation pattern and it provides very good coverage performance thanks to its symmetry. Altitude and inclination are fixed, planes are equally spaced in RAAN and satellites within a plane are equally spaced in mean anomaly.

- Heterogeneous Walker constellations, which mix walker planes at different altitudes and inclinations while keeping a certain level of symmetry with planes and satellites within a plane equally spaced in RAAN and mean anomaly, respectively.

- Sun-Synchronous Trains, such as the NASA A-train, which are defined by a fixed altitude, a reference Longitude Time of the Ascending node (LTAN) and the RAAN and the mean anomalies are defined by the difference between LTANs of all the satellites composing the train.

- String-of-pearls constellations, in which all spacecraft are located in the same orbital plane and two consecutive satellites are separated by a small amount of degrees in mean anomaly.

- Ad-hoc constellations, which essentially chooses n random satellites from the Planet Lab's database in order to model secondary launch opportunities.

- Hybrid Constellations, which combine two or more of the aforementioned constellations, which could be of the same or different type.

The focus of the remainder of this chapter is to describe the evolutionary formulations used by the generic multi-objective evolutionary algorithm search strategy, which at the same time constitute the base of the KDO intelligent search strategy.

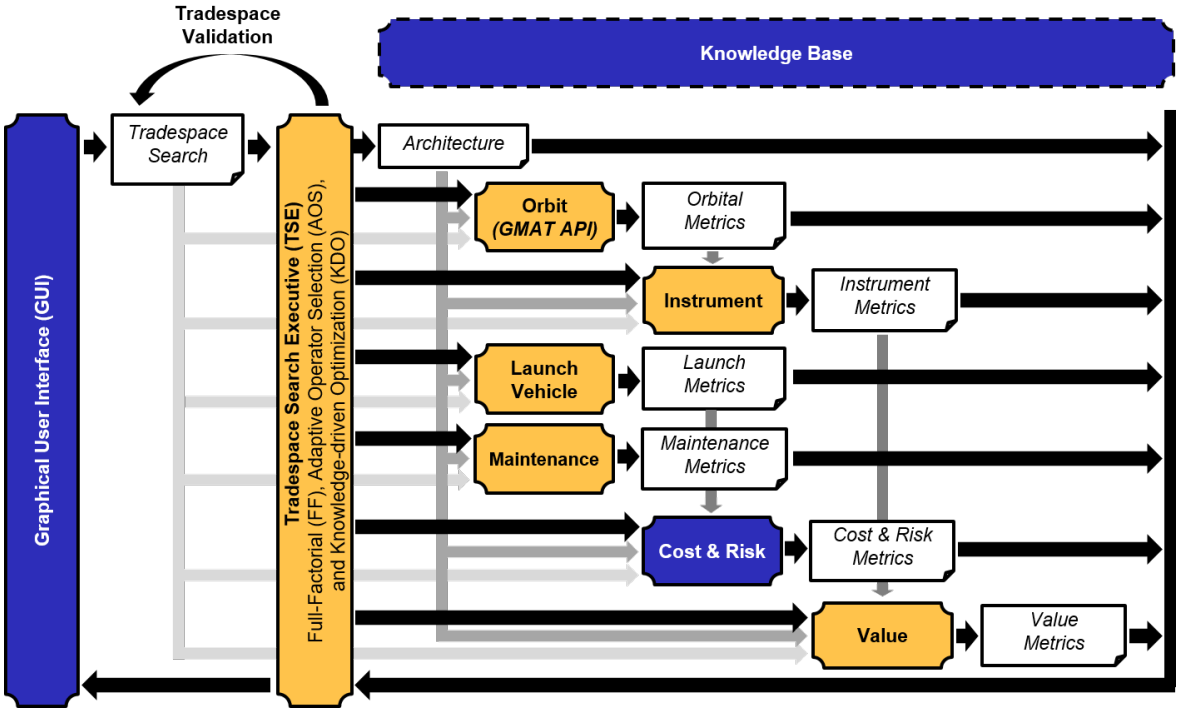


Figure 2.1: TAT-C ML module architecture.

2.3 Evolutionary formulations for hybrid constellations

2.3.1 Delta Homogeneous Walker

The well known Walker pattern was already included in the first version of TAT-C. A Walker constellation is defined by a 5-tuple (a, i, t, p, f) consisting of the altitude a and the inclination i of the orbital plane in which all t spacecraft are placed. In this type of constellations, the satellites are distributed in p planes equally spaced in RAAN and the mean anomaly spacing between two satellites in adjacent planes is defined by the phasing parameter f and equal to $\frac{360^\circ}{t} f$.

A chromosome of fixed length equal to 5 (one for each of the aforementioned parameters: altitude, inclination, number of satellites, number of planes and phasing parameter) was chosen to represent a Walker constellation and depicted in Figure 2.2. The altitude, in-

Figure 2.2: Fixed-length chromosome encoding a Delta Homogeneous Walker constellation.

clination and number of satellites variables are encoded as integer variables corresponding to the indexes of a list of allowed values, which are defined in the tradespace search JSON file which, as already mentioned, is the main input to the TAT-C tool. The number of planes and phasing parameter variables, however, are defined as real values that take values from 0 to 1 that are later converted to their actual values: First, the possible options for number of planes are computed depending on the number of satellites t – p has to be a divisor of t. Then, the real value from 0 to 1 is mapped onto the actual number of planes option taking into account all of the possible plane options. For instance, if there are $t = 3$ satellites, then the possible values for p are $p = 1, p = 3$. Therefore, any values less than 0.5 in the number of planes variable are mapped to $p = 1$, whereas values greater than 0.5 are mapped to $p = 3$.

2.3.2 Heterogeneous Planes (or Delta Heterogeneous Walker)

This second constellation pattern was called Heterogeneous Planes or Delta Heterogeneous Walker because satellites are still distributed in planes equally spaced in RAAN and satellites within a plane are also equally spaced in mean anomaly. However, the heterogeneity comes from the fact that these planes are placed in different altitudes and/or inclinations, unlike in the Delta Homogeneous Walker pattern, where all satellites share common values of altitude and inclination. This type of constellation would be desirable in applications where diverse instruments are placed in the different satellites forming the constellation, and the maximum performance of these instruments are achieved in different altitude/inclination values. Finally, similar to Delta Homogeneous Walker constellations, a phasing parameter f

Figure 2.3: Fixed-length chromosome encoding a Delta Heterogeneous Walker constellation.

is defined to determine the phasing between satellites in adjacent planes.

A variable-length chromosome shown in Figure 2.3 was chosen for Delta Heterogeneous constellations. This chromosome is defined by the total number of satellites n, p 2-tuples that contain the altitude and inclination of the p planes forming the Delta heterogeneous Walker constellation and the phasing parameter f. All planes contain n/p satellites and, as mentioned earlier, they are equally spaced in RAAN and satellites within a plane are equally spaced in mean anomaly. The altitudes, inclinations and number oif satellite variables in the chromosome are encoded as integer variables corresponding to the indexes of a list of allowed altitudes, inclinations and total number of satellites. Similarly to what is done in the chromosome for Walker constellations, the phasing parameter f is encoded as a real variable between 0 and 1 that is later converted to its actual value taking into account the total number of planes p in the constellation.

2.3.3 Sun-Synchronous Train

A Sun-synchronous train is a new constellation included in the latest version of TAT-C. Inspired by constellations such as the NASA A-Train, SSO trains are constellations that contain satellites with different Longitude Time of the Ascending Node (LTAN), equally separated by a certain amount of time $t_{interval}$, that aim to observe the same points in the Earth within a certain de-correlation time. For a given altitude (which at the same time determines the SSO inclination since we are assuming circular orbits) and the reference Longitude Time of the Ascending Node (LTAN) of the first satellite, the RAANs for all satellites are computed taking into account the difference in LTAN between consecutive satellites. Moreover,

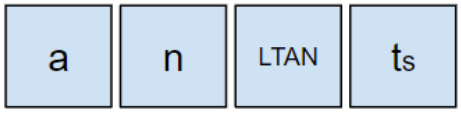


Figure 2.4: Fixed-length chromosome encoding a Sun Synchronous Train constellation.

to ensure that the same points are revisited by all satellites in the constellation on the same day, the initial true anomaly spacing between the different satellites has to be normalized to the orbital period using the following equation:

$$\nu_{sat_2} = \nu_{sat_1} - 360^\circ \cdot \frac{LTAN_{sat_2} - LTAN_{sat_1}}{T_{sat_1}} \tag{2.1}$$

where ν_{sat_1} and ν_{sat_2} are the true anomalies of two consecutive satellites. $LTAN_{sat_1}$ and $LTAN_{sat_2}$ are the longitude time of the ascending node of the satellites and it is assumed that $LTAN_{sat_2}$ is greater than $LTAN_{sat_1}$ – i.e. sat_2 is following sat_1. Finally, T_{sat_1} is the orbital period of the first satellite.

A chromosome of fixed length equal to 4 encoding the altitude, number of satellites, LTAN of the reference or first satellite and satellite interval time was chosen to represent a SSO train constellation and shown in Figure 2.4. All variables are encoded as integer variables corresponding to the indexes of a list of possible allowed values, which are defined, again, in the tradespace search JSON file.

2.3.4 String-of-pearls

The string-of-pearls configuration consists of placing all satellites in the constellation in the same orbital plane. The different spacecraft are separated by a certain phasing value in true anomaly. This usually corresponds to the cheapest configuration when it comes to launching a fleet of small satellites, since only one launch is required.

A chromosome of fixed length equal to 5 was chosen to encode a string-of-pearls con-

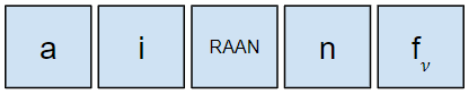


Figure 2.5: Fixed-length chromosome encoding a string-of-pearls constellation.

stellation and shown in Figure 2.5. The first 3 variables (altitude, inclination and RAAN) define the orbital plane where the satellites are placed. The last two variables correspond to the number of satellites and the true anomaly phasing between them. All variables are encoded as integer variables corresponding to the indexes of a list of possible allowed values defined in the tradespace search JSON file.

2.3.5 Ad-Hoc

Ad-hoc constellations investigate cheaper options for launching CubeSats as secondary payloads by using the Planet Labs satellites database. The motivation behind this approach is that Planet launches their 3U Cubesats on secondary launches whenever they become available, so it is a representative way to model upcoming rideshare launching services. The way the TSE enumerates the different architectures with n satellites would be by choosing a random combination among all possible n-combinations of all the satellites available in the Planet Labs database.

The chromosome for this type of constellation is very simple and only encodes an integer variable corresponding to the index of a list of possible number of satellites forming the ad-hoc constellation.

2.3.6 Hybrid constellations

The full power of the TAT-C ML tool comes with the concept of hybrid constellations, which allow to explore a very wide tradespace of constellation space never explored before. Hybrid constellation allow to mix one or more constellation types previously explained (of same or different type). For instance, a constellation mixing a Walker constellation at a low

inclination and a Sun-synchronous train could be generated. This example of hybrid constellation would be desirable in applications whose objective is to monitor the tropics region as well as the Poles. Another application where hybrid constellations are advantageous is when the satellites carry distinct instruments and need to be placed at different altitudes. In that case, a hybrid constellation formed by two Walker constellations at different altitudes could potentially be a good design alternative.

2.3.6.1 Chromosome

The chromosome for hybrid constellations consists of several constellation sub-types chromosomes, one next to each other, and in the tradespace search JSON file specified order. Figure 2.6 shows the general structure of a hybrid architecture (top) as well as a particular example consisting of a hybrid constellation formed by 2 homogeneous Walker and a SSO train (bottom). The hybrid chromosome allows for the number of satellites variable for each base constellation to be set to 0. This allows to explore also the space of constellations which contain a lower number than the maximum allowed by the hybrid chromosome (defined again in the tradespace JSON file). For instance, for the chromosome shown at the bottom of Figure 2.6, the GA would search through the space of constellations formed either by 2 Walkers and a SSO train, 2 Walkers, a Walker and a SSO train, 1 Walker or 1 SSO train.

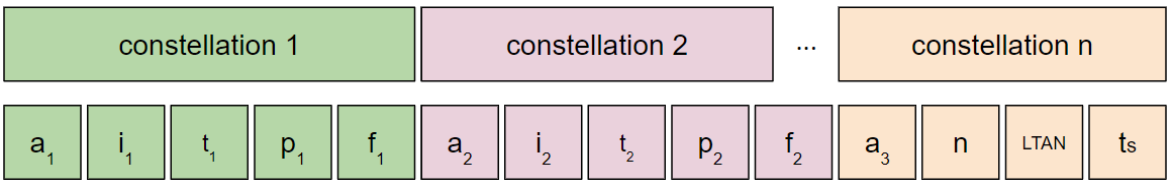


Figure 2.6: Chromosome encoding a hybrid constellation.

2.3.6.2 *Operators*

The challenge of the hybrid constellation evolutionary formulation proposed comes with the design of the operators. The main issue is that well-known operators such as crossover cannot be applied to variable-length chromosomes since the length of the two parents can potentially be different. The hybrid chromosome can be variable-length due to the hypothetical inclusion of a Delta Heterogeneous Walker constellation. The methodology adopted in this thesis consists of pairing the chromosomes of the same constellation type from both parents and apply the operators independently for each constellation sub-type. This methodology is illustrated in Figure 2.7. The operator chosen to evolve each of the constellation sub-types will depend on either if the constellation sub-type is encoded in a fixed length chromosome (homogeneous Delta Walker, SSO trains, String of Pearls and Ad-hoc constellations) or in a variable-length chromosome (heterogeneous Delta Walker). The different operators available for fixed and variable length chromosome are discussed below.

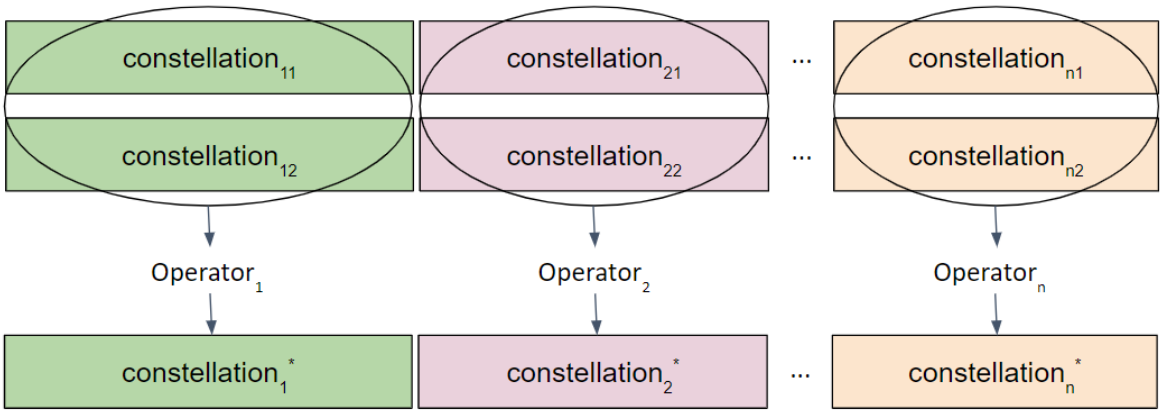


Figure 2.7: Genetic evolution of the hybrid constellation chromosome

2.3.6.2.1 Fixed-length chromosomes – For constellation sub-types with fixed-length chromosomes such as homogeneous Delta Walker, SSO trains, String of Pearls and Ad-hoc con-

stellations, the following steps are taken in order to evolve the corresponding base constellation chromosome:

1. Separate integer variables and real variables of the base constellation chromosome.

2. Perform with probability p_c either a Simulated Binary Crossover operation for real variables or a two-point crossover operation for integer variables.

3. Additionally, only for integer variables, the resulting child from the two-point crossover operation undergoes a mutation operation with probability p_m.

2.3.6.2.2 Variable-length chromosomes – The operators listed so far cannot be applied to the variable-length heterogeneous walker chromosome because the length of the two parents can potentially be different. Instead, 2 new operators were created: the $cut\ and\ splice$ operator and the $pairing$ operator. The $cut\ and\ splice$ operator cuts the two parent chromosomes at a randomly selected point, which is restricted to be between two planes as opposed to within a plane, and splices the cut chromosomes to create two offspring with different chromosome lengths (see Figure 2.8). The number of satellites for the offspring (n_i*) are chosen in the following way: if n_1 and n_2 are multiples of the resulting number of planes, choose either one randomly. If only n_1 or n_2 are multiples of the resulting number of planes, choose that value for n_i*. If neither n_1 nor n_2 are multiples of the number of planes, choose randomly any possible option from the list of allowed total number of satellites. The $pairing$ operator "pairs" each of the planes from the two parent chromosome with the fewest planes with a plane from the other parent and chooses with equal probability the values of altitude and inclination of the paired planes (see Figure 2.9). The $pairing$ operators allows to generate combinations of altitudes/inclinations that do not yet exist within the solutions in the population, whereas the $cut\ and\ splice$ operator is not able to do so.

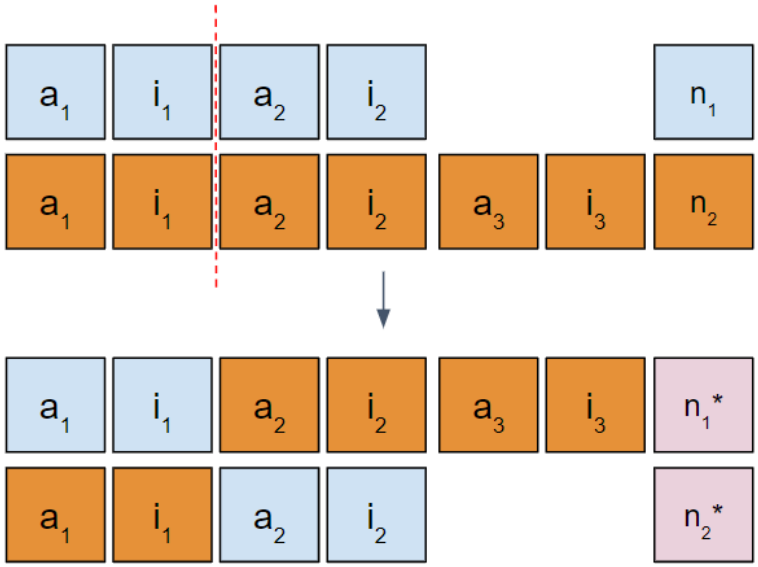


Figure 2.8: Cut and splice operator for the variable length chromosome from formulation 2.

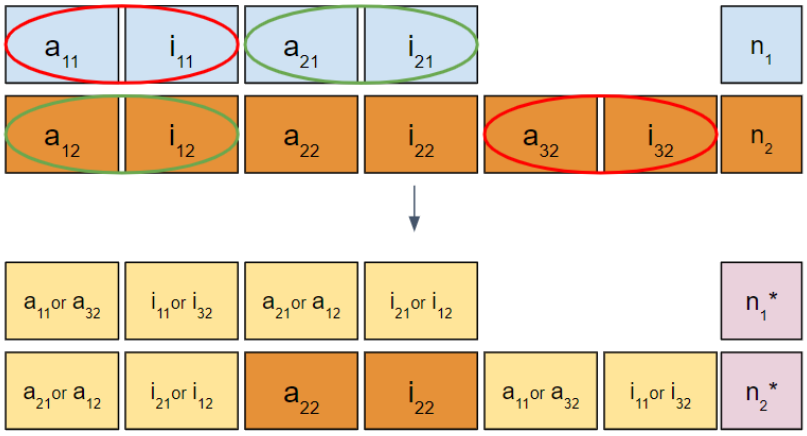


Figure 2.9: Pairing operator for the variable length chromosome from formulation 2.

2.4 Case Study 1: The performance of the proposed evolutionary formulations

In this case study, we assess the performance of two new evolutionary formulations for heterogeneous constellation design. These two formulations are compared to a third for-

mulation proposed earlier in the literature, which consists of a variable-length chromosome containing n 4-tuples to encode a constellation of n satellites, which provides the highest level of heterogeneity in all of the satellite's orbital elements and allows to generate constellations with no symmetries at all. The performance of these formulations is compared by solving a constellation design problem where the objectives are to optimize coverage performance and lifecycle cost. Search performance is assessed by looking at the evolution of hypervolume with the number of function evaluations.

- Formulation 1 - The hybrid Walker: The main idea behind this first formulation is to combine multiple Walker constellations at different altitudes and/or inclinations into a hybrid architecture. The rationale is that the resulting designs will conserve a symmetric geometry while allowing its satellites to be placed in higher/lower altitudes and more/less inclined orbits, which can be desirable for certain applications [29]. A fixed-length chromosome containing m 5-tuples is used to encode m Walker constellations and shown in Figure 2.10. Each constellation is defined by a 5-tuple (a, i, t, p, f) consisting of the altitude a, the inclination i, the number of satellites t, the number of planes p and the phasing parameter f. In this formulation, the number of satellites in each constellation can be set to 0 to effectively reduce the number of Walker constellations in the hybrid design to anywhere between 0 and m. The altitude, inclination and number of satellites variables are encoded as integer variables corresponding to the indexes of a list of allowed values. The number of planes and phasing parameters are defined as real variables taking values from 0 to 1 that are later converted to their actual values: First, the possible options for number of planes are computed depending on the number of satellites – p has to be a divisor of t. Then, the real value from 0 to 1 is mapped onto the actual number of planes option taking into account all of the possible plane options. For example, if there are $t = 3$ satellites, then the possible values for p are $p = 1, p = 3$. Therefore, any values less than 0.5 in the number of planes

variable are mapped to $p = 1$, whereas values greater than 0.5 are mapped to $p = 3$. The operators used for this first formulation are m two-point crossovers between the m walker constellations of each parent followed by a mutation operation. This procedure is illustrated in Figure 2.11 for two parents containing 2 Walker constellations (i.e. $m = 2$).

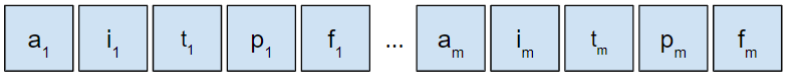


Figure 2.10: Fixed length chromosome encoding a hybrid architecture containing m Walker constellations.

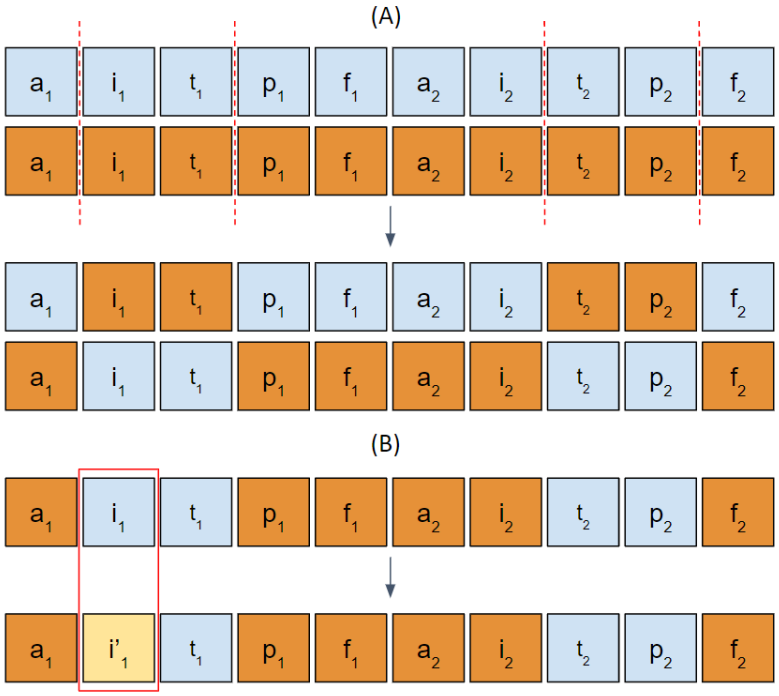


Figure 2.11: Operators used in formulation 1 with m two-point crossovers (A) followed by a mutation operation (B)

- Formulation 2 - Heterogeneous planes: This second formulation intends to combine multiple planes at different altitudes and inclinations without dividing the overall constellation into smaller Walker constellations as described in subsection 2.3.2. This allows us to generate designs which are less symmetric than the ones in the previous formulation, as well as to have a greater number of planes at different altitudes and inclinations. Indeed, in formulation 1, the maximum possible number of different inclination/altitude pairs would be equal to m (i.e. the maximum number of Walker constellations in the hybrid architecture). Figure 2.3 shows the variable-length chromosome used in this second problem formulation, which encodes a heterogeneous constellation defined by the total number of satellites n, and p 2-tuples that contain the altitude and inclination of the p planes forming the constellation. Note that, unlike in Formulation 1, p is variable, and therefore the length of the chromosome depends on the architecture. The two-point crossover operator cannot be applied to the variable-length chromosome of this second formulation because the length of the two parents can potentially be different. Instead, either the $cut\ and\ splice$ operator or the $pairing$ operator shown in Figures 2.8 and 2.9 are selected with probability equal to 0.5, followed again by a mutation operation.

- Formulation 3 - General: With this last formulation, we are able to explore the largest possible tradespace of constellation designs. It uses a variable-length chromosome, shown in Figure 2.12, containing n 4-tuples to encode a constellation of n satellites, where each satellite is defined by a 4-tuple with its altitude, inclination, RAAN, and mean anomaly. The original formulation [64] used 6-tuples, which additionally contained the eccentricity and argument of perigee for each satellite. However, this work focuses on assessing designs with circular orbits. All variables in the chromosome are encoded as bounded real variables. Similar to the procedure described for the heterogeneous planes formulation, this last formulation uses either a $cut\ and\ splice$ or a

modified simulated binary crossover operator, selected with probability equal to 0.5, followed by a mutation operation. The modified simulated binary crossover is similar to the $pairing$ operator used in formulation 2: it starts by identifying the chromosome parent containing a smaller number of satellites. Then, it pairs randomly each of the satellites of this parent with another satellite from the other parent. Finally, the paired satellites undergo a simulated binary crossover operation [104] and the unpaired satellites of the large parent retain their original values. We refer the reader to the original paper by Hitomi and Selva [64] to obtain a detailed description of the operators used in this general formulation.

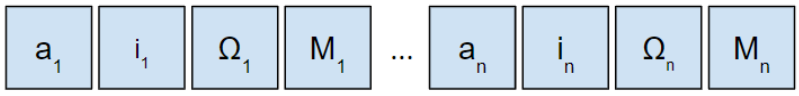


Figure 2.12: Variable length chromosome encoding a constellation defined by $4n$ orbital elements.

2.4.1 Constellation design problem

The proposed formulations are applied to solve a multi-objective constellation design problem with four objectives selected in order to maximize coverage performance while minimizing overall cost. These objectives are the following:

- Minimize mean revisit time, which is the average length of the gap intervals for all the points of interest seen by the constellation.

- Minimize mean response time, which is defined as the time average of the response time. Response time corresponds to the time from when a random request is received to observe a point of interest until the constellation can actually observe it. Mean response time is known to be more sensitive to the number of planes than mean revisit time, so there can be a trade-off between these two parameters [32].

- Maximize the ratio between the number of points observed by the constellation and the total number of points in the region/s of interest. Since mean revisit time and mean response time are calculated only on observed points and do not take into account unobserved points, this third coverage metric was included as objective in order to penalize those designs that observe fewer points in the area of interest.

- Minimize life-cycle cost, which includes launch, program, integration, ground, hardware, recurring and non-recurring costs.

Since the evolutionary formulations discussed in this thesis were developed in the context of a NASA-funded project to develop a tool called Tradespace Analysis Tool for Constellations using Machine Learning (TAT-C ML), this tool was used to evaluate each of the constellation designs and obtain their respective objective values. TAT-C ML is intended to be used during early stages of the design of Earth observation missions and it is planned to

be released open source within the next year. The details of both cost and orbital propagation models used for TAT-C ML are provided in [42].

The case study defined to compare the three evolutionary formulations has the goal to design a constellation of satellites able to monitor the Tropics and the North Pole regions, with two areas of interest located between latitudes of -30° and 30°, and 60° and 90°, respectively. This example was chosen because the disconnected region of interest is one case where heterogeneous planes might be preferable to symmetric constellations with planes at a single combination of altitude and inclination.

In this work, only low-Earth, circular orbits are considered and each satellite in the constellation carries the same instrument with a rectangular field of view defined by a half cross-track angle of 57° and a half along-track angle of 20°. Tables 2.1, 2.2 and 2.3 show the possibles values/ranges of the design variables for each of the 3 formulations. In the hybrid Walker formulation, we consider the tradespace composed by up to three walker constellations, each containing up to 4 satellites, generating a hybrid architecture formed by a maximum of 12 satellites. The 3 Walkers have an altitude between 400 km and 800 km and the option to set the number of satellites equal to 0 to allow for the generation of designs that only contain 1 or 2 constellations. The first and second constellations aim to cover the area around the tropics with low inclinations, whereas the Walker containing Sun-Synchronous Orbits (SSO) is the one responsible for covering the North Pole region.

Similarly, the second formulation considers a heterogeneous constellation space containing up to 12 satellites distributed in up to 12 planes. Each of these planes is located at an altitude of 400km, 500km, 600km, 700km or 800km, and at an inclination of 0°, 10°, 20°, 30°, 51.6°, 90° or SSO. By breaking the Walker pattern symmetry, this formulation opens up the tradespace to a larger set of heterogeneous constellation designs.

Finally, the design variables considered in the general formulation include the number of satellites in the constellation and each satellite's orbital elements. The number of spacecraft

varies from 1 to 12, altitude from 400 to 800 km, inclination from 0° to SSO, and RAAN and mean anomaly from 0° to 360°. Since this problem formulation uses continuous variables for each of the satellite's orbital elements, the optimization algorithm explores a tradespace that contains all of the possible LEO circular constellations within the specified variables bounds.

In all of the formulations, the satellites are propagated for a week using the GMAT [105] orbital propagator included in TAT-C ML. All coverage statistics are computed against a grid of 200 points spread across the two regions of interest.

Formulation 1		
Walker	**Decision**	**Allowed values**
1	Altitudes Inclinations Number of Satellites Number of Planes	[400, 500, 600, 700, 800] km [0°, 10°, 20°, 30°] [0, 1, 2, 3, 4] [1, 2, 3, 4]
2	Altitudes Inclinations Number of Satellites Number of Planes	[400, 500, 600, 700, 800] km [ISS] [0, 1, 2, 3, 4] [1, 2, 3, 4]
3	Altitudes Inclinations Number of Satellites Number of Planes	[400, 500, 600, 700, 800] km [SSO] [0, 1, 2, 3, 4] [1, 2, 3, 4]

Table 2.1: Possible values for the design variables in formulation 1

Formulation 2	
Decision	**Allowed values**
Altitudes Inclinations Number of Satellites Number of Planes	[400, 500, 600, 700, 800] km [0°, 10°, 20°, 30°, ISS, 90°, SSO] [1, 2, 3, 4, 6, 8, 10, 12] [1, 2, 3, 4, 6, 8, 12]

Table 2.2: Possible values for the design variables in formulation 2

2.4.2 Simulation setup

The search performed by evolutionary algorithms is stochastic, and consequently the results obtained may vary from run to run. For this reason, a total number of 30 simulations were run for each of the three formulations to be able to perform a fair statistical comparison. The ϵ-MOEA algorithm [106] was used as the baseline algorithm. This algorithm is steady-state (i.e., only one individual in the population is evolved per step) and uses an ϵ-dominance archive to maintain a well-spread set of Pareto-optimal solutions. The probability of the operators to apply crossover for formulation 1 and $cut\ and\ splice$ or $pairing$ for formulations 2 and 3 was set to $p_c = 0.9$. The probability of performing mutation for each variable (e.g., altitude, inclination) of the resulting chromosomes was set to $p_m = 0.2$ in all three evolutionary formulations proposed. An initial population of 100 random architectures was created at the beginning of each optimization run. Finally, the termination criteria was set to 1,000 function evaluations ($NFE = 1000$).

After running the total number of 90 simulations, the 90,000 architectures generated were joined together to compute the maximum and minimum values obtained for all 4 objectives, which were later used to scale between 0 and 1 the objective values of all the enumerated designs. This is common practice before proceeding to the computation of the hypervolume, an extensively used metric to evaluate the search performance of genetic algorithms [107]. The hypervolume of a set of points is the closed area (2 objectives), volume (3 objectives) or hy-

Formulation 3		
Decision	**Lower bound**	**Upper bound**
Altitude	400 km	800 km
Inclination	0°	SSO
Number of Satellites	1	12
RAAN	0°	360°
Mean anomaly	0°	360°

Table 2.3: Ranges for the design variables in formulation 3

pervolume (4+ objectives) generated by the union of all the rectangles/cuboids/hypercuboids defined by a reference vector v_{ref} (typically the anti-utopia point) and each of the points in the set. It is easy to see that only the objective vectors of non-dominated solutions contribute to the hypervolume. The higher the hypervolume indicator, the better quality has the Pareto front obtained by the evolutionary algorithm. Hypervolume was computed using the Pagmo Python library [108] after each function evaluation for every single run using the reference point $v_{ref} = [1, 1, 1, 1]$, which corresponds to the maximum (worst) possible value for each scaled objective.

In the following Section, the hypervolume indicator is used to assess the differences in performance of the 3 evolutionary formulations proposed. Particularly, the non-parametric Wilcoxon rank sum test with a significance level of 0.05 was used to compare the differences in hypervolume between all 3 formulations. Additionally, the compositions of the 3 resulting final Pareto fronts are analyzed and compared against each other.

2.4.3 Results

Figure 2.13 shows the evolution of hypervolume with respect the number of function evaluations (NFE) for each of the formulations proposed in this case study. The thick solid line represents the average hypervolume of the 30 runs and the lighter shaded area shows one standard deviation from the mean value. It is observed that the third formulation starts performing better than the other two, providing a statistically greater hypervolume ($p < 0.0444$ and $p < 0.0428$) than formulations 1 and 2 between 7 and 76 and 1 and 87 NFE, respectively. However, this first portion of the plot simply corresponds to the first 100 designs randomly generated. Then, between 100 and 200 NFE, the hybrid Walker becomes the best formulation and continues to outperform the other two until the end of the run. Conversely, the heterogeneous planes formulation starts performing worse than the other two but, around 300 NFE, it becomes better than the most general formulation in mean value. The variance

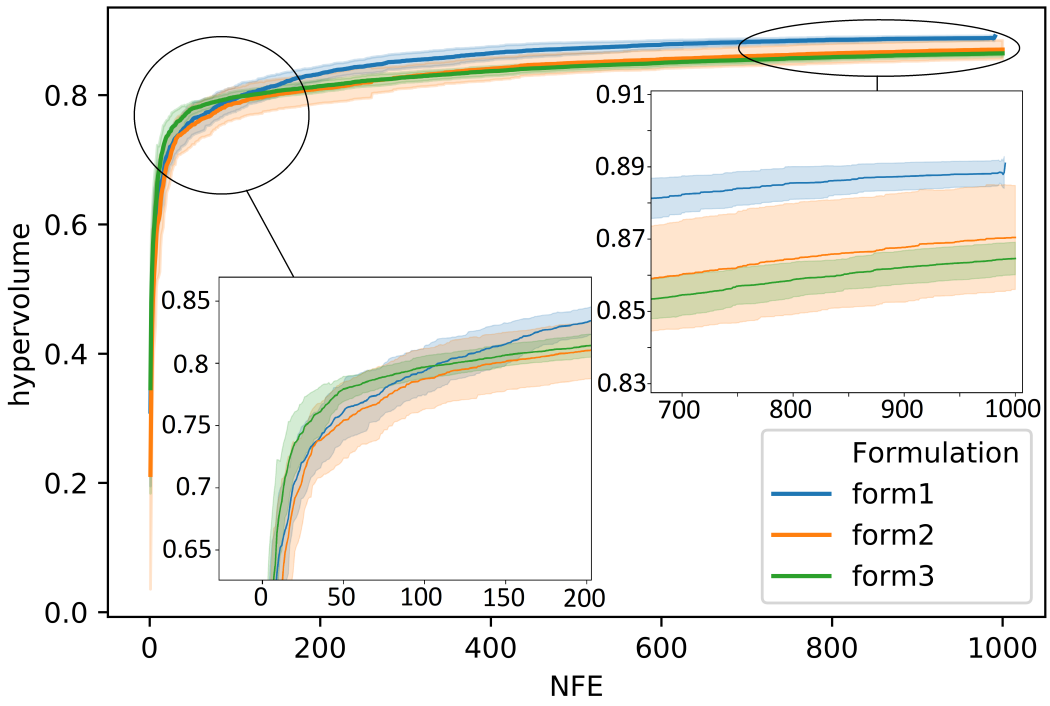


Figure 2.13: Hypervolume as a function of the number of function evaluations for each of the three formulations.

in hypervolume for this second formulation is larger than the other two and, despite its mean hypervolume is higher than the general formulation at 1000 NFE, a final lower hypervolume is obtained in some of the runs. The results from the Wilcoxon rank sum test showed that formulation 1 maintains a statistically greater hypervolume ($p < 0.0476$ and $p < 0.0444$) than formulations 2 and 3 after 138 and 140 function evaluations, respectively. Similarly, formulation 2 offers a statistically greater hypervolume ($p < 0.04926$) than formulation 3 from 433 function evaluations on.

Additionally, Figure 2.14 plots the ratio between the number of runs that reach the values of 0.85 and 0.87 hypervolume and the total number of runs (i.e., 30) with respect to the number of function evaluations. In other words, this figure shows the probability of attaining such hypervolume values by different values of NFE for each of the three evolutionary

formulations. In the bottom plot, 0.87 corresponds to the maximum value of hypervolume achieved by all three evolutionary formulations. It can be observed that the hybrid Walker formulation performs the best, with a probability of reaching 0.87 hypervolume of almost 60% within 400 NFE and 100% within less than 700 NFE. On the other hand, the heterogeneous planes formulation only reaches 0.87 hypervolume in roughly 70% of the runs, while the most general formulation only achieves that threshold value with a probability lower than 20%. Conversely, in the top plot of Figure 2.14 it is observed that formulations 2 and 3 attain 0.85 hypervolume with 80% and 100% probability, respectively. Even though the heterogeneous planes formulation provides higher probability of attaining 0.85 hypervolume than the general formulation for low NFEs, at 700 NFE formulation 3 becomes more reliable, reaching a value of 100% probability within only 800 NFE. This is due to the high variance of formulation 2 observed in Figure 2.13, where it is shown that all the runs for formulation 3 reach a hypervolume greater than 0.85, but some of the runs in formulation 2 do not.

The behaviour seen in Figures 2.13 and 2.14 can be explained by two main reasons:

- The selected constellation design problem, despite considering two different regions of interest at different latitudes (Tropics and North Pole), can still benefit from architecture designs mixing symmetric constellations at different inclinations (i.e., hybrid Walkers), since the points of interest are distributed uniformly across longitude.

- The three formulations considered have different design spaces. As previosuly mentioned, the tradespace of the third formulation contains all possible circular LEO constellations between 400km and 800km of altitude. Therefore, this formulation reasonably needs more time to explore a larger tradespace and converge to the symmetric designs offered by the hybrid Walker formulation, which logically needs a smaller number of function evaluations to do so. Finally, the heterogeneous planes formulation lies in between formulations 1 and 3 regarding design space size and symmetry of

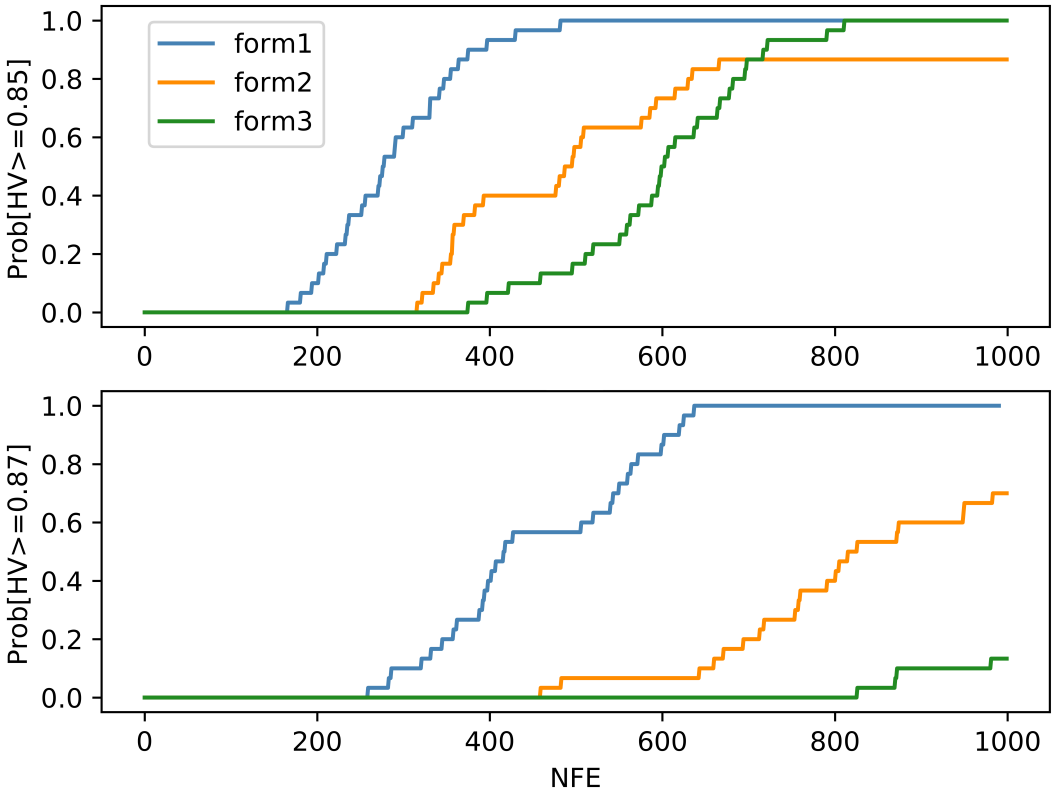


Figure 2.14: Probability of attaining 0.85 and 0.87 HV for each of the three formulations.

the constellations generated. Consequently, it offers higher search performance than the general formulation in the long run despite having smaller values of hypervolume after the random initialization at the beginning of the genetic algorithm. In summary, having a larger design space is a double-edged sword: on the one hand, it can help finding more novel constellation designs one could not think of. On the other hand, it will take more computation time to find constellation designs that one already knows are good, such as a symmetric Walker constellation.

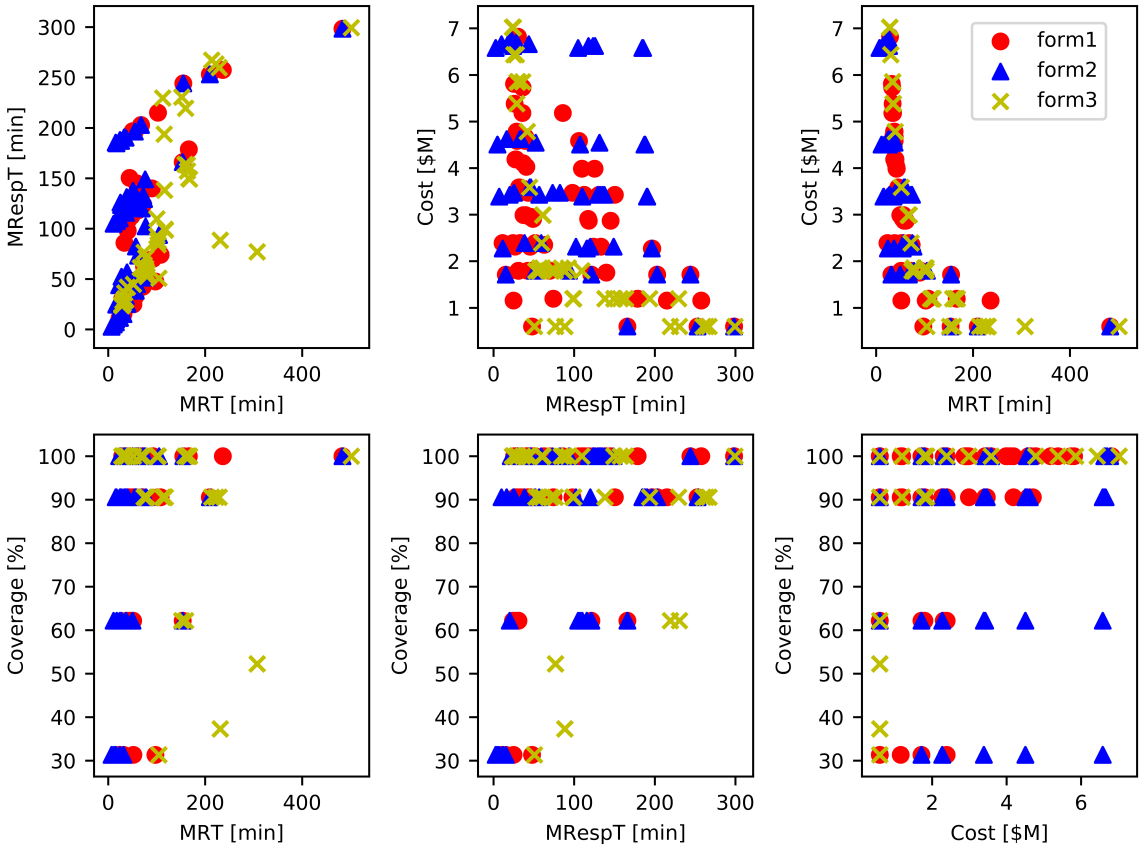


Figure 2.15: Comparison of Pareto fronts obtained in a representative run for each of the three formulations.

In an effort to compare the final non-dominated designs or Pareto front of the three studied formulations, Figure 2.15 shows the projection of the three 4-dimensional Pareto fronts onto the six different planes generated by all possible pairs of the four objectives considered in the optimization problem. Even though these six plots do not show actual 2-dimensional Pareto fronts but a projection of three 4-dimensional Pareto fronts, it can be noted that for most of the 2-D scatter plots, formulations 1 and 2 outperform formulation 3 as they both obtain better projected "Pareto" fronts.

While Figure 2.15 plots the Pareto fronts on the objective space, Table 2.4 aims to describe the obtained non-dominated solutions on the design space. In 3 representative runs (one for each formulation), the Pareto fronts obtained contain 65, 58 and 50 constellation designs for formulation 1, 2 and 3 respectively. In Table 2.4, just a small illustrative sample of this set of architectures is shown.

The hybrid Walker formulation effectively generates architectures that mix Walkers at low inclinations to monitor the Tropics and SSO Walkers to track the North Pole. These hybrid Walker constellations grant 100% coverage but also single Walker designs are found by the genetic algorithm, which are cheaper but provide worse coverage statistics.

Similarly, the heterogeneous planes formulation enumerates designs that mix planes at lower inclinations (20° and 30°) with polar and SSO planes to successfully observe both the tropical and polar regions. More affordable options with just one SSO plane are also found in the Pareto front, which provide 100% coverage at a much lower cost. Finally, constellations with just one plane at very low inclinations that have very good values of mean response and revisit times are also enumerated. However, these constellations only cover a bit over 30% of the whole area of interest.

Finally, the third and most general formulation enumerates designs that mix satellites at low inclinations (roughly between 15° and 30°) and high inclinations (roughly between 70° and 90°) to again track both areas of interest in our problem. As in formulation 1,

Decisions	**Objectives** (MRT, MRespT, Cost, Cov)
Formulation 1 - Hybrid Walker	
(800km 20° t:4 p:4 f:0) (800km SSO t:3 p:1 f:0) 7 sats	(2408.19s, 2200.86s, $4.11M, 100%)
(400km 0° t:1 p:1 f:0) 1 sat	(5844.69s, 2857.73s, $0.6M, 31.34%)
(800km 20° t:4 p:2 f:0) (800km 51.6° t:4 p:1 f:0) 8 sats	(2112.14s, 1748.14s, $4.58M, 90.55%)
(800km 20° t:4 p:4 f:0) (600km 51.6° t:2 p:1 f:0) (800km SSO t:4 p:4 f:3) 10 sats	(1957.52s, 1519.79s, $5.81M, 100%)
(800km 20° t:4 p:2 f:0) (800km 51.6° t:4 p:4 f:3) (800km SSO t:4 p:1 f:0) 12 sats	(1728.78s, 1821.48s, $6.82M, 100%)
Formulation 2 - Heterogeneous planes	
(800km SSO) (800km 30°) 4 sats	(4576.04s, 8932.34s, $2.31M, 100%)
(800km SSO) 1 sat	(28984.02s, 17902.67s, $0.6M, 100%)
(800km SSO) (800km 30°) 12 sats	(1504.54s, 7547.62s, $6.62M, 100%)
(800km 30°) (700km 90°) (700km 20°) (800km 20°) 12 sats	(1513.80s, 1701.70s, $6.7M, 100%)
(400km 0°) 8 sats	(666.39s, 299.76s, $4.5M, 31.34%)
Formulation 3 - General	
(790.83km 20.19°) (759.193km 79.75°) (748.54km 31.13°) (717.03km 26.77°) (795.53km 98.19°) (790.64km 5.34°) (745.1km 27.79°) (716.79km 26.14°) (776.59km 26.18°) 9 sats	(2079.68s, 1728.04s, $5.38M, 100%)
(732.10 20.19°) (785.13 14°) (784.15 23.43°) 3 sats	(4633.70s, 3644.62s, $1.85M, 90.55%)
(734.51 2.28°) 1 sat	(6278.17s, 3048.05s, $0.6M, 31.34%)
(717.04km 26.77°) (777.33km 15.99°) (737.61km 26.57°) (799.05km 20.26°) (696.33km 21.71°) (776.21km 79.63°) (702.03km 85.87°) (623.84km 25.07°) (738.89km 72.72°) (727.69km 82.93°) (773.5km 12°) (698.31km 90.54°) 12 sats	(1673.67s, 1476.89s, $7.02M, 100%)

Table 2.4: Sample of architectures in the Pareto front obtained in a randomly selected run for each of the three formulations

cheap solutions with just 1 satellite at a very low inclination are found in the final Pareto front. Almost all the non-dominated designs in all of the formulations contain high values of altitude between 700km and 800km. This happens because the cost model used to evaluate each of the designs is not sensitive to altitude variations. Consequently, higher altitudes dominate lower altitudes, as they provide better coverage statistics "for free".

2.5 Case Study 2: The versatility of the proposed evolutionary formulations

The purpose of this second case study is to evaluate the ability of the hybrid chromosome to adapt to different constellation design problems. This is done because the rationale behind the hybrid chromosome is its capability of considering a very broad design space of

hybrid/heterogeneous constellation designs and finding the optimal areas of this very wide constellation space within a low number of function evaluations. Consequently, the mission designer will no longer be responsible for choosing a narrower design space with the risk of missing potentially good architectures.

2.5.1 Constellation design problems

To show the versatility of the hybrid chromosome, in this second case study, 5 different coverage problems were solved using the same hybrid constellation design space formed by 4 base constellations: a high inclination Walker, a low inclination Walker, a heterogeneous planes configuration and a Sun synchronous Train, with a combined maximum of 20 satellites. The difference between the 5 coverage problem was their regions of interest:

- Global coverage. 510 point grid with points of interest distributed across latitudes between -90° and 90° and longitudes between -180° and 180°.

- Tropics coverage. 237 point grid with points of interest distributed across latitudes between -30° and 30° and longitudes between -180° and 180°.

- Tropics and Poles coverage. 315 point grid with points of interest distributed across three regions of interest: (1) Tropical region, with latitudes between -30° and 30° and longitudes between -180° and 180° (2) South Pole, with latitudes between -90° and -60° and longitudes between -180° and 180° and (3) Nort Pole, with latitudes between 60° and 90° and longitudes between -180° and 180° .

- Continental US and Europe coverage. 140 point grid with points of interest distributed across two regions of interest: (1) Continental US, with latitudes between 25° and 50° and longitudes between -125° and -65° and (2) Europe, with latitudes between -10° and 45° and longitudes between 35° and 70°.

- 7 locations coverage. 7 point grid consisting of 7 randomly chosen locations in the Earth: (1) Washington DC (38° N, 77° W), (2) Honolulu (21° N, 157° W), (3) Brasilia (15° S, 47° W), (4) Abu Dhabi(24° N, 54° E), (5) Moscow (55° N, 37° E), (6) Pyongyang (39° N, 125° E) and (7) Mediterranean Sea (41° N, 11° E)

For all coverage problems, the number of points at each latitude was chosen to be proportional to the cosine of the latitude to obtain equal horizontal distances between points. Therefore, fewer points are placed in higher latitudes to avoid statistically weighting more the poles in the global coverage metrics. Also, by solving these 5 different coverage problems we are also testing the performance of the hybrid chromosome to search through the space of constellation designs to cover symmetrical, asymmetrical, connected and disconnected regions of interest.

The hybrid chromosome was used to solve a multi-objective constellation design problem with four objectives selected in order to maximize coverage performance while minimizing overall cost. These selected objectives were the same as in the previous case study and include:

- Minimize mean revisit time, which is the average length of the gap intervals for all the points of interest seen by the constellation.

- Minimize mean response time, which is defined as the time average of the response time. Response time corresponds to the time from when a random request is received to observe a point of interest until the constellation can actually observe it. Mean response time is known to be more sensitive to the number of planes than mean revisit time, so there can be a trade-off between these two parameters [32].

- Maximize the ratio between the number of points observed by the constellation and the total number of points in the region/s of interest. Since mean revisit time and mean

response time are calculated only on observed points and do not take into account unobserved points, this third coverage metric was included as objective in order to penalize those designs that observe fewer points in the area of interest.

- Minimize life-cycle cost, which includes launch, program, integration, ground, hardware, recurring and non-recurring costs.

As it was done in the previous case study in Section 2.4, the TAT-C ML tool was used to evaluate each of the constellation designs and obtain their respective objective values. Again, only low-Earth, circular orbits are considered and each satellite in the constellation carries the same instrument with a rectangular field of view. In the first three coverage problems (Global, Tropics and Tropics + Poles) the FOV defined by a half cross-track angle of 57° and a half along-track angle of 20° and in the last two coverage problems (Continental US + Europe and 7 locations) a narrower FOV was considered with half cross-track and half along-track angles of 10°. In all 5 coverage problems, the satellites are propagated for a week using the GMAT [105] orbital propagator included in TAT-C ML.

Table 2.5 shows the possibles values/ranges of the design variables for each of the 4 base constellations forming the hybrid design space used to solve the 5 multi-objective coverage problems considered. The 4 base constellations forming the hybrid architecture that contains a maximum of 20 satellites are the following:

- A low inclination Walker constellation containing up to 4 satellites with an altitude between 400 km and 800 km and inclination between 0° and 30°.

- A high inclination Walker constellation containing up to 4 satellites with an altitude between 400 km and 800 km and inclination values of 51.6°, 90° or SSO.

- A heterogeneous Planes constellation with up to 8 satellites and 8 different planes. Each of these planes is located at an altitude of 400km, 500km, 600km, 700km or

800km, and at an inclination of 0°, 10°, 20°, 30°, 51.6°, 90° or SSO. By breaking the Walker pattern symmetry, this formulation opens up the tradespace to a larger set of heterogeneous constellation designs.

- A SSO Train with up to 4 satellites, with an altitude between 400 km and 800 km. two possible LTAN of 10:30:00 and 13:30:00 and temporal spacing between two consecutive satellites of 30s, 60 or 120s.

base constellation	**Decision**	**Allowed values**
Walker	Number of satellites	0-4
	Number of planes	1-4
	Altitude	400, 500, 600, 700, 800 km
	Inclination	0°, 10°, 20°, 30°
Walker	Number of satellites	0-4
	Number of planes	1-4
	Altitude	400, 500, 600, 700, 800 km
	Inclination	51.6°, 90°, SSO
Heterogeneous planes	Number of satellites	0-8
	Number of planes	1-8
	Altitude	400, 500, 600, 700, 800 km
	Inclination	0°, 10°, 20°, 30°, 51.6°, 90°, SSO
SSO-Train	Number of satellites	0-4
	Altitude	400, 500, 600, 700, 800 km
	Reference LTAN	10:30, 13:30
	Satellite spacing	30, 60, 120 s

Table 2.5: Hybrid constellation design space used to solve all 5 coverage problems, formed by 4 base constellations: a high inclination Walker, a low inclination Walker, a heterogeneous planes configuration and a Sun synchronous Train, with a combined maximum of 20 satellites.

Again, the idea of this second case study is to observe if the hybrid chromosome is able to find the different hybrid/non-hybrid designs that offer the best performance for each of the 5 different coverage problems. This will allow to prove the versatility of the hybrid

chromosome to adapt to different constellation design problems, initially considering the same very broad design space of hybrid/heterogeneous constellation designs and freeing the mission designer from the responsibility of choosing a narrower design space with the risk of missing potentially good architectures.

2.5.2 Simulation setup

A total number of 5 optimization runs were performed for each of the five coverage problems since, again, the search carried out by an EA is stochastic and results vary from run to run. As in the first case study, the ϵ-MOEA algorithm [106] was used as the baseline algorithm. The probability of the operators to apply crossover for fixed-length base constellation chromosomes and $cut\ and\ splice$ or $pairing$ for variable-length base constellation chromosomes was set to $p_c = 0.9$. The probability of performing mutation for each variable (e.g., altitude, inclination, etc.) of the resulting hybrid chromosomes was adaptably set to change one variable in the chromosome in average to favor the exploration of the genetic algorithm. An initial population of 500 random hybrid architecture designs was created at the beginning of each optimization run. Finally, the termination criteria was extended to 5,000 function evaluations ($NFE = 5000$) since the design space considered now is much larger than in the previous case study in Section 2.4.

After running the total number of 5 simulations for each coverage problem, the 25,000 architectures generated were joined together to compute the maximum and minimum values obtained for all 4 objectives, which were later used to scale between 0 and 1 the objective values of all the enumerated designs. Finally, the final combined Pareto Fronts for each of the 5 coverage problems were computed to assess the differences between them. Indeed, data mining algorithms are very useful to discover relationships and patterns in data, and suitable to analyze and compare the compositions of the 5 resulting combined Pareto fronts. Specifically, in this case study, association rule mining was used to study the design features

that appear in the different Pareto fronts and reveal data patterns associated with the optimal region in the objective space for each of the 5 coverage problems solved. In particular, the FP-Growth algorithm [109] was used to generate associating rules to help describe the architectures found in the different Pareto fronts. Contrarily to the famous apriori frequent pattern mining algorithm, FP-Growth internally uses a FP-tree (frequent pattern tree) data structure without generating the candidate sets explicitly, which makes it specially attractive for large data sets.

Association rule mining is successful at extracting common patterns of decision variables that are more likely to occur in high quality/optimal solutions by assigning class labels to solutions or designs based on their quality/optimality. In this particular case study, rules of the type $A \rightarrow P$, where A is a design feature (e.g. hybrid/non-hybrid constellation type) and P is a class label that determines the quality of a solution (e.g. presence in the Pareto Front). This way, given a set of architectures for each of the 5 coverage problems, it is possible to study what hybrid (or non-hybrid) architectures offer the best performance in each scenario. When applying association rule mining algorithms to large data sets, there can be an enormous number of generated rules. In order to filter interesting rules, special metrics are used to help focus our attention on those rules that efficiently capture the patterns and relationships between a given set of solutions. The most common ones are the following:

- Support $(0 \leq support(A) \leq 1)$. The support of a feature A is defined as the fraction of the total space of solutions that contains the feature and defined as:

$$support(A) = \frac{|S_A|}{|S|}, \tag{2.2}$$

  where $|\cdot|$ is the cardinality operator, S_A is the set of solutions in S that contain design feature A. A high $support(A)$ indicates that A is naturally present in the set S. Similarly, the support of a rule $A \rightarrow P$ is the fraction of solutions in a set S that contains a

design feature A and also labeled with class P and defined as:

$$support(A \rightarrow P) = \frac{|S_A \cap S_P|}{|S|}, \tag{2.3}$$

where S_A again is the set of solutions in S that contain design feature A and S_P are the solutions in S labeled with class P.

- Confidence $(0 \leq confidence(A \rightarrow P) \leq 1)$. The confidence of a rule $A \rightarrow P$ is the probability of a solution being labeled with class P given that the solution contains design feature A. It is defined as:

$$confidence(A \rightarrow P) = \frac{support(A \rightarrow P)}{support(A)} = \frac{|S_A \cap S_P|}{|S_A|}. \tag{2.4}$$

  A high $confidence(A \rightarrow P)$ suggests that A is a quasi sufficient design feature for a solution to be labeled with P. In parallel, a high $confidence(P \rightarrow A)$ suggests that A is a quasi necessary design feature for a solution to be labeled with P.

- Lift $(0 \leq lift(A \rightarrow P) < \infty)$. The lift of a rule $A \rightarrow P$ is used to measure how much more often the design feature A and a solution being labeled with P occur together than we would expect if they were statistically independent. If A and P are independent, the lift score will be exactly 1. Lift is defined as the confidence of the rule normalized by the support of P:

$$lift(A \rightarrow P) = \frac{confidence(A \rightarrow P)}{support(P)} = lift(P \rightarrow A). \tag{2.5}$$

  A high lift indicates that A and P are probably not statistically independent and, therefore, it exists some sort of statistical relationship between design feature A and class P.

Every design generated by the EA in all runs from the 5 coverage problems was associated to a label corresponding to their hybrid or non-hybrid constellation type. These labels are the following:

1. ***'LowIncWalker'***. Label corresponding to a low inclination $(inc \leq 30^{\circ})$ walker constellation.

2. ***'HighIncWalker'***. Label corresponding to a high inclination $(inc \geq 30^{\circ})$ walker constellation.

3. ***'HeteroHigh'***. Label corresponding to a heterogeneous planes constellation with all planes at a high inclination $(inc \geq 30^{\circ})$.

4. ***'HeteroLow'***. Label corresponding to a heterogeneous planes constellation with all planes at a low inclination $(inc \leq 30^{\circ})$.

5. ***'HeteroMix'***. Label corresponding to a heterogeneous planes constellation which mixes planes at low $(inc \leq 30^{\circ})$ and high $(inc \geq 30^{\circ})$ inclinations.

6. ***'Train'***. Label corresponding to a SSO train constellation.

7. **Multiple** Hybrid constellation labels. If any of the designs is formed by 2 or more base constellations of the above mentioned labels, a hybrid label is created with the different base constellation labels separated by a $+$ sign. For instance, if an enumerated design is formed by a low inclination Walker and a SSO train, the label associated to that architecture would be *'LowIncWalker+Train'*.

Finally, a second label *'InParetoFront'* was added to all the optimal designs that are part of the 5 final combined Pareto Fronts of the different coverage problems solved. For this analysis, all the architectures that provided a percent coverage less than 100% were filtered

out since constellations that do not cover all points of interest are inherently poor architecture designs.

2.5.3 Results

To assess the differences between the results obtained in each of the following scenarios with the ultimate goal of evaluating the ability of the hybrid chromosome to adapt to different coverage problems, two post-optimization analysis based on data mining were done:

1. For each coverage problem, the support of the constellation type label was assessed in the Pareto Front set (i.e. only the optimal designs). This analysis will tell us what are the hybrid/non-hybrid configurations of the constellation designs that are present in each of the 5 combined Pareto Fronts.

2. For each coverage problem, associating rule mining was run to generate rules of the type $A \rightarrow P$, where A is a design feature corresponding to the label that describes the hybrid/non-hybrid constellation type and P is the label that determines the presence of the solution in the Pareto Front (i.e. 'InParetoFront'). The rules were generated over the set if ALL enumerated designs (regardless of their quality or presence in the optimal set) and were ordered in descending order based on the lift metric. In this analysis, we are essentially trying to find the rules with high lift since that indicates there is statistical dependence between a certain design feature A and optimality (or presence in the Pareto front).

2.5.3.1 Whole Earth

Table 2.6 shows the composition of the combined Pareto Front for the whole Earth coverage problem. It can be observed that the optimal set contains a few different hybrid configurations that mix Walker, heterogeneous planes constellations and trains, with satellites both at high and low inclinations with the objective of covering all the globe. 22.09% of the

designs found in the Pareto Front correspond to hybrid designs formed by a heterogeneous planes constellation with all planes at high inclination, a high inclination Walker constellation and a low inclination Walker. Other hybrid configurations mixing Walkers at high and low inclination plus a SSO train are also present in the Pareto Front. There are also non-hybrid designs with just a high inclination heterogeneous planes constellation (2.71%) or a high inclination Walkers (3.49%).

support	feature
0.220930	(HeteroHigh+HighIncWalker+LowIncWalker)
0.127907	(HighIncWalker+LowIncWalker+Train)
0.096899	(HeteroHigh+HighIncWalker+LowIncWalker+Train)
0.093023	(HeteroHigh+HighIncWalker)
0.077519	(HighIncWalker+LowIncWalker)
0.065891	(HeteroMix+HighIncWalker)
0.054264	(HeteroMix+HighIncWalker+LowIncWalker)
0.054264	(HeteroMix+HighIncWalker+LowIncWalker+Train)
0.050388	(HighIncWalker+Train)
0.038760	(HeteroHigh+HighIncWalker+Train)
0.034884	(HighIncWalker)
0.027132	(HeteroHigh)
0.019380	(HeteroMix+HighIncWalker+Train)
0.019380	(HeteroHigh+LowIncWalker)
0.011628	(HeteroHigh+Train)

Table 2.6: Support of the constellation type labels in the Pareto Front set for the whole Earth coverage problem.

That being said, the results presented need some further analysis since, when looking at Table 2.6, just a high inclination Walker constellation apriori seems like very good architecture to track a symmetrical region of interest that covers all latitudes. However, it is shown that less than 4% of the optimal architectures have these configurations –and this could be seen as bad– but in reality, what happens is that combinatorially speaking there are many less High inclination Walkers designs than hybrids combining multiple base constellations. Consequently, Table 2.7 provides the association rules with highest lift generated

by the FP-growth algorithm to better study what design features (or constellation configurations) describe better the designs found in the Pareto Front. It can be observed that the rules $HeteroHigh \rightarrow InParetoFront$ and $HighIncWalker \rightarrow InParetoFront$ are the ones with the highest lift, which indicates that non hybrid designs with just a high inclination heterogeneous planes configuration or just a high inclination walker are highly correlated to optimality for the whole Earth coverage problem. This makes a lot of sense since these non-hybrid designs are able to offer good coverage performance observing all latitudes of interest at a lower cost than hybrid configurations.

antecedents	consequents	support	confidence	lift
(HeteroHigh)	(InParetoFront)	0.000479	0.368421	20.858629
(HighIncWalker)	(InParetoFront)	0.000616	0.089109	5.045015
(HeteroHigh+HighIncWalker)	(InParetoFront)	0.001643	0.083045	4.701698
(HeteroHigh+HighIncWalker+LowIncWalker)	(InParetoFront)	0.003902	0.070370	3.984109
(HeteroHigh+Train)	(InParetoFront)	0.000205	0.069767	3.949973
(HeteroHigh+LowIncWalker)	(InParetoFront)	0.000342	0.032680	1.850205
(HeteroMix+HighIncWalker)	(InParetoFront)	0.001164	0.032381	1.833289
(HeteroHigh+HighIncWalker+Train)	(InParetoFront)	0.000685	0.028169	1.594825
(HeteroHigh+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.001712	0.026261	1.486772
(Train)	(InParetoFront)	0.000068	0.025000	1.415407
(HighIncWalker+Train)	(InParetoFront)	0.000890	0.017173	0.972274
(HeteroMix+HighIncWalker+LowIncWalker)	(InParetoFront)	0.000958	0.013372	0.757047
(HighIncWalker+LowIncWalker)	(InParetoFront)	0.001369	0.012763	0.722607
(HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.002259	0.009737	0.551295
(HeteroMix+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.000958	0.007968	0.451126
(HeteroMix+HighIncWalker+Train)	(InParetoFront)	0.000342	0.006964	0.394264
(HeteroHigh+LowIncWalker+Train)	(InParetoFront)	0.000068	0.005291	0.299557

Table 2.7: Association rules generated by the FP-growth algorithm, ordered in descending order based on the lift metric, for the whole Earth coverage problem.

2.5.3.2 Tropics

Table 2.8 shows the composition of the combined Pareto Front for the the TROPICS coverage problem. It can be observed that almost half of the designs in the Pareto Front contain

a hybrid configurations that mix a low inclination Walker and a heterogeneous planes with all satellites at low inclinations with the objective of covering the tropical regions. Roughly 16% of the designs found in the Pareto Front correspond to cheaper non-hybrid designs with just a low inclination heterogeneous planes constellation or a low inclination Walker since now we are only interested in observing lower latitudes.

support	feature
0.478528	(HeteroLow+LowIncWalker)
0.141104	(HeteroLow+HighIncWalker+LowIncWalker)
0.079755	(LowIncWalker)
0.079755	(HeteroLow)
0.061350	(HighIncWalker+LowIncWalker)
0.036810	(LowIncWalker+Train)
0.036810	(HeteroLow+HighIncWalker+LowIncWalker+Train)
0.030675	(HeteroLow+HighIncWalker)
0.024540	(HeteroLow+Train)
0.024540	(HeteroLow+LowIncWalker+Train)

Table 2.8: Support of the constellation type labels in the Pareto Front set for the Tropics coverage problem.

When looking at the generated association rules in the Tropics coverage problem in Table 2.9, it can be observed that the rules $LowIncWalker \rightarrow InParetoFront$ and $HeteroLow \rightarrow InParetoFront$ are the ones with the highest lift, which indicates that non-hybrid designs with just a heterogeneous planes configuration with all planes at low inclination or configurations with just a low inclination Walker are highly correlated to optimality for the whole Tropics coverage problem. As in the whole Earth coverage problem, to track a symmetric non-disconnected region of interest, these non-hybrid designs are able to offer good coverage performance-cost trade-off. As also observed in Table 2.8, the hybrid configuration with highest lift and therefore better describes the optimal designs for this coverage problem is the one that mixes a low inclination Walker and a heterogeneous planes with all satellites at

low inclinations.

antecedents	consequents	support	confidence	lift
(LowIncWalker)	(InParetoFront)	0.000931	0.139785	11.970051
(HeteroLow)	(InParetoFront)	0.000931	0.128713	11.021928
(HeteroLow+LowIncWalker)	(InParetoFront)	0.005588	0.040583	3.475176
(HeteroLow+Train)	(InParetoFront)	0.000287	0.033898	2.902776
(HeteroLow+HighIncWalker)	(InParetoFront)	0.000358	0.029240	2.503857
(HeteroLow+HighIncWalker+LowIncWalker)	(InParetoFront)	0.001648	0.016347	1.399811
(HeteroLow+HighIncWalker+Train)	(InParetoFront)	0.000072	0.008475	0.725694
(HeteroLow+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.000430	0.008032	0.687806
(LowIncWalker+Train)	(InParetoFront)	0.000430	0.006141	0.525887
(HighIncWalker+LowIncWalker)	(InParetoFront)	0.000716	0.005485	0.469731
(HeteroLow+LowIncWalker+Train)	(InParetoFront)	0.000287	0.005464	0.467934

Table 2.9: Association rules generated by the FP-growth algorithm, ordered in descending order based on the lift metric, for the Tropics coverage problem.

2.5.3.3 Tropics and Poles

Table 2.10 shows the support of constellation type labels in the Pareto Front set for the Tropics and Poles. It can be observed that most of the optimal designs correspond to hybrid designs mixing high and low inclination base constellations. In these configurations, the low inclination satellites will track the tropics while the high inclination satellites will track the Poles. For instance, 22.69% of the designs found in the Pareto Front correspond to hybrid designs formed by a high inclination Walker constellation and a low inclination Walker. 21.3% of the designs in the Pareto Front add a train to this hybrid configuration to provide better coverage performance. Finally, there also appears a non-hybrid design (4.63% of the optimal designs) with just a high inclination Walker which allows to track all the points in the three disconnected regions of interest at a lower cost.

The generated association rules with highest lift for the Tropics and Poles problem, shown in Table 2.11, indicate that non hybrid designs with just a high inclination Walker

support	feature
0.226852	(HighIncWalker+LowIncWalker)
0.212963	(HighIncWalker+LowIncWalker+Train)
0.087963	(HeteroMix+HighIncWalker+LowIncWalker)
0.083333	(HeteroMix+HighIncWalker+LowIncWalker+Train)
0.064815	(HeteroLow+HighIncWalker+LowIncWalker+Train)
0.050926	(HeteroHigh+LowIncWalker+Train)
0.046296	(HighIncWalker)
0.046296	(HeteroHigh+LowIncWalker)
0.046296	(HeteroHigh+HighIncWalker+LowIncWalker)
0.027778	(HeteroHigh+HighIncWalker+LowIncWalker+Train)
0.023148	(HeteroLow+HighIncWalker+Train)
0.023148	(HeteroMix+LowIncWalker+Train)
0.018519	(HeteroLow+HighIncWalker+LowIncWalker)
0.013889	(HighIncWalker+Train)

Table 2.10: Support of the constellation type labels in the Pareto Front set for the Tropics and Poles coverage problem.

constellation are probably a good cheap design alternative to cover all points of interest of the three disconnected region of interest and provide a decent coverage performance. Similarly, the hybrid configurations with highest lift are the ones that mix a low inclination Walker to track the tropics, a heterogeneous planes constellation with all satellites at high inclinations to track the Poles and an optional SSO train to boost coverage performance.

2.5.3.4 Continental US and Europe

Similarly to the Tropics and Poles coverage problem, the Continental US and Europe disconnected region of interest is better tracked by hybrid designs mixing high and low inclination satellites. As shown in Table 2.12, 29.13% of the designs found in the Pareto Front correspond to hybrid designs formed by a high inclination Walker constellation and a low inclination Walker. Likewise, 15.53% of the designs in the Pareto Front add a train to this hybrid configuration to provide better coverage performance. Finally, there also appears a non-hybrid design (3.88% of the optimal designs) with just a high inclination Walker which allows to track all the points in the two disconnected regions of interest at a more reasonable

antecedents	consequents	support	confidence	lift
(HighIncWalker)	(InParetoFront)	0.000718	0.101010	6.517022
(HeteroHigh+LowIncWalker+Train)	(InParetoFront)	0.000789	0.088000	5.677630
(HeteroHigh+LowIncWalker)	(InParetoFront)	0.000718	0.087719	5.659519
(Train)	(InParetoFront)	0.000144	0.052632	3.395712
(HeteroLow+HighIncWalker+Train)	(InParetoFront)	0.000359	0.036496	2.354690
(HeteroHigh+HighIncWalker+LowIncWalker)	(InParetoFront)	0.000718	0.034965	2.255892
(HeteroMix)	(InParetoFront)	0.000072	0.029412	1.897603
(HeteroLow+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.001005	0.027944	1.802913
(HighIncWalker+LowIncWalker)	(InParetoFront)	0.003516	0.022727	1.466330
(HeteroMix+HighIncWalker+LowIncWalker)	(InParetoFront)	0.001363	0.018304	1.180975
(HeteroHigh+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.000431	0.012712	0.820151
(HeteroMix+LowIncWalker+Train)	(InParetoFront)	0.000359	0.012690	0.818763
(HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.003301	0.012036	0.776518
(HeteroMix+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.001292	0.010746	0.693333
(HeteroLow+HighIncWalker+LowIncWalker)	(InParetoFront)	0.000287	0.010610	0.684547
(HeteroMix+HighIncWalker)	(InParetoFront)	0.000144	0.008264	0.533211
(HighIncWalker+Train)	(InParetoFront)	0.000215	0.005505	0.355148
(LowIncWalker+Train)	(InParetoFront)	0.000072	0.001464	0.094463

Table 2.11: Association rules generated by the FP-growth algorithm, ordered in descending order based on the lift metric, for the Tropics and Poles coverage problem.

cost.

support	feature
0.291262	(HighIncWalker+LowIncWalker)
0.155340	(HighIncWalker+LowIncWalker+Train)
0.126214	(HeteroHigh+HighIncWalker+LowIncWalker)
0.106796	(HeteroHigh+HighIncWalker+LowIncWalker+Train)
0.077670	(HeteroHigh+LowIncWalker+Train)
0.058252	(HeteroHigh+LowIncWalker)
0.038835	(HighIncWalker)
0.029126	(HeteroLow+HighIncWalker+LowIncWalker+Train)
0.019417	(HeteroLow+HighIncWalker+LowIncWalker)
0.019417	(HeteroMix+LowIncWalker+Train)
0.019417	(HeteroLow+HighIncWalker+Train)
0.019417	(HeteroHigh+HighIncWalker+Train)

Table 2.12: Support of the constellation type labels in the Pareto Front set for the continental US and Europe coverage problem.

When looking at the generated association rules of this fourth coverage problem in Table 2.13, it can be observed that the rules $HighIncWalker \rightarrow InParetoFront$ and $HeteroHigh \rightarrow InParetoFront$ are the ones with the highest lift, which indicates that non-hybrid designs with just a heterogeneous planes configuration with all planes at a high inclinations or configurations with just a high inclination Walker are highly correlated to the fact of being present in the optimal set of solutions. Other high lift rules such as $HeteroHigh + LowIncWalker \rightarrow InParetoFront$ and $HeteroHigh+HighIncWalker+LowIncWalker \rightarrow InParetoFront$ indicate that hybrid designs mixing low inclination Walkers with high inclination heterogeneous plane constellations are also good design alternatives to track Europe and the US.

antecedents	consequents	support	confidence	lift
(HighIncWalker)	(InParetoFront)	0.000609	0.125000	7.975728
(HeteroHigh)	(InParetoFront)	0.000152	0.125000	7.975728
(HeteroHigh+LowIncWalker)	(InParetoFront)	0.000913	0.067416	4.301516
(HeteroHigh+HighIncWalker+LowIncWalker)	(InParetoFront)	0.001978	0.064356	4.106315
(HighIncWalker+LowIncWalker)	(InParetoFront)	0.004565	0.037083	2.366100
(HeteroHigh+LowIncWalker+Train)	(InParetoFront)	0.001217	0.035714	2.278779
(HeteroHigh+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.001674	0.024664	1.573686
(HeteroMix+Train)	(InParetoFront)	0.000152	0.023810	1.519186
(HeteroHigh+HighIncWalker+Train)	(InParetoFront)	0.000304	0.023529	1.501314
(HeteroLow+HighIncWalker+Train)	(InParetoFront)	0.000304	0.021277	1.357571
(HeteroLow+HighIncWalker+LowIncWalker)	(InParetoFront)	0.000304	0.020202	1.289007
(HeteroHigh+HighIncWalker)	(InParetoFront)	0.000152	0.013158	0.839550
(HeteroMix+LowIncWalker+Train)	(InParetoFront)	0.000304	0.011905	0.759593
(HeteroLow+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.000456	0.009901	0.631741
(HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.002435	0.008538	0.544767
(HeteroMix+HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.000152	0.001029	0.065644

Table 2.13: Association rules generated by the FP-growth algorithm, ordered in descending order based on the lift metric, for the continental US and Europe coverage problem.

2.5.3.5 7 locations

Finally, Table 2.14 displays the support of the constellation type labels in the Pareto Front set for the 7 locations coverage problem. We observe that more than 90% of the architectures in the optimal set are formed by a Walker constellation and a Train (73.17% include a low inclination Walker and the remaining 19.15% includes a high inclination Walker instead). A non-hybrid design with just a high inclination Walker also appears in the Pareto Front.

support	feature
0.731707	(LowIncWalker+Train)
0.195122	(HighIncWalker+Train)
0.048780	(HighIncWalker+LowIncWalker+Train)
0.024390	(HighIncWalker)

Table 2.14: Support of the constellation type labels in the Pareto Front set for the 7 locations coverage problem.

The generated association rule with highest lift for this last coverage problem, shown in Table 2.15, indicates that the hybrid configuration that mixes a low inclination Walker and a train is the one that better explains the solutions in the Pareto Front.

antecedents	consequents	support	confidence	lift
(LowIncWalker+Train)	(InParetoFront)	0.006225	0.069124	8.124649
(HighIncWalker)	(InParetoFront)	0.000208	0.027778	3.264905
(HighIncWalker+Train)	(InParetoFront)	0.001660	0.008859	1.041299
(HighIncWalker+LowIncWalker+Train)	(InParetoFront)	0.000415	0.001266	0.148780

Table 2.15: Association rules generated by the FP-growth algorithm, ordered in descending order based on the lift metric, for the 7 locations coverage problem.

So what is the main conclusion of this data mining analysis? The main point is that

exploring the same tradespace of hybrid constellations for 5 different coverage problems, the proposed evolutionary formulation (hybrid chromosome and respective operators) allows to guide the optimization search towards different regions of the tradespace. This proves the point that the proposed method adapts to the coverage problem in hand and frees the designer from the responsibility of choosing a narrower design space with the risk of missing potentially good architectures.

2.6 Conclusion

This work introduced the concept of hybrid constellations using a new evolutionary formulation which allows to explore a very wide tradespace of constellation designs never explored before. In the first case study, two instances of this new evolutionary formulation (hybrid Walker and heterogeneous planes) were used to study heterogeneous constellations mixing satellites at different altitudes and inclinations and compare the performance of the proposed hybrid constellations evolutionary formulations. So far, this part of the constellation design space had been understudied and this thesis showed that it has potential to be a cost-efficient way of satisfying mission requirements for certain Earth observing applications. By solving a constellation design problem optimizing mean revisit time, mean response time, coverage and cost, it was shown that these evolutionary formulations obtain a higher convergence rate and diversity (measured by the final hypervolume) than a third formulation already existing in the literature, which used a variable-length chromosome with continuous variables representing the 4 orbital elements of each satellite.

Also, no previous work had provided an evolutionary formulation with similar level of performance (convergence rate and hypervolume) that adapts to the nature of the coverage problem in hand (connected/disconnected regions of interest and wide/small FOV). In the second case study, 5 different coverage problems (global coverage, tropics coverage, tropis + poles coverage, continental US + Europe coverage and 7 different locations coverage)

were solved considering the same tradespace of hybrid constellations (including combinations of Walker, heterogeneous and train base constellations). The results showed the ability of the proposed hybrid evolutionary formulations to find what combinations of these base constellations work best in each coverage problem. This property of the hybrid chromosome considerably helps the mission designer, who will no longer be responsible for choosing a narrower design space – which could potentially depend on the application/coverage problem to solve– with the risk of missing potentially good architectures.

The termination criteria of the multi objective evolutionary algorithm was set to a maximum number of function evaluations equal to 1,000 in the first case study and 5,000 in the second. Considering tradespaces that have millions or even billions of possible architectures, tenths of thousands of function evaluations are clearly not enough for the algorithm to converge. In fact, in both case studies the hypervolume kept increasing until the end of the simulations. Moreover, in the first case study, the three studied formulations explore tradespaces of different sizes. More specifically, the design space in formulation 3 is larger than the one in formulation 2, whose tradespace size is larger than the one in formulation 1. Consequently, the general formulation might require more function evaluations to converge than any of the other two, and so does formulation 2 with respect to 1. For all these reasons, another opportunity for future work would be analyzing the same problem for a larger number of function evaluations to see if the formulations exploring larger tradespaces eventually find better designs.

There is also room to improve the chromosome representation of hybrid constellations, which could be made variable-length to avoid having to set the number of satellites t to 0 to turn "off" a certain base constellation, which constitutes a source of redundancy in the chromosome representation. Moreover, this fix would allow to not having to specify the maximum number of base constellations in the hybrid architecture (e.g. in the constellation design problem presented in Section 2.4.1, the maximum number of Walkers was set to 3).

On the other hand, the heterogeneous plane base constellation assumes equal number of satellites in each plane. This design constraint could be also removed to include even less symmetric architectures. Lastly, to further open up the tradespace and to better exploit the advantages of constellations that mix satellites at different altitudes and inclinations, future work may also include the possibility of considering constellations composed by spacecraft carrying distinct instruments.

Finally, the creation of other operators and the use of credit assignment strategies in multi objective adaptive operator selection [110, 111, 112] could help get better convergence rates but, again, the algorithmic details were not the focus of this work.

3. AUTONOMOUS DELAY TOLERANT NETWORK MANAGEMENT USING REINFORCEMENT LEARNING*

3.1 Introduction

DTN node management is necessary to guarantee the correct operation of DTN protocols, which generally assume that there is enough memory available to store and forward data as bundles arrive to the different nodes in the system. In this work, we propose a Reinforcement Learning (RL) formulation for DTN node management and demonstrate its efficiency in a realistic lunar scenario using a medium-fidelity simulator. The high-level goal of the DTN node is to avoid memory overflows and maximize quality of service, including probability of successful data delivery to the Deep Space Network (DSN), while minimizing the amount of system resources provisioned. RL is a reasonable approach to manage a DTN node thanks to its ability to adapt to changes in the network, which can be triggered at any point in time by different circumstances. For example, the traffic conditions could unexpectedly change due to the failure of a network node. Furthermore, prior work has showed that RL outperform more traditional approaches based on queuing theory, which only offer steady-state formulations, to deal effectively with transient states in other networking and communication problems [97]. In this context, the RL agent perceives changes in the node's and the network's state and triggers actions to optimize its performance or protect itself against anticipated failures. The RL agent chooses from a set of actions which include dropping incoming packets, increasing or decreasing the data rate of the different neighbor node links, modifying bundle routes to use crosslinks instead of a direct route to the ground,

*Part of this chapter is reprinted with permission from "Autonomous Delay Tolerant Network Management Using Reinforcement Learning" by Pau Garcia Buzzi, Daniel Selva and Marc Sanchez Net, 2021. Journal of Aerospace Information Systems, Volume 18, Number 7, pp. 404-416 Copyright 2021 by the American Institute of Aeronautics and Astronautics, Inc.

or simply not altering any network parameter. The reward function is designed in such a way that the agent maximizes the number of bits received by the Deep Space Network (DSN) while minimizing the capacity allocated to all controlled links, and controlling buffer utilization to avoid memory overflows. Additionally, in order to assess the potential of using RL for DTN management, the performance of our best trained agent is benchmarked against other non-RL based policies including a random policy, a reference policy that maximizes all data rates and does not take any action, and a ruled-based expert policy.

Therefore, the primary contribution of this chapter is the formulation of an RL model to autonomously manage a Delay Tolerant Network minimizing memory overflows, anticipating networks failures and dynamically adapting to the different conditions of the network. In order to validate this formulation, we implemented it in a software tool based on an existing Python-based simulator for DTN called DtnSim [113], and benchamrked the performance of the best RL policies against other more traditional non-RL policies used in network management in a rich and realistic lunar scenario.

The remainder of this chapter is organized as follows: Section 3.2 contains a literature review of memory management in DTNs and how RL algorithms have been applied to communications and networking, as well as very brief introductions to Q-Learning and Deep Q-Learning (DQL). Section 3.3 presents the proposed system model and RL problem formulation, explaining in detail the state space, the action space and the reward function. Section 3.4 presents the experimental setup used for validation, including a description of the Lunar exploration mission case study used for training the RL agent and benchmarking its performance against the other non-RL approaches. This section also includes a brief description of the DTN simulator used to recreate the DTN environment as well as all the parameters used for the training/evaluation of the RL agent. Finally, Sections 3.5 and 3.8 present the results, and the conclusions and opportunities for future work, respectively.

3.2 Background and Literature Review

3.2.1 Memory management in DTN

Memory management and buffer dimensioning are important to ensure successful operation of the DTN communication protocols. Most studies found in the literature that evaluate the performance of the DTN protocol stack assume infinite available memory [114, 115, 116, 117, 118]. In other words, these works consider that having enough memory is an underlying assumption for the DTN protocols to work successfully. However, a problem might arise if the network gets congested and the number of bundles stored in the system buffers increases over time. In that scenario, a DTN node might need to take some actions such as dropping packets with the lowest priority, or altering some network characteristics such as asking for more bandwidth to adapt to this undesirable situation. In [119, 120], Mahendran et al. provide analytically derived expressions to optimize buffer size for an opportunistic DTN under the assumptions of a Poisson traffic and nodes moving following a random waypoint model. These expressions, however, can only be applied to scheduled DTNs in which contacts have arbitrary duration, and data flows exhibit limited variability (changes in the arrival rate). Consequently, in scenarios where node contacts and data flows might change dynamically over time, a more robust memory management strategy is needed. In stochastic network optimization, memory management has also been studied in depth and it is usually associated with ensuring system stability (e.g., if the average input rate is less or equal than the average output rate). Neely [121] presents a survey of topics traditionally tackled in optimization of stochastic communications networks under memory constraints as well as methods for ensuring network stability without requiring prior knowledge of input and output rates. The research gap that we are addressing in this thesis is the lack of existence of a method to autonomously manage buffer utilization in Delay Tolerant Network nodes, since all works in the literature assume nodes have enough memory to store and forward data as

bundles arrive to the different nodes in the system.

3.2.2 Reinforcement learning in communications and networking

Reinforcement learning is a branch in machine learning that has significantly contributed to the development of artificial intelligence over the last few years [93]. In RL, there is an agent which is able to interact with an environment (or world) by performing actions and observing states and rewards. By interpreting the world as a Markov Decision Process (MDP), the goal of RL is to adjust the agent's strategy or policy (which maps states to actions) through a learning or training process to maximize the cumulative expected reward. Except for certain cases where function approximation, bootstrapping and off-policy learning are combined (commonly known as the Deadly triad [93]), RL ensures finding the optimal policy (see the convergence proof of temporal difference learning and the Bellman's equation [93]). Also, since the intelligent agent has to gain insight about an initially unknown environment, this training process can in some cases be very long and resource-expensive and, therefore, inapplicable.

However, with the emergence of Deep Reinforcement Learning (DRL), RL has started to be applied in complex systems including communications and networking over the last decade, tackling issues such as dynamic network access, data rate control, wireless caching and data offloading, networks security, connectivity preservation, traffic routing, resource sharing and data collection [97]. Of specific relevance to this work, Tesauro et al. [122] successfully apply RL to resource allocation in finite capacity systems, where servers in a data center are allocated to different user applications in order to maximize revenue. This work is of special relevance for the work presented in our paper since it was shown that RL, combined with theoretical queuing models, can efficiently learn high quality management policies robust to transients in traffic. However, this was done for a different problem (resource allocation rather than buffer control). Similarly, in [123], Tesauro uses decom-

positional RL for online resource allocation in a distributed multi-application computing environment with independent time-varying load in each application. The author justifies the use of decompositional RL as an alternative to using DRL in systems with very large state/action spaces where tabular Q-learning is infeasible. Their results demonstrate how RL can outperform traditional approaches based on queuing theory, which cannot adapt to dynamic network conditions. This work motivates the use of RL in our problem, where there is no explicit mathematical model for traffic, and thus theoretical queuing theory is very challenging to be applied to the buffer utilization control in DTN nodes. Abdallah and Lesser [124] present a new gradient ascent learning algorithm called Weighted Policy Learner to solve the distributed task allocation problem in a system containing mediators (clients) and servers with different characteristics such as time of task completion. They use the turn-around-time (TAT) as reward signal, which is the time interval between a task arrival and its completion. Similarly, in [125], Li et al. propose a novel model-free DRL approach that learns to control a Distributed Stream Data Processing System by smartly assigning workload to workers/machines (i.e., servers). They minimize average end-to-end processing time and use Apache Storm for data extraction. Their results are compared to the ones provided by the Apache Storm's default scheduler and by a state-of-the-art model-based method. The mathematical structure of the task allocation problem solved in Abdallah and Lesser and Li et al. is indeed akin to the structure of the DTN management problem solved in our paper. Indeed, just like servers with different processing capabilities in the task allocation problem, certain DTN nodes can be more congested or have less contact times with relay orbiters than others, therefore requiring more time to process and deliver packets of information to the next node in the transmission path. The main difference between these two works and ours is that they only focus on optimizing processing time and there does not exist any notion of node congestion since servers can accept unlimited tasks. In our work, besides maximizing the number of processed bundles, we add buffer congestion to our reward function to min-

imize bundle drops in highly congested nodes. Mao et al. [126] present DeepRM, a Deep Q-Network (DQN) solution to the resource management problem in networking (packing tasks with multiple resource demands). They optimize average job slowdown and completion time, and their solution performs comparably or better than state-of-the-art heuristics such as Shortest-Job-First (SJF). Chen et al. [127] improve upon DeepRM by modifying the structure of the state space, changing its reward function, and by adding a convolutional input layer to the function approximator. With these improvements, they show that DRL can outperform more traditional resource allocation algorithms in a variety of complex environments. Similarly to what it is done in Mao et al. and Chen et al., we aim to prove that RL, despite not being as interpretable/explainable, can also outperform more traditional heuristic based policies carefully designed by DTN experts. There exist many other more recent works that also apply RL to the networking scheduling/resource allocation problem [128, 129, 130, 131], but none of them have looked at the buffer utilization management problem in communication nodes studied in this paper.

Finally, Harkavy and Sanchez Net [38] present the first attempt found in the literature to apply RL to DTN node management and introduce the foundational work that motivated this paper. The authors consider the problem of autonomously avoiding memory overflows in a DTN node. To train the autonomous agent, they consider both Deep Q-Network (DQN) and Tabular Q-learning for stabilizing a DTN node with 2 problem formulations of increasing complexity. In both of them, they assumed that the best action would be chosen based on the current state of the environment with the goal of minimizing packet loss. The authors demonstrated the ability of the RL agent to stabilize an M/M/1/K queuing system and to stabilize a nonstationary queuing system with delayed actions. In the present work, we intend to demonstrate the ability of using RL systems to manage a complex DTN node in a realistic simulated scenario consisting of a lunar exploration mission. This DTN node is indeed more complex than a M/M/1/K queuing system, considering that 1) arrivals do not necessarily

occur at a rate λ according to a Poisson process; 2) traffic conditions evolve over time; 3) service time does not necessarily have an exponential distribution with rate parameter μ since, for example, transmission of bundles in a DTN depends on the contact times between nodes, which can be discontinuous. In a previous conference paper [39], we presented a first approach using RL to manage a DTN node in the same realistic lunar exploration mission scenario we are using in this current work. However, the chosen formulation was only able to manage steady-state traffic flows and would fail at attempting to adapt to varying traffic conditions.

3.3 Problem Formulation

3.3.1 System model

Figure 3.1 illustrates the system model used in this work, which focuses on the downlink to Earth. In this thesis we focus on a cis-lunar mission, as detailed in section 3.4.1, but the proposed network model could be also used in other applications, such as Mars exploration missions. The system model consists of the following entities:

- An *intelligent* node where the RL agent is placed. It has a fixed maximum buffer size U_{RL}.

- One or multiple *neighbor* nodes, which send packets of information to the intelligent node. These nodes have infinite memory but they have a virtual maximum buffer size U from the intelligent node's perspective. In other words, even though the neighbor nodes can store unlimited bits of information, after exceeding a certain maximum buffer size U, they will be considered congested by the RL agent. This is needed because the RL agent needs some notion of how the memory of the neighbor nodes is evolving. Without this piece of information, the RL agent would learn greedy policies which would keep the intelligent node at a low memory utilization, while unlimitedly congesting the neighbor nodes (for instance, by setting a high data rate for the outgo-

ing links and a low data rate for the incoming links). This phenomenon of propagating the congestion to other network nodes is known as *back-propagation*[39], and it is mentioned many times throughout this thesis.

- One or multiple *DSN* nodes, which correspond to the endpoints of the downlink.
- One or multiple *relay* nodes (or crosslinks) connected to the intelligent node, which can be used as bridges with the communication with the DSN. Similarly to the neighbor nodes, relay nodes also have infinite memory but they have a virtual buffer capacity U from the RL agent's perspective. The data rate Rb_{cxl} at which the relay nodes transmit packets to the DSN is not controlled by the RL node.

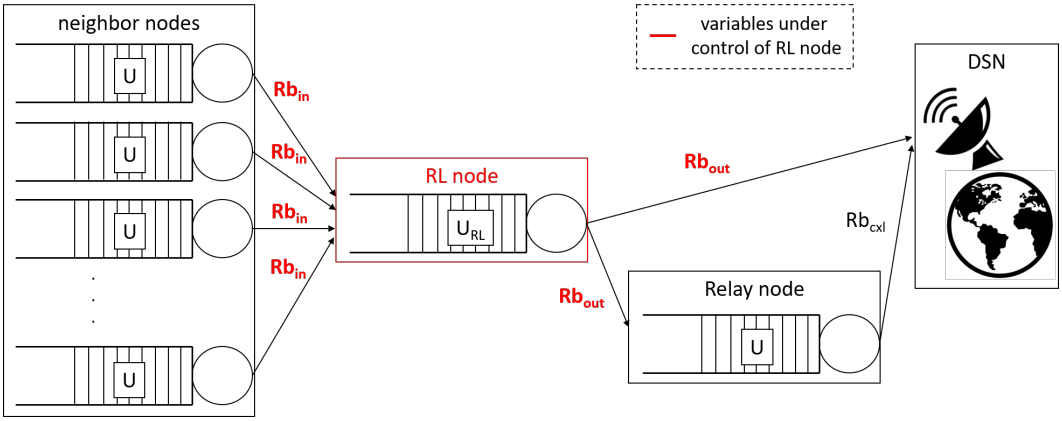


Figure 3.1: Illustration of the proposed system model. The RL node is highlighted in red. Variables controlled by the RL node are in red bold font

The proposed system model makes a few assumptions involving traffic and network environment characteristics. These assumptions are the following:

1. Contacts between nodes in the network are constant since node mobility is not considered.

2. Memory management is performed assuming each node of the system has a single finite memory buffer. In reality, each DTN node contains several buffers for different purposes. For example, when bundles arrive at a DTN node, they are put in a specific buffer waiting to be routed. After finding their optimal route, the bundles are forwarded to another buffer (still inside the DTN node) to wait until the next hop in the transmission path becomes available.

3. All traffic has the same priority level, and each traffic flow is modeled as a Markov Chain, which is then discretized into a packet stream for simulation purposes.

4. The data rates Rb_{in} of the radios of all neighbor links transmitting packets of information to the intelligent node have the same value. This was done to significantly reduce the action space of the RL agent, which is detailed in subsection 3.3.2.

5. Similarly, the data rates Rb_{out} of the radios of the RL agent transmitting packets of information to the relay nodes or the DSN also have the same value.

6. The RL agent is able to command changes to the radio data rates of the neighbor nodes.

7. The memory state of all neighbor nodes is available to the RL node. This requires implementing a mechanism in the system architecture by which the different nodes can share their memory state with one another. In the near future, we are planning on implementing this functionality into the Interplanetary Overlay Network (ION), JPL's implementation of the DTN protocol stack for spacecraft.

In future work, we are planning on relaxing some of these assumptions. For example, we will add priorities to the bundles to give the RL agent the ability to selectively drop certain packets when it gets congested, and incorporate astrodynamics into the mobility models of the different nodes.

3.3.2 RL agent

3.3.2.1 State Space

The state space of our problem formulation is defined by network parameters that are assumed available to the RL node at all times. It consists of the RL node's memory percentage of utilization u, the most congested neighbor node memory percentage of utilization $\max_{i=1,...,N} u_i$, the data rate Rb_{in} of all links transmitting bundles to the RL node (assumed the same for all links), and the data rate Rb_{out} of the downlinks with the DSNs and the crosslink with the relay nodes (also assumed the same for all output links):

$$\vec{s} = \left[u, \max_{i=1,...,N} u_i, Rb_{in}, Rb_{out} \right] \tag{3.1}$$

This work adds the most congested neighbor node information to the state space representation presented in [39], which only included the memory state of the RL node, and the capacity allocated to the links between RL node and the other nodes in the network. This extra piece of information was found to be crucial for the intelligent agent to be able to dynamically adapt to varying traffic conditions, as it will be seen in the results section. These network parameters successfully characterize the current state of the DTN node and they allow the RL node to take actions accordingly, with the ultimate goal of maximizing the expected cumulative discounted reward. In the state vector shown in eq. 3.1, both the memory utilization percentage from the RL node and most congested neighbor node are real numbers. On the other hand, Rb_{in} and Rb_{out} can take N_{Rb} possible values between Rb_{min} and Rb_{max} (for instance, 10 possible values between 1 Mbps and 1Gbps), emulating the radio's ability to double or halve the commanded transmitting data rate. Since the state vector includes real variables, the total number of states of the system is infinite. Finally, the input to the NN that predicts the expected Q-values for each possible action a at a given state s is the vector $\vec{s}$, as it will be shown in Section 3.4.3.

3.3.2.2 Action Space

The action space in our problem formulation includes 7 different types of actions that the RL node can take depending on the current state of the system:

1. Drop packets. This action is intended to be performed when the RL node gets very congested and there is no other way to decongest the network but to start dropping incoming packets.

2. Increase (double) the data rate (Rb_{in}) of all links from the neighbor nodes to the RL node.

3. Decrease (by half) the data rate (Rb_{in}) of all links from the neighbor nodes to the RL node.

4. Increase (double) the data rate (Rb_{out}) of the downlinks from the RL node to the DSN and crosslinks.

5. Decrease (by half) the data rate (Rb_{out}) of the downlinks from the RL node to the DSN and crosslinks.

6. Route bundles through crosslinks instead of sending them straight to the DSN.

7. Do nothing (i.e., not change any parameter of the network).

3.3.2.3 Reward function

The reward function to solve the RL problem takes into account the amount of information successfully received by the DSN, the resources allocated, and the network memory state. The reward increases with the number of bits that go through the RL node and arrive at the DSN (referenced later as *benefit*) and decreases with the capacity in bits allocated to all

links controlled by the RL node (referenced later as *cost*). The energy in Joules of a link is proportional to the capacity allocated to the link in bits, as shown in the following formula:

$$\mathrm{E}_{link} = \int P(t)dt = \int E_b \cdot R_b(t)dt \propto \int R_b(t)dt = cost_{link}, \tag{3.2}$$

where E_b is the energy per bit, which we assume constant [J/bit].

The reward function was designed in a way that the RL node attempts to maximize the amount of bits sent by the RL node and received by the DSN while minimizing the data rate of all controlled links in every time step window (during which the network traffic conditions are assumed to be stationary). Additionally, if the buffers of the DTN nodes in the network near maximum utilization, then the reward function rapidly switches to a different mode where off-loading of data is heavily prioritized to prevent memory overflows. The reward function is computed in the following way:

$$R(s,a) = f(u) \cdot f\left(\max_{i=1,\ldots,N} u_i\right) \cdot \frac{1}{f_{\,\mathrm{mod}}\,(\eta)} \cdot \frac{\#\text{ bits }_{X \rightarrow \text{ RL node } \rightarrow DSN}}{\mathrm{cost}_{X \rightarrow \text{RL node}} + \mathrm{cost}_{\text{RL node} \rightarrow DSN} + \mathrm{cost}_{\text{RL node} \rightarrow cxl}}, \tag{3.3}$$

where $\#\text{ bits }_{X \rightarrow \text{ RL node } \rightarrow DSN}$ is the number of bits sent from the RL node to the DSN, $\mathrm{cost}_{X \rightarrow \text{RL node}}$ is the cost in bits of all the neighbor node links transmitting bundles to the RL node, $\mathrm{cost}_{\text{RL node} \rightarrow DSN}$ is the cost in bits of the two links between the RL node and the DSNs, and $\mathrm{cost}_{\text{RL node} \rightarrow cxl}$ is the cost in bits of the crosslink between the RL node and the secondary orbital relay in the network. The $\mathrm{cost}_{\text{cxl} \rightarrow DSN}$ is not taken into account since it does not directly involve the lunar RL node. As previously mentioned, the network traffic conditions are assumed to be stationary during a simulation time step.

The factor $f_{\,\mathrm{mod}}\,(\eta)$ intends to capture the difference in energy for different coding schemes. Assuming that the allocated bandwidth in all the links is constant, when using lower data rates, a coding scheme with more redundant bits can be used and, therefore, the

energy per bit –and consequently the cost– reduces. This factor is computed in the following way [*]:

$$f_{mod}(\eta) = \frac{2^{\eta} - 1}{\eta}, \tag{3.4}$$

where η is known as the spectral efficiency and it corresponds to the fraction between the current data rate and the Shannon bandwidth, which is equal to the maximum value of Rb_{out}.

On the other hand, $f(u) = \frac{1}{1+\exp(-a(b-u))}$ is a function that depends on the buffer utilization u and aims to be equal to 1 (unimportant) when buffers are empty (u=0) and penalize (reward tending to 0) as the buffers get full above a certain buffer utilization threshold u_{th} ($u > u_{th}$ with $u_{th} \in [0.6 - 0.8]$). An example of this function is plotted in Figure 3.2.

The reward function from eq. 3.3 has two memory factors using the function from Figure 3.2, $f(u)$ and $f\left(\max_{i=1,\dots,N} u_i\right)$. The former aims to control buffer utilization of the RL node, while the latter aims to prevent back-propagation (congestion of other the other nodes in the network) [39].

3.4 Experimental Setup

3.4.1 Lunar Scenario Case Study

A test scenario consisting of a Lunar mission was used for training and later benchmarking the RL agent. This scenario, illustrated in Figure 3.3, models a Moon-to-Earth network with 46 nodes and two activities: (1) A human outpost on the lunar surface; and (2) a mining operation on the inside of a crater. Both activities generate more than 2,500 traffic flows, most of which originate at the lunar surface and need to be delivered to Earth. The RL agent is located on the primary relay between the Moon and Earth (notionally, the lunar Gateway), which acts as a Gateway for the Direct-to-Earth link. The Gateway communicates with 11

[*]This expression comes from Shannon's limit on power efficiency, which tells us the minimum E_b/N_0 (energy per bit to noise power spectral density ratio) required to achieve a certain spectral efficiency: $\eta = \frac{R_b}{B} < \frac{C}{B} = \log_2\left(1 + \frac{E_b R}{N_0 B}\right) \rightarrow \frac{E_b}{N_0} > \frac{2^{\eta}-1}{\eta}$.

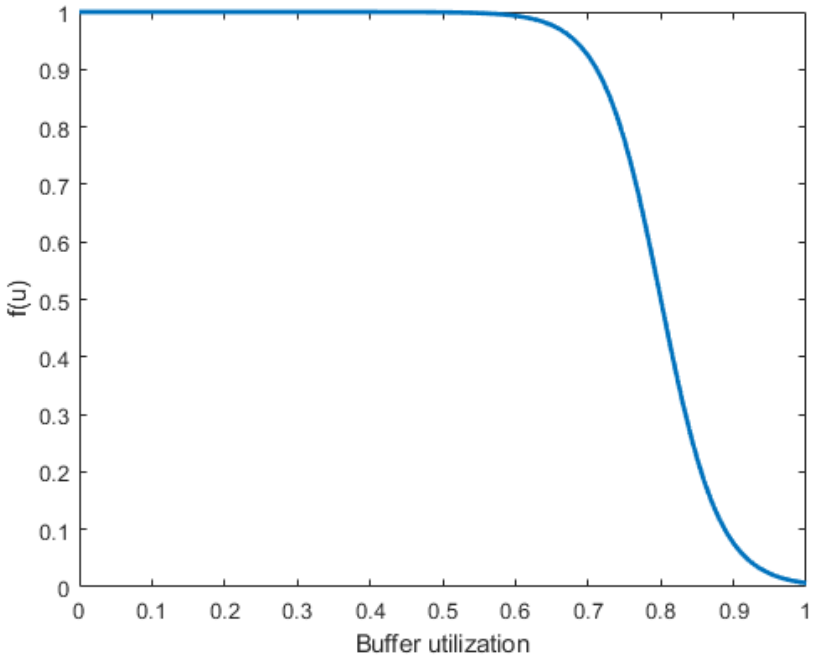


Figure 3.2: Memory factor function to account for node buffer utilization in the reward function computation for $a = 25$ and $b = 0.8$.

neighbor nodes, including 7 relays and 4 user nodes located on the surface of the Moon, as well as with two DSN nodes and a secondary relay orbiting the moon.

3.4.2 DtnSim - Simulation Environment

A medium fidelity Python-based simulator for DTN developed at JPL and called DtnSim [113] was used to simulate the communications network of the cis-lunar mission described in the previous subsection. This simulation environment allows us to simulate the transmission of traffic flows between the different nodes of the system as well as to accurately represent the memory state of all entities forming the system. As mentioned in Section 3.3, our system model assumes a single buffer for each node in the system. Consequently, the percentages of memory utilization of the different nodes of the system were computed in the following way:

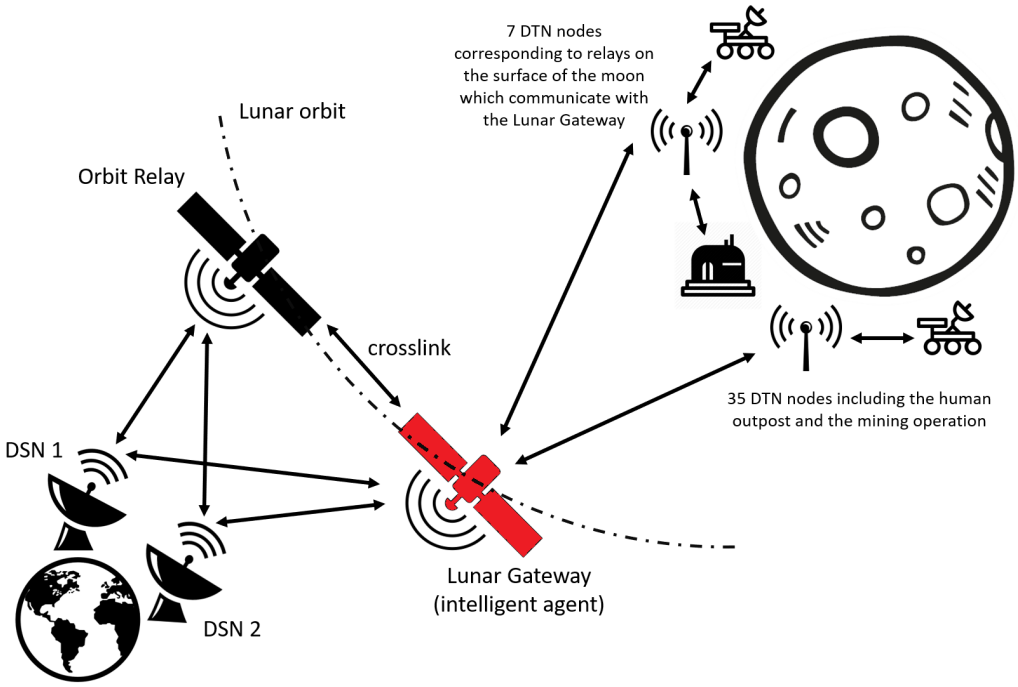


Figure 3.3: Illustration of the Delay Tolerant Network of the Lunar mission used as training/test scenario.

- For the RL node (Gateway), the percentage of memory utilization u was computed by diving the number of bits inside all queues of the RL node divided by the maximum buffer size U_{RL}, which was set to 80 Gbits.

- For the neighbor nodes, the percentage of memory utilization u_i was computed by diving the number of bits inside all queues of node i divided by the virtual maximum buffer size U, which was set to 8 Gbits for all neighbor nodes.

3.4.3 Training and Evaluation of the RL agent

The DTN RL agent was trained using DQL by means of the Python-based library Stable-Baselines [132], which contains a set of improved implementations of RL algorithms based on OpenAI Baselines [133]. The neural network structure is illustrated in Figure 3.4 and consists of an input layer of size 4 (the state vector length) and two fully connected linear

hidden layers with 64 nodes each. The input values were normalized between -1 and 1 and layer normalization was used, as recommended in [134]. The output layer is equal to the size of the action space and each output corresponds to the Q-value of taking each of the possible actions. The discount factor was chosen to be $\gamma = 0.99$, since we are not only interested in high immediate rewards but also high future rewards. Furthermore, $\gamma = 0.99$ is a commonly used value in the literature [96]. The exploration rate ϵ was chosen to follow a linear schedule within the first 200 episodes starting from $\epsilon = 1$ and ending at $\epsilon = 0.02$, to have high exploration at the beginning of the training and more exploitation as the agent learns the dynamics of the system. The training was performed for 1000 episodes, each consisting of 30 minutes of simulation with an action step of 30 seconds. The episode duration was chosen to be 30 minutes to be short enough to allow for a "fast" training process (each episode of 30 minutes of simulation time took about 10-15 minutes of real time to complete), and also long enough to include two different traffic flows: (1) a nominal constant traffic of $\sim$600 Mbps until minute 15, and (2) a period of low constant traffic of $\sim$40 Mbps until minute 30. The training was performed for 4 learning rates and, for each of them, three policies were saved during the training process: the one that provides maximum reward over any single episode, the one that provides maximum average reward over the last 30 episodes, and the last policy obtained at the end of the 1000^{th} episode. A key factor for a successful training of the RL agent is that, in each episode, the initial data rates Rb_{in} and Rb_{out} were randomized to favor visiting the largest possible number of states – or maximum number of possible combinations of network parameters– during training. The action time step of the RL agent was chosen to be 30 seconds, thus allowing the agent to take 60 actions per episode. This value was deemed by a DTN expert to be appropriate, since it seems reasonable to not change the parameters of the DTN network more frequently than every 30 seconds. All the simulation parameters and training hyperparameters are listed in Table 3.1.

After training, the best policies found were each evaluated for 100 extra episodes to

DtnSim Parameters	
Episode duration (training)	30 min (60 steps)
Episode duration (evaluation)	45 min (90 steps)
Maximum buffer size of RL node (U_{RL})	80 Gbits
Virtual maximum buffer size of neighbor nodes (U)	8 Gbits
Stable-Baselines Parameters	
Action time step	30 s
Discount factor (γ)	0.99
Exploration rate (ϵ) start/end	1 / 0.02
Number of training episodes	1,000 (60,000 steps)
Learning rate (α)	[0.01, 0.001, 0.0001, 0.00001]
Target network update frequency (C)	5 episodes (300 steps)

Table 3.1: DtnSim simulation parameters and Stable-Baselines DQL training hyperparameters

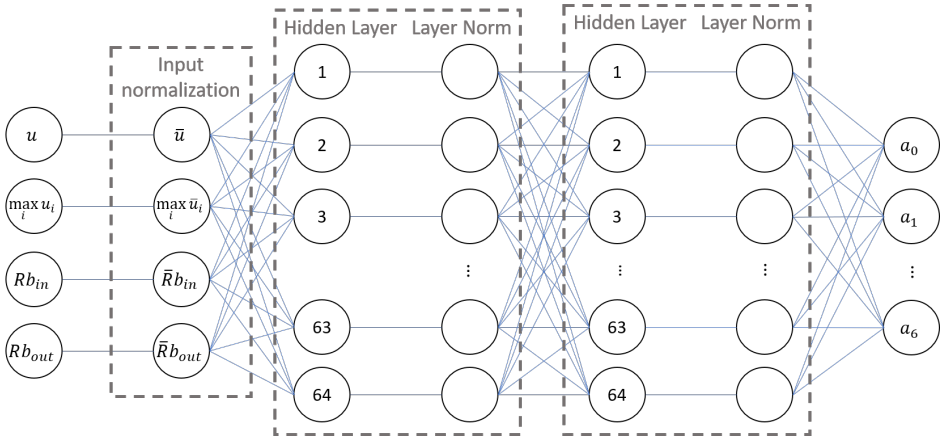


Figure 3.4: Deep Q-Network structure, with an input layer of size 4 (the state vector length), 2 hidden layers with 64 nodes each and layer normalization and an output layer of size 7 (the action space size)

measure their performance. More than 1 episode is needed to evaluate the performance of a policy since the initial data rates in the network are randomly set. Each of the evaluation episodes had a simulation time of 45 mins, and contained a traffic decrease and increase at 15 and 30 minutes respectively, as shown in Figure 3.5. The evaluation episodes were chosen

to be 15 mins longer than the training episodes to be able to assess the performance of the RL agent in a situation (traffic increase) that was not part of the training (which contained a traffic decrease only). Moreover, there was no need to train the agent with the full 45 minutes episode since, by randomizing the initial data rates, some training episodes would start with low data rates, thus congesting the nodes with the initial nominal traffic flow. These circumstances would be equivalent to a rise in traffic, since the RL node would need to increase the data rates of the different links in the network to avoid its congestion and/or prevent the back-propagation phenomenon from happening. This apparently simple traffic pattern, combined with the random initial data rates for all links, can be considered complete and sufficient since it implicitly includes all three main situations (or modes) that the RL agent could encounter:

- RL node **near but under** capacity (bundle arrival rate $\sim$ bundle outgoing rate). This mode is inherently included in the traffic pattern in Figure 3.5 when the system is in steady state (both for low and nominal constant traffic) with adequate values of data rate to keep all nodes in the network at a reasonable memory utilization (around 80%).

- RL node **over** capacity (bundle arrival rate $>>$ bundle outgoing rate). This mode is implicitly included in the transient period right after the traffic increase. As it will be seen in the results section, when the traffic switches from low to nominal value, the data rates are set to low values. During this transient period, the RL node will get congested and, therefore, it will resemble a system over capacity until the system stabilizes with the new higher data rates.

- RL node **largely under** capacity (bundle arrival rate $<<$ bundle outgoing rate). This mode is implicitly included in the transient period right after the traffic decrease. Again, it will be observed in the results section that, before the traffic decreases, the data rates are set to high values when the traffic suddenly slows down. During this tran-

sient period, more resources (data rate) than needed are going to be allocated to the links of the network. Consequently, the RL node will rapidly decongest, resembling a system under capacity until the system stabilizes with the new lower data rates.

This justifies not training the RL agent in other scenarios with different data rates and/or traffic volume conditions, since all the possible states of the system in terms of overall traffic and data rates are implicitly included in the used scenario. Similarly to what was done during training, in each evaluation episode, the initial data rates Rb_{in} and Rb_{out} were randomized to confirm the robustness of the RL agent to deal with different initial network states.

Finally, the performance of the obtained RL policies will be compared with the performance of 3 non-RL benchmark policies:

1. A **reference** policy which since the beginning of the simulation has all links (both Rb_{in} and Rb_{out}) set to the maximum data rate and changes nothing throughout the whole simulation. Consequently, this policy provides the highest benefit but also the highest cost, and has very little variance.

2. A **random** policy which takes random actions.

3. An **expert** policy based on rules (or if-then statements) that determine what actions to take depending on the network state s. To come up with this policy, a DTN expert, Dr. Sanchez-Net determined what actions to take in different network state conditions, effectively providing a policy in the form of a look-up table. Since the memory utilization state variables are continuous, they were discretized to three possible values: LOW ($u < 50\%$), OKAY ($50\% < u < 80\%$) and HIGH ($u > 80\%$). Similarly, instead of using the 10 possible values of data rates, only two cases (and their negated values) are used to define these rules: MINIMUM and MAXIMUM. The full list of rules are provided in Appendix A, and a few compacted rules are listed below:

(a) **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *drop packets*.

(b) **IF** (memory neighbor is OKAY and memory RL node is OKAY) **THEN** *do nothing*.

(c) **IF** (memory neighbor is HIGH and memory RL node is LOW or OKAY and Rb_{in} is not MAXIMUM) **THEN** *increase* Rb_{in}.

(d) **IF** (memory neighbor is LOW or OKAY and memory RL node is HIGH and Rb_{out} is not MAXIMUM) **THEN** *increase* Rb_{out}.

(e) **IF** (memory neighbor is LOW or OKAY and memory RL node is LOW and Rb_{out} is not MINIMUM) **THEN** *decrease* Rb_{out}.

3.5 Results

Figure 3.6 shows the training results of our DTN intelligent agent by plotting the cumulative reward at the end of each of the episodes. While the goal of RL is to learn the optimal policy that maximizes the final cumulative expected reward, there is a particular case where finding the optimal policy cannot be ensured (see the convergence proof of temporal difference learning and the Bellman's equation [93]). This corresponds to the case where function approximation, bootstrapping and off-policy learning are combined (commonly known as the Deadly triad [93]). This is the case in our scenario, where: (1) we are using a NN to estimate the Q-value for each state-action pair (function approximation); (2) DQL uses the temporal-difference method, which uses an estimated value in the update step to estimate the same value (bootstrapping); and (3) DQN learns the value of the optimal policy independently of the agent's actions or how the experience is generated (off-policy learning). Therefore, it is not possible to prove the convergence of our scenario. However, in Figure 3.6, it can be observed that the RL agent is able to learn policies which provide a much

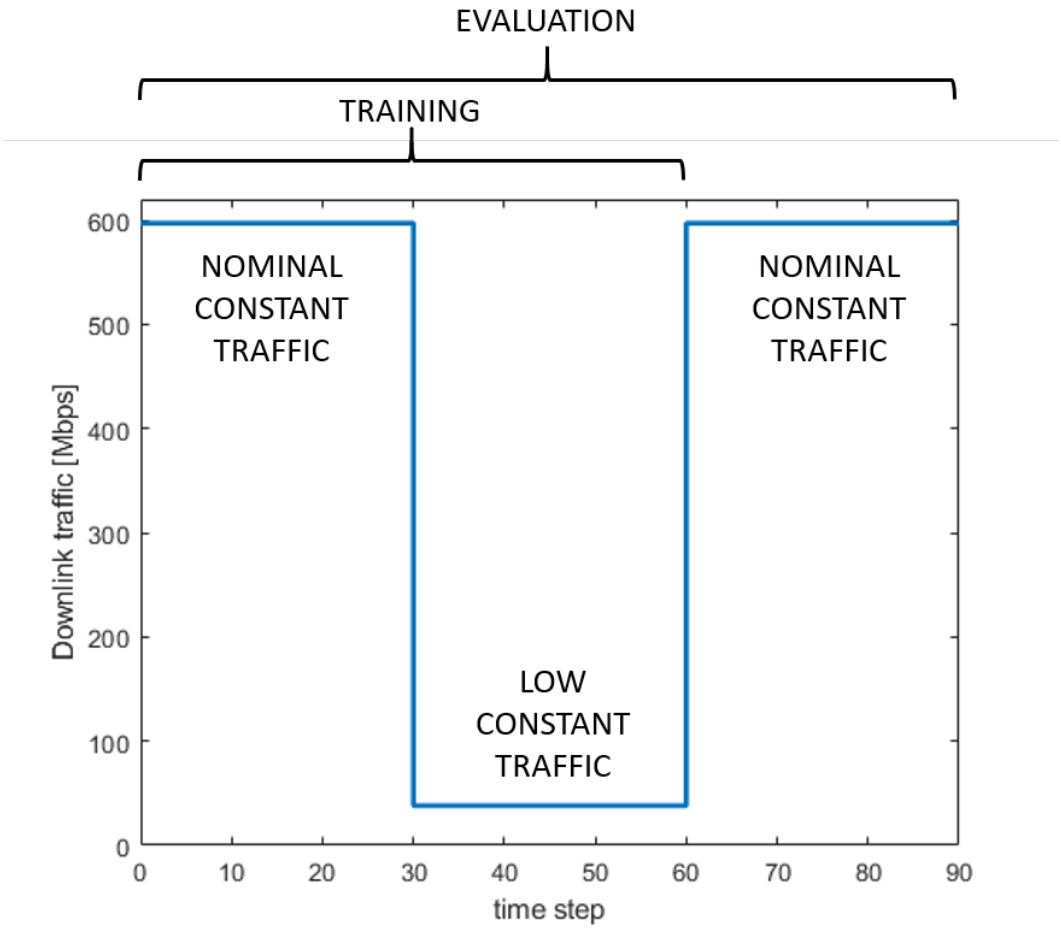


Figure 3.5: Average downlink traffic volume in Mbps for all training (60 time steps) and evaluation episodes (90 time steps). It starts with a nominal traffic of $\sim$600 Mbps until time step 30. Then, there is a period of low traffic of $\sim$40 Mbps until time step 60, where traffic increases again and remains at a nominal value of $\sim$600 Mbps until the end of the simulation.

higher reward value than the ones followed during the exploration phase at the beginning of the training process, where essentially random actions are taken. Within just 200 training episodes, the RL algorithm is able to find policies with high final cumulative reward, which stabilize within a range of final cumulative reward values between 10 and 15. We can conclude that the RL algorithm "converges" to a good policy, since high reward values are achieved by the end of the training process for the learning rate parameters of 0.01 and 0.001

and the final remaining variance comes from to the randomization of the initial data rates at the beginning of every episode.

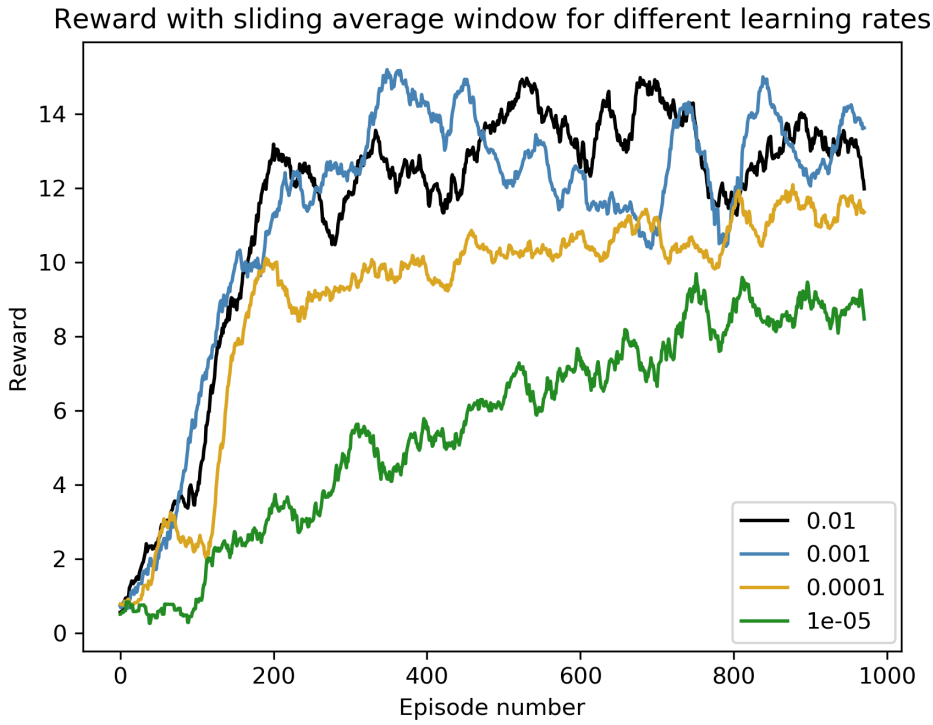


Figure 3.6: Final cumulative reward for different learning rates as a function of number of episodes using a moving average window of size 30.

Figure 3.7 compares the performance of the RL policy that provides the highest average reward for the 100 evaluation episodes (last policy for the learning rate of 1e-3) with the three benchmark policies.

The different boxplots shown in Figure 3.7 are the cumulative episode reward, the benefit (or the number of bits that go through the RL node and arrive to the DSN), and the cost (or the sum of the capacity allocated in bits to all links controlled by the lunar RL node). The major takeaways of Figure 3.7 are the following:

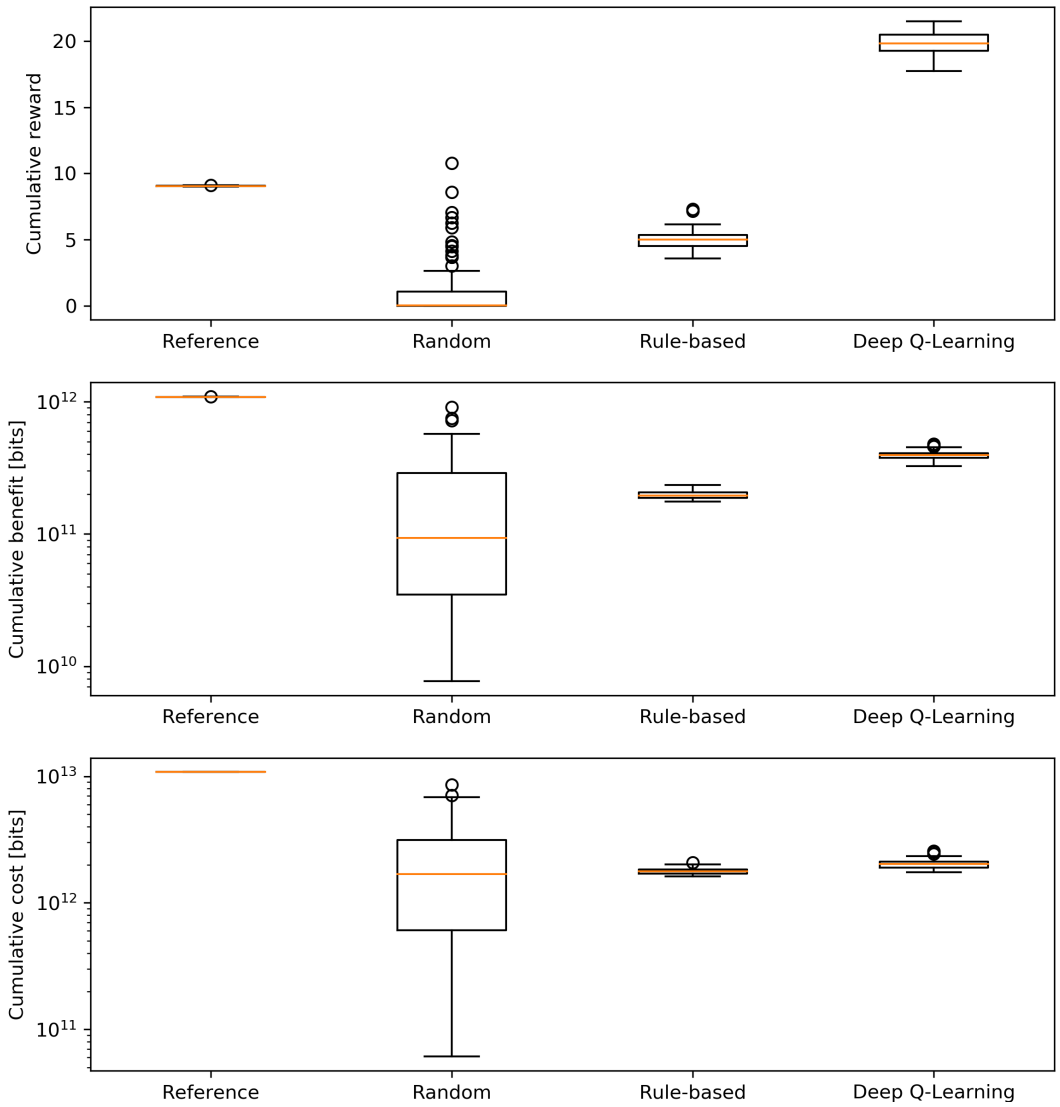


Figure 3.7: Comparison between RL and non-RL policies across three dimensions for 100 evaluation episodes: the episode cumulative reward (top); the amount of information in bits that goes through the RL node and arrive at the DSN, or *benefit* (middle); and the total capacity allocated in bits to all links controlled by the RL node, or *cost* (bottom).

1. The best DQL policy provides the highest reward, outperforming all non-RL policies, finding an adequate balance between benefit and cost.

2. The best DQL policy provides higher benefit than the expert policy at a similar cost.

3. The benefit for the DQL policy is obviously lower than the one obtained for the reference policy but at a significantly lower cost, since the reference policy allocates the maximum capacity to all links at all times.

4. The variance of the DQL policy is similar to the variance of the expert policy, which is a consequence of the randomization of the initial data rates. Contrarily, the variance of the DQL policy is much smaller than the variance of the random policy since taking random actions can lead to end in a wide range of network states, which can be good but also really bad. Finally, the variance of the reference policy is very small since instead of randomizing the initial data rates, they are set to the maximum value from the beginning until the end of the simulation.

Figure 3.8 plots the memory utilization of the of the RL node, together with the amount of bits stored in the neighbor nodes for a representative evaluation episode. In these three subplots, the memory of the RL node, the data rate Rb_{in} of all links transmitting bundles to the RL node, and the data rate Rb_{out} of the downlinks with the DSNs and the crosslink are plotted in solid lines. The number of bits stored in memory for all RL node's neighbor links is represented by dashed lines. It is observed that, at the beginning of this particular episode, the data rates Rb_{in} and Rb_{out} are set to very low values (the initial data rates are randomly selected in the 100 evaluation episodes). Thus, most of the neighbor nodes get congested in just a few time steps. The RL node notices it, and starts increasing Rb_{in} to decongest the neighbor nodes, avoiding the back-propagation phenomenon. After successfully decongesting the neighbor nodes, the RL agent increases the Rb_{out}, since at that point the RL node is also at maximum buffer utilization. Then, the RL algorithm finds a good combination of Rb_{out} and Rb_{in} to keep all nodes in the network at a reasonable memory utilization (around 80%) and, therefore, optimize resource allocation. Then, a decrease in traffic happens at time step 30 and the RL agent reacts to it by decreasing both Rb_{out} and Rb_{in} to optimize

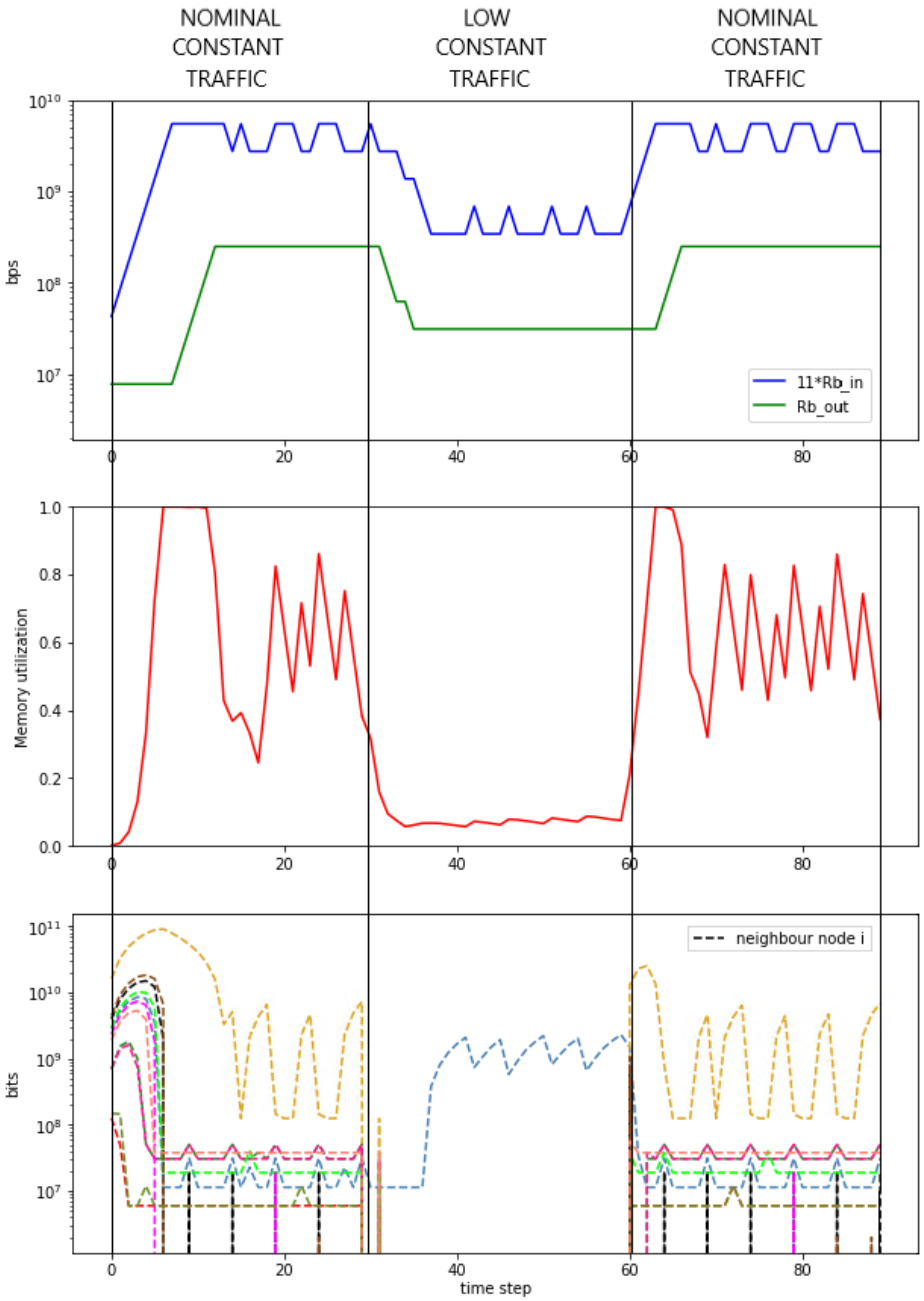


Figure 3.8: Progression over time for a representative evaluation episode of the data rate Rb_{in} of all links transmitting bundles to the RL node [bps] and the data rate Rb_{out} of the downlinks with the DSNs and the crosslink [bps] *(top)*, the memory utilization over time of the RL node node *(middle)*, and the amount of bits stored in memory for all RL node's neighbor nodes [bits] *(bottom)*.

cost, maintain the RL node at very low memory utilization and keep the most congested neighbor node (the blue dashed line) at a reasonable buffer utilization. At time step 60, the traffic increases again and the RL agent reacts to it by increasing both Rb_{out} and Rb_{in} and minimizing memory overflows. It can be observed that the RL agent is able to completely avoid memory overflows in steady-state traffic conditions. However, during transient periods where the data rates are set to a low value and the traffic increases, the RL agent is not able to avoid two small intervals of memory overflow of the RL node (right before time step 10 and right after time step 60). These small periods of memory overflow of the RL node are a consequence of the RL agent not being able to increase fast enough the link data rates, which could be solved choosing a smaller action time step to allow the system to react faster to varying traffic conditions. However, decreasing the action time step could lead to overreacting to small fluctuations in the network state and potential system instabilities. That being said, despite these short and expected intervals of memory overflow, the behavior in this representative episode illustrates how the RL agent is not only able to autonomously manage the DTN in steady-state traffic conditions [39], but also dynamically adapt to varying traffic flows, including both increases and decreases of traffic volume.

3.6 Adding priorities to bundles

After RL was shown to be able to successfully manage a DTN, the next step involves including the option into the intelligent nodes of selectively drop bundles when buffers get saturated in memory based on the priorities of the bundles. During the course of a space exploration mission, the different space assets including relays, rovers or human outposts send to ground and to each other different types of data including Biomedical, Caution/Warning, Command/Teleoperation, Health/Status, Navigation Products, Networking, Video, Science or Voice information among others. Unarguably, depending on the information contained in two different bundles, there might exist a preference whether to drop one or the other. For

instance, in the context of a human spaceflight mission to the Moon or Mars, Biomedical and Health Status bundles have a much higher importance/priority than bundles containing a Science HD video and, therefore, the latter should be dropped first.

In order to accomplish that, two different strategies were considered and compared. For both of them, the bundles were split into three priority categories (HIGH, MEDIUM and LOW). The derails of these two approaches are described below:

1. **Full RL approach**. The first action from the space described in Section 3.3.2.2 was divided in 3 different actions which involved: (1) dropping bundles of LOW priority, (2) dropping bundles of low and medium priorities and (3) dropping ALL bundles (LOW, MEDIUM and HIGH). Furthermore the reward function from Section 3.3.2.3 was used:

$$R(s,a) = f(u) \cdot f\left(\max_{i=1,\ldots,N} u_i\right) \cdot \frac{1}{f_{\,\mathrm{mod}}\,(\eta)} \cdot \frac{benefit}{cost} \tag{3.5}$$

but the calculation of *benefit* was slightly modified and computed the following way:

$$\begin{aligned} benefit = \ & \text{bits}_{\ X \rightarrow \text{ RL node } \rightarrow DSN} - k_1 \cdot \#\ \text{bits dropped}_{low} \\ & -k_2 \cdot \#\ \text{bits dropped}_{med} - k_3 \cdot \#\ \text{bits dropped}_{high} \end{aligned} \tag{3.6}$$

As it can be seen in Eq. 3.5, in order to penalize bundle dropping, the dropped bits were subtracted, using different weights k_1, k_2 and k_3 depending on their respective bundle priority level, to the bits successfully delivered to the DSNs. In other words, the new benefit measure not only seeks to maximize the amount of information that reaches the final destination but also penalizes bundle dropping proportional to a three-level priority scale.

2. **Hybrid approach**. In this second strategy, the first action from the space described in Section 3.3.2.2 was eliminated. In other words, the responsibility of dropping packets

was removed from the intelligent agent. Instead 3 very simple rules were incorporated inside the DTN node to take care of selective bundle dropping depending on the level of buffer congestion:

(a) **IF** (0.85 < memory RL node < 0.9) **THEN** *drop packets LOW priority*.

(b) **IF** (0.9 <= memory RL node < 0.95) **THEN** *drop packets LOW and MEDIUM priority*.

(c) **IF** (memory RL node >= 0.95) **THEN** *drop ALL packets (LOW, MEDIUM and HIGH priorities)*.

Similarly to the first approach, the reward function shown in Eq. 3.5 was used but the calculation of *benefit* was computed the following way:

$$benefit = \text{ bits }_{X \rightarrow \text{ RL node } \rightarrow DSN} - \# \text{ bits dropped} \tag{3.7}$$

Again, in order to penalize bundle dropping, the dropped bits were subtracted to the bits successfully delivered to the DSNs. This time, however, there was no need to assign different weights to the bits dropped based on priority since dropping packets is no longer responsibility of the intelligent agent.

Figure 3.9 compares the performance of the Full RL and Hybrid approaches over 100 evaluation episodes across 5 different metrics: benefit, cost and number of bits dropped for each of the three different levels of priority considered. It can be observed that the hybrid approach provides higher benefit and lower amount of information dropped than the Full RL approach. In fact, the Hybrid approach only drops packets of low priority and considerably reduces important information loss. This happens to the expense of allocating more resources to the different channels (higher cost) but, according to experts, allocating more bandwidth to the network links is highly preferable to losing high priority or sensitive information.

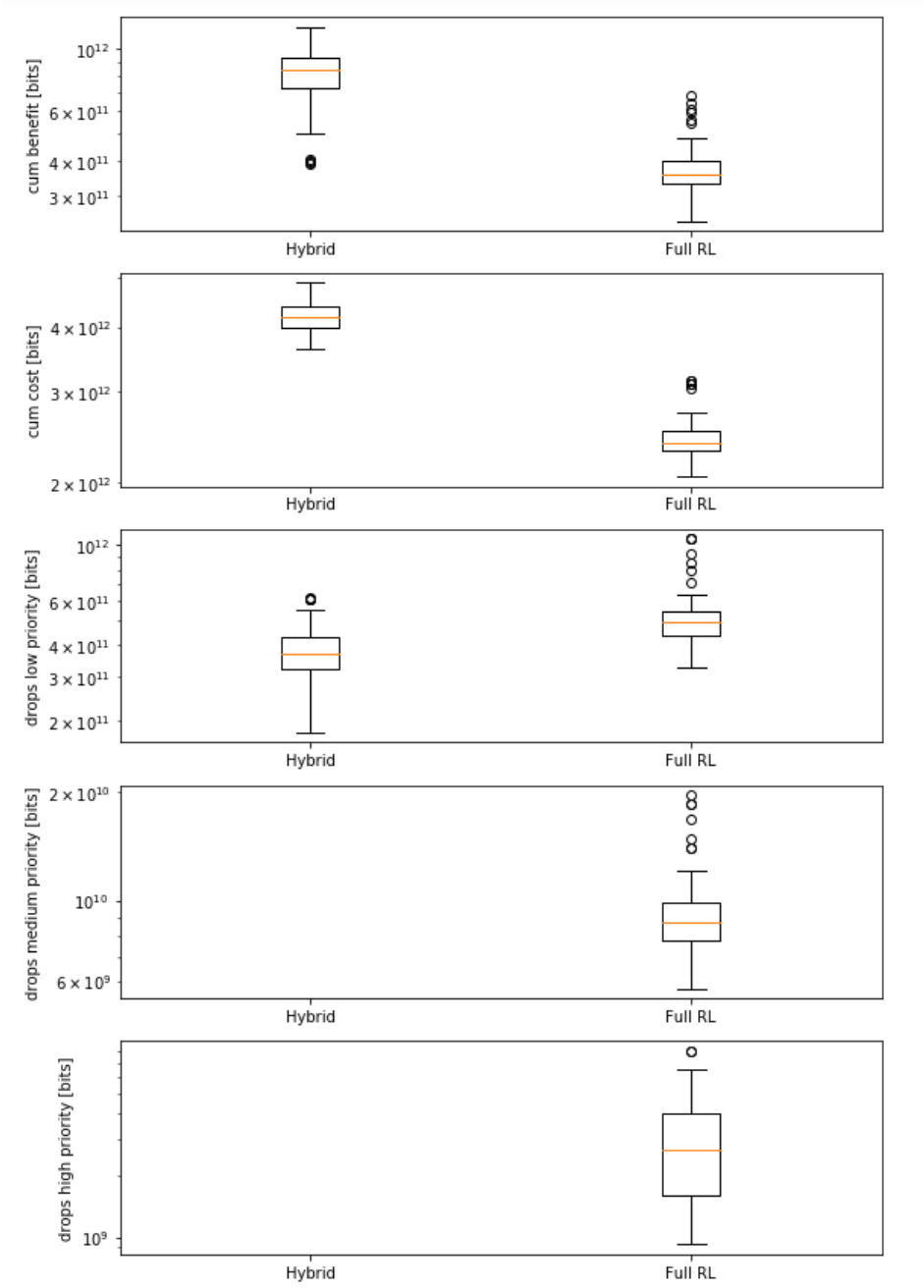


Figure 3.9: Comparison of the performance between Full RL and Hybrid approaches for 100 evaluation episodes across: *benefit*, *cost*, number of low priority bits dropped, number of medium priority bits dropped and number of high priority bits dropped (from top to bottom)

For the hybrid approach, the progression over time of the data rate Rb_{in} of all links transmitting bundles to the RL node, the data rate Rb_{out} of the downlinks with the DSNs and the crosslink, the memory utilization of the RL node and the number of bits dropped for the three levels of priority is shown for a representative episode in Figure 3.10. Similarly to what we observed in Figure 3.8, at the beginning of the episode, the data rates Rb_{in} and Rb_{out} are set to very low values since the initial data rates are randomly selected in the evaluation episodes. Thus, the RL node and neighbor nodes get congested in just a few time steps. The RL node notices it, and starts increasing Rb_{in} to avoid the back-propagation phenomenon. After successfully decongesting the neighbor nodes, the RL agent increases the Rb_{out}, since at that point the RL node is also at high buffer utilization. Then, the RL algorithm finds a good combination of Rb_{out} and Rb_{in} to keep all nodes in the network at a reasonable memory utilization (around 80%) and, therefore, optimize resource allocation. The main difference between the behavior of this new hybrid intelligent agent with respect the behavior offered by the one in Section 3.5 when priorities were not considered is that now the intelligent agent uses both low priority packet dropping and changes in Rb_{out} to keep the RL node at around 80% of memory utilization during periods of nominal traffic. In Figure 3.8, the RL agent was able to do so by dropping packets regardless of their priority and without changes in the bandwidth allocated to the communication channels between the RL node and the DSNs. We can observe that the intelligent agent still reacts to a decrease in traffic at time step 30 by decreasing both Rb_{out} and Rb_{in} to optimize cost. Similarly, at time step 60, the traffic increases again and the RL agent reacts to it by increasing both Rb_{out} and Rb_{in} and minimizing memory overflows. In fact, with this new Hybrid approach, the RL agent is able to completely avoid memory overflows in both steady-state and transient traffic conditions.

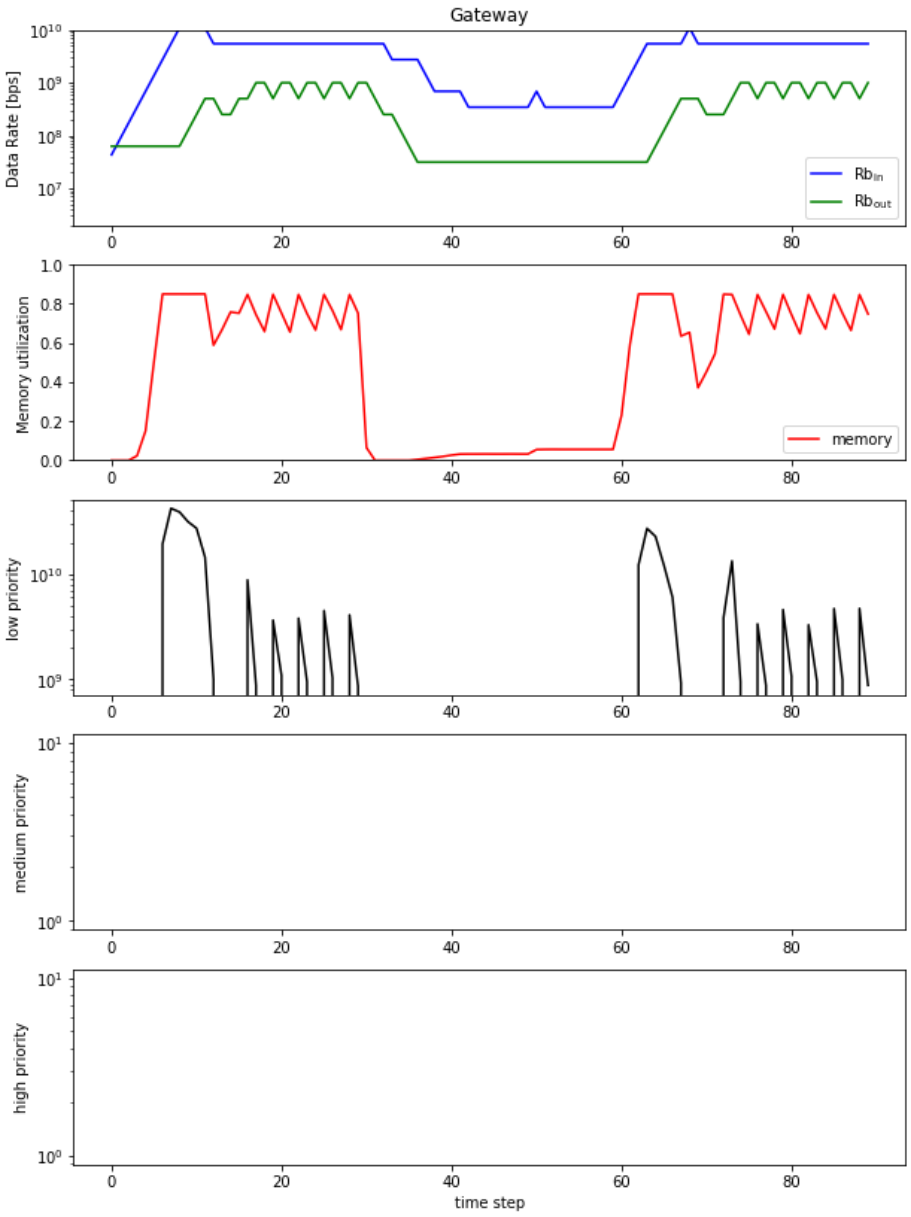


Figure 3.10: Progression over time for a representative evaluation episode of the data rate Rb_{in} of all links transmitting bundles to the RL node [bps] and the data rate Rb_{out} of the downlinks with the DSNs and the crosslink [bps], the memory utilization of the RL DTN node, the amount of low priority information dropped [bits], the amount of medium priority information dropped [bits] and the amount of high priority information dropped [bits] *(from top to bottom)*.

3.7 DQL for Delay Tolerant Network management in a Reactive Imaging Earth Observing Constellation

This section aims to extend the work previously done by exploring the performance of DQL in another scenario with a substantially different network topology and traffic characteristics. This new scenario, illustrated in Figure 3.11 consists of a Walker constellation, formed by 24 satellites distributed across 3 planes, observing floods in 5 global regions over a simulation time of 6 hours. The satellites are placed at a 710km altitude and 98.5 degrees of inclination orbit configuration, which requires 8 satellites in each plane to ensure consistent in-plane line-of-sight and restricts cross-plane visibility to the polar regions. Moreover, in this new scenario, astrodynamics models are incorporated into the mobility of the different network nodes.

3.7.1 Problem Formulation

The methodology followed in this new scenario consists of a centralized approach to train an intelligent agent that will will be placed in ground which not only control one but all DTN nodes in the network. This leads to a much larger action space – 51 actions were considered versus 7 in previous previous – including the following:

- 24 different actions to increase the data rate of radio of node i ($i = 1, ..., 24$)
- 24 different actions to decrease the data rate of radio of node i ($i = 1, ..., 24$)
- An action to increase of the data rate of all node's radios
- An action to decrease of the data rate of all node's radios
- An action to not alter any parameter in the network

The state of the network, which will be available at all times for the intelligent agent, is

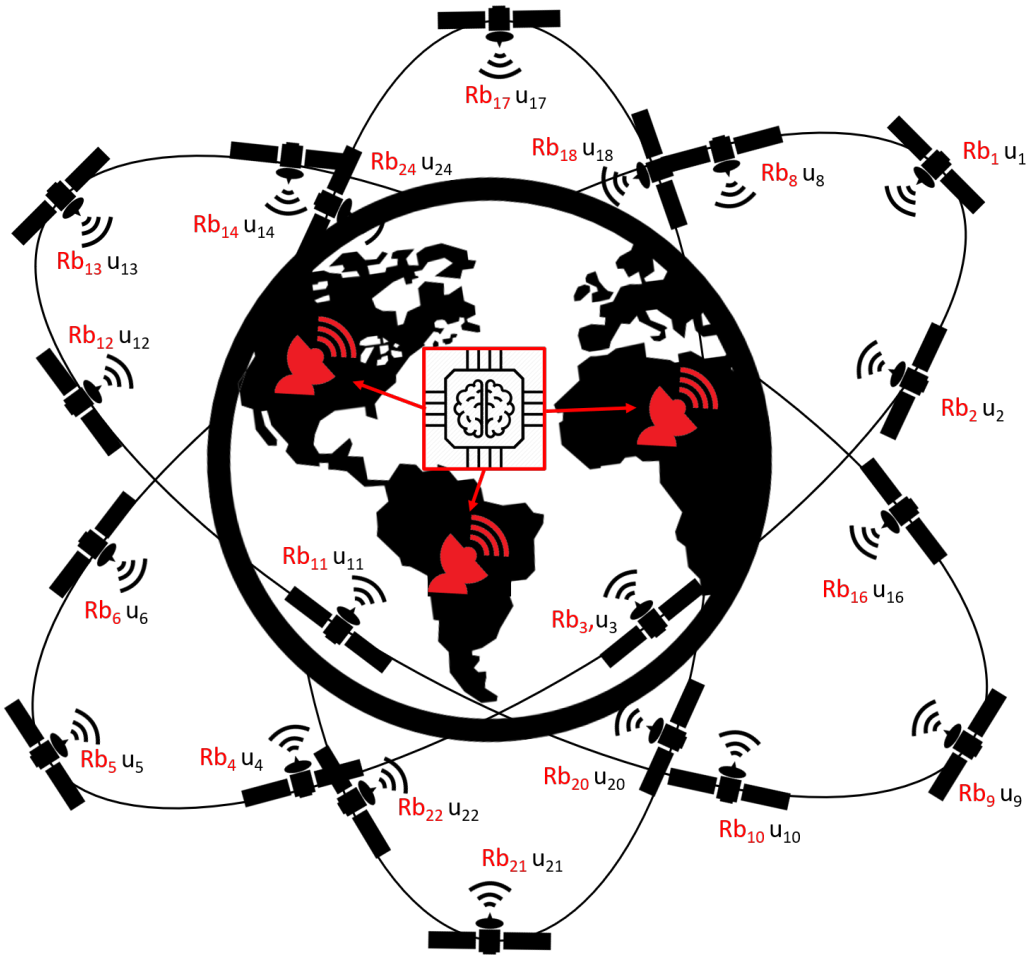


Figure 3.11: Illustration of the Delay Tolerant Network of the Earth Observing mission scenario. The variables controlled by the centralized intelligent agent, which is part of the DSN ground operations, are shown in red font.

characterized by the following vector:

$$\vec{s} = [u_1, ..., u_{24}, Rb_1, ..., Rb_{24}]\,, \tag{3.8}$$

where u_i is the memory utilization of node i and Rb_i is the data rate of i's node radio. Rb_i can take up to 10 values from 1 kbps to 1 Mbps simulating againg the radio's ability to

double or halve the commanded transmitting data rate.

The agent's goal will be to maximize the number of bits that reach their final destination while minimizing the capacity allocated to all the channels in the network and controlling the buffer memory state of all nodes to avoid bundle drops. Therefore, the reward function considered in this new scenario is of the same nature as Eq. 3.3 and presented below:

$$R(s,a) = f\left(\max_{i=1,\ldots,24} u_i\right) \cdot \frac{\#bits_{dest}}{cost}, \tag{3.9}$$

where $\#bits_{dest}$ is the total number of bits that arrive at the destination, $cost$ are the total amount of resources allocated in bits to all network node's radios and $f(u)$ is the memory factor function to account for node buffer utilization plotted in Figure 3.2.

3.7.2 Training and evaluation of the intelligent agent

Following a very parallel process to the one in Section 3.4.3, the training was performed for 1000 episodes, each consisting of 180 time steps. The neural network structure is very similar to the one shown in Figure 3.4 but this time consists of an input layer of size 48 (the state vector length) and two fully connected linear hidden layers with 64 nodes each. The input values were normalized between -1 and 1 and layer normalization was included after each hidden layer. The output layer is equal to the size of the action space and each output corresponds to the Q-value of taking each of the possible actions. This time, however, different time steps were considered to assess the sensitivity of this parameter in the intelligent agent's performance. In each episode, the initial data rates Rb_i were randomized to favor visiting the largest possible number of states during training. All the simulation parameters and training hyperparameters are listed in Table 3.2.

In order to benchmark the effectiveness of using DQL in this Earth observation mission scenario, the performance of the trained intelligent agent was assessed in 100 extra evaluation episodes and compared to a non-RL expert rule-based policy governed by the following set

DtnSim Parameters	
Episode duration (training and evaluation)	180 steps
Maximum buffer size of network nodes	80 Kbits
Stable-Baselines Parameters	
Action time step	5, 10, 15, 30, 45, 60 s
Discount factor (γ)	0.99
Exploration rate (ϵ) start/end	1 / 0.02
Number of training episodes	1,000 (180,000 steps)
Learning rate (α)	[0.01, 0.001, 0.0001]
Target network update frequency (C)	5 episodes (900 steps)

Table 3.2: DtnSim simulation parameters and Stable-Baselines DQL training hyperparameters

of rules:

1. **IF** COUNT($u_{i=1,...,24}$>0.8) > 2:

   THEN *INCREASE the data rate of all node's radios*.

2. **IF** COUNT($u_{i=1,...,24}$>0.8) == 1:

   THEN *INCREASE the data rate of radio of node j for which u_j>0.8*.

3. **IF** COUNT($u_{i=1,...,24}$>0.8) == 0 and COUNT(0.5<$u_{i=1,...,24}$<0.8) == 0:

   THEN *DECREASE the data rate of all node's radios*.

4. **IF** COUNT($u_{i=1,...,24}$>0.8) == 0 and COUNT(0.5<$u_{i=1,...,24}$<0.8) == 24:

   THEN *not alter any parameter in the network*.

5. **ELSE**:

   DECREASE the data rate of radio of node i with minimum memory utilization.

3.7.3 Results

The plot shown in Figure 3.12 aims to assess the effect on the selected action time step in the intelligent agent's performance. In particular, in this figure we are evaluating the agent's

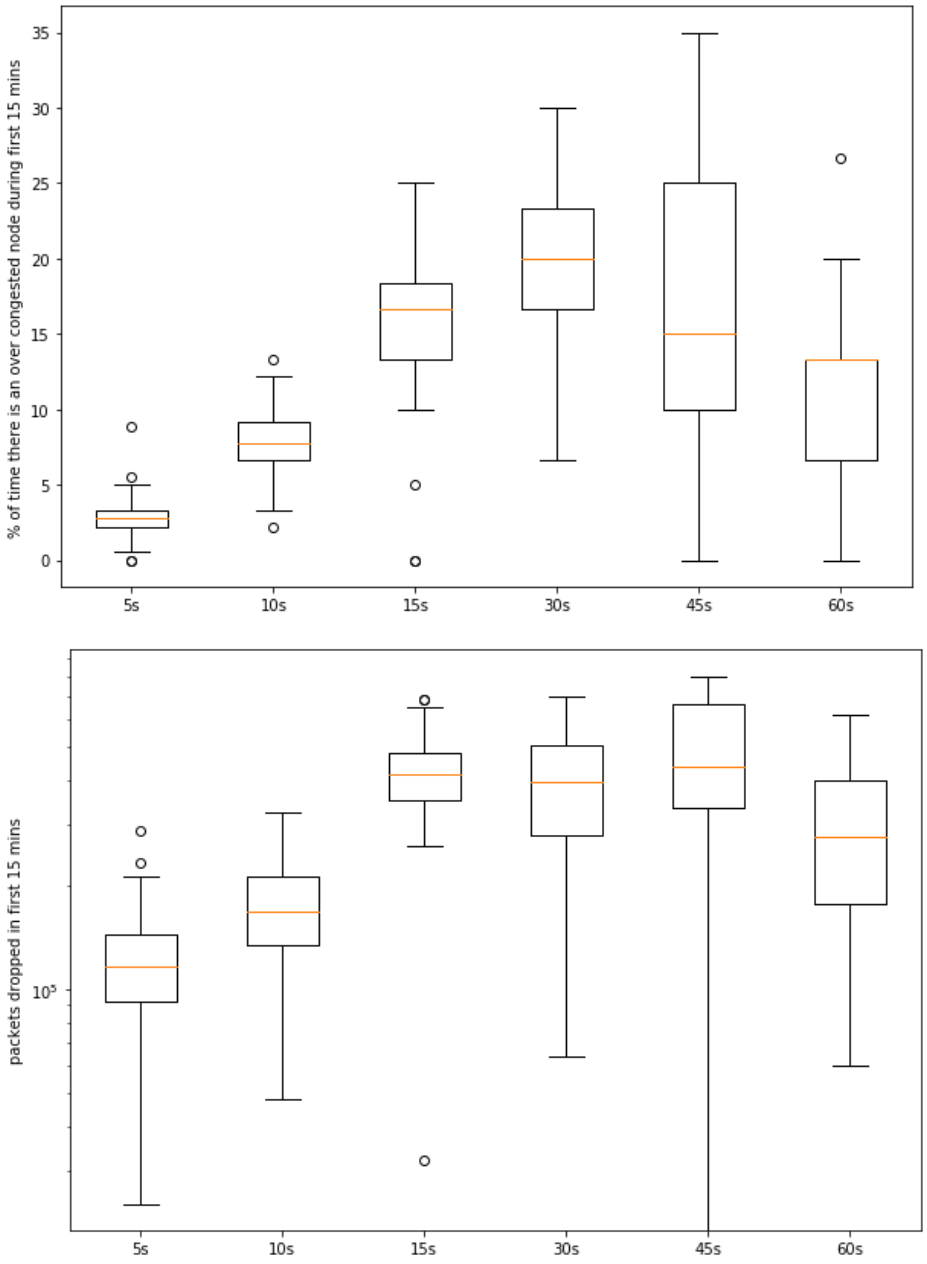


Figure 3.12: Comparison of the agent's performance between Full RL and Hybrid approaches for 100 evaluation episodes. The *top* plot shows the percentage of time there is at least one over congested node and the *bottom* plot shows the total amount of bundles dropped in bits due to buffer memory saturation.

ability to prevent memories overflow by looking at the percentage of time in which there are over congested nodes and the amount of packets dropped due to buffer memory saturation for 7 different time step values: 5, 10, 15, 30, 45 and 60 seconds.

It can be observed the trend that, as the time step grows, the percentage of time nodes get saturated increases together with the amount of information dropped. This is consequence of the agent's inability to react fast enough and increase the radio data rates of the nodes getting saturated in memory if the time step is too large. In the bottom plot, it can be seen a jump in performance when choosing a time step smaller than 15s. Figure 3.13 aims to explain the reason why this happens. Indeed, the top plot of this figure shows the amount of information in bits that reaches every single satellite in the constellation during the 6 hours of simulation with ideal unconstrained links. If we zoom in to the first 10 minutes (bottom plot) we can observe that the bundles (all bundles have a size of 4 kbits) arrive at the different nodes in the network with a frequency of approximately 15 seconds. A good rule of thumb that can be extracted from this observation is to choose a time step length such that the average number of bits received during that time step does not exceed the 5% of the buffer size of the DTN nodes. In this example, since packets of 4 kbits arrive at a 15s rate and considering that the size of the buffers was 80 kbits, an appropiate time step choice would be $t_{step} < 0.05 * 8e4 * 15/4e3 = 15s$. A more general expression would be the following:

$$t_{step} \leq 0.05 \cdot \frac{Size_{buffer}}{L \cdot \lambda} \tag{3.10}$$

where λ corresponds to the bundle arrival rate in s^{-1} and L corresponds to the bundle length in bits.

Figure 3.14 shows the behavior of the trained intelligent agent for a 10s action time step. Particularly, this figure plots the progression over time of the memory utilization and the data rate of all radios of the satellites in the constellation for a representative evaluation episode.

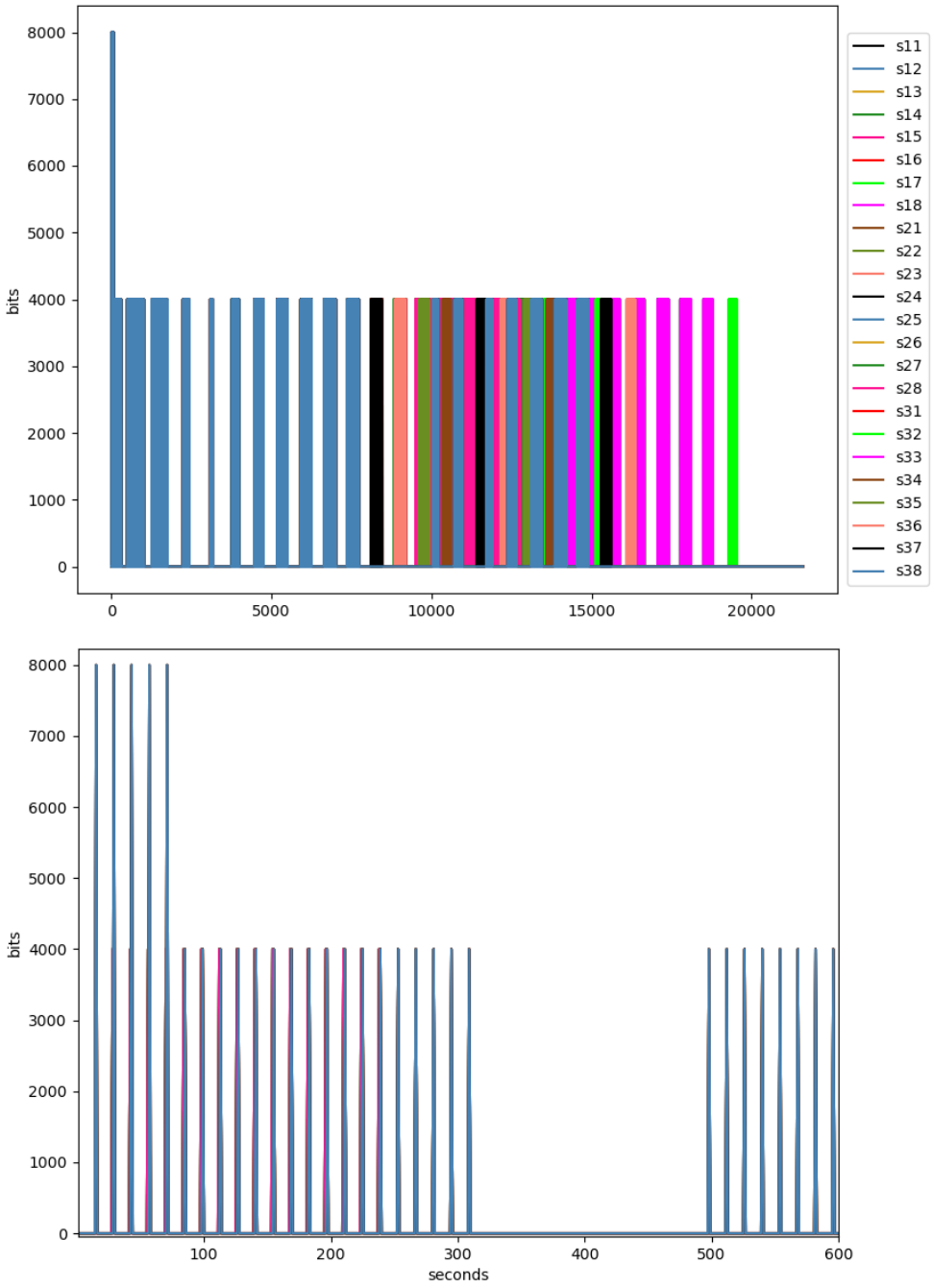


Figure 3.13: Traffic in all nodes forming the Earth Observing constellation with ideal unconstrained links. The *top* plot shows the amount of information in bits that reaches every single satellite in the constellation during the 6 hours of simulation and the *bottom* plot zooms in the first ten minutes.

At the beginning of the episode, there appear to be small periods of memory overflows in some nodes since the intelligent agent prioritizes decreasing the data rates of all radios to optimize the cost or resource allocation. After that first stage, at around time step 25 the intelligent agent starts to smartly distribute the resources among the different nodes in the system, changing the data rates of the different radios with the goal of avoiding memory overflows.

Finally, the intelligent agent's performance was compared to the expert policy described in the previous section. Figure 3.15 presents a boxplot for comparison between the performance the intelligent agent and rule-based expert policies for 100 evaluation episodes across reward, benefit and cost. Similarly to what we observed in the lunar scenario in previous sections, the RL policies outperform the expert policies based on rules/heuristics by obtaining a higher reward through higher benefit or bit successfully delivered in their intended destination while also allocating more efficiently the bandwidth to the different radios of the satellites forming the Earth observing constellation.

3.8 Conclusion

This chapter describes a new method that uses RL, and in particular Deep Q-Learning, to manage a Delay Tolerant Network. The proposed method is successfully applied to the management of two different networks: (1) a cis-lunar network consisting of an orbital relay that serves as the primary Gateway to exchange data between Earth and the Moon and (2) an Earth Observing multi satellite network consisting of Walker constellation, formed by 24 satellites distributed across 3 planes, observing floods in 5 global regions over a simulation time of 6 hours. First, in this work it is shown that RL, and in particular DQL, is able to learn good policies within ~ 200 training episodes using a well-tuned learning rate. Secondly, the policies learned by the RL agent provide higher benefit (amount of information successfully sent to the DSN) than a rule-based policy designed by a DTN expert, and at a similar cost

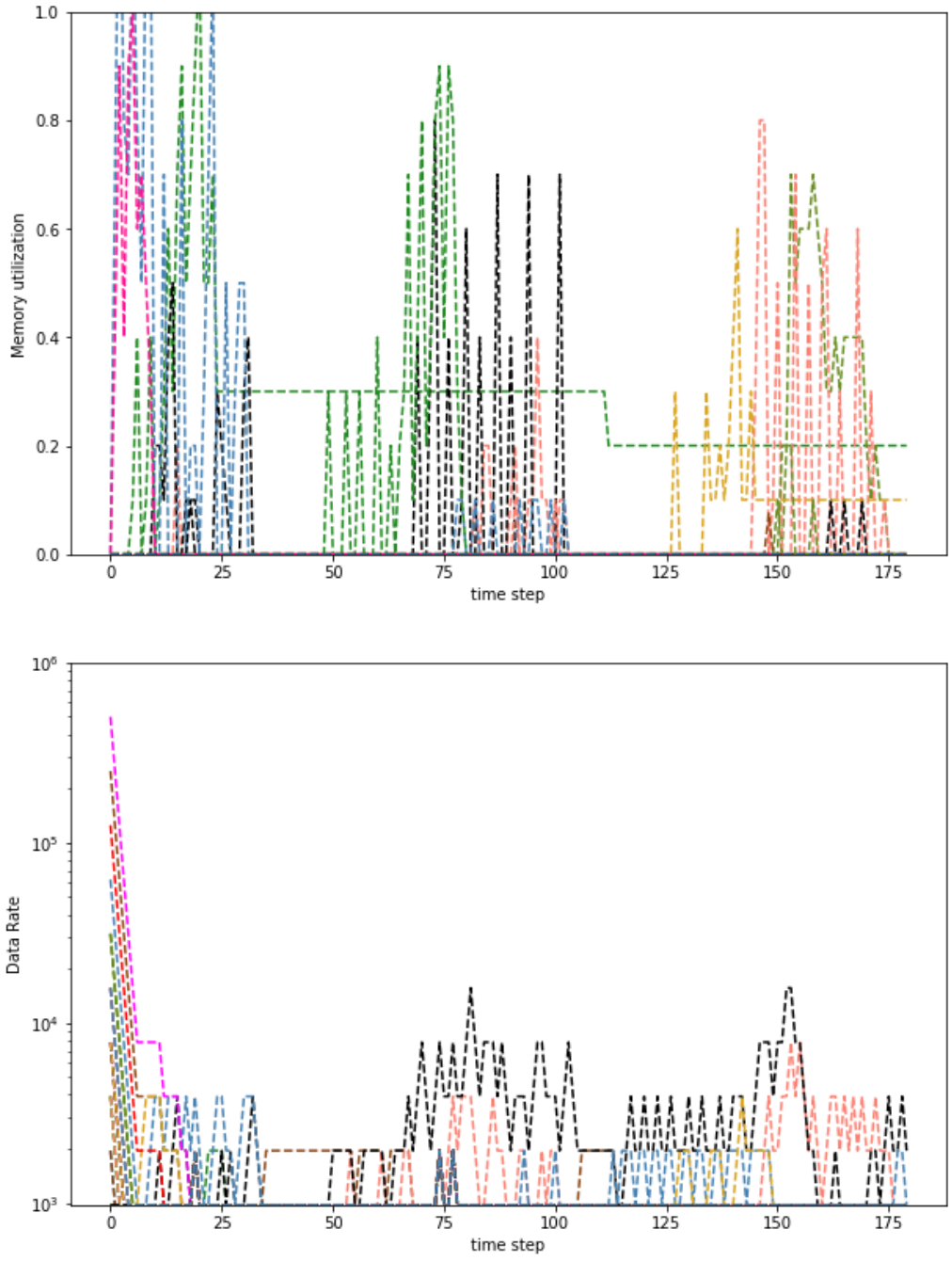


Figure 3.14: Progression over time for a representative evaluation episode of the memory utilization (*top*) and the data rate in bps (*bottom*) of all radios of the satellites in the constellation. Each color represents a different satellite or DTN node.

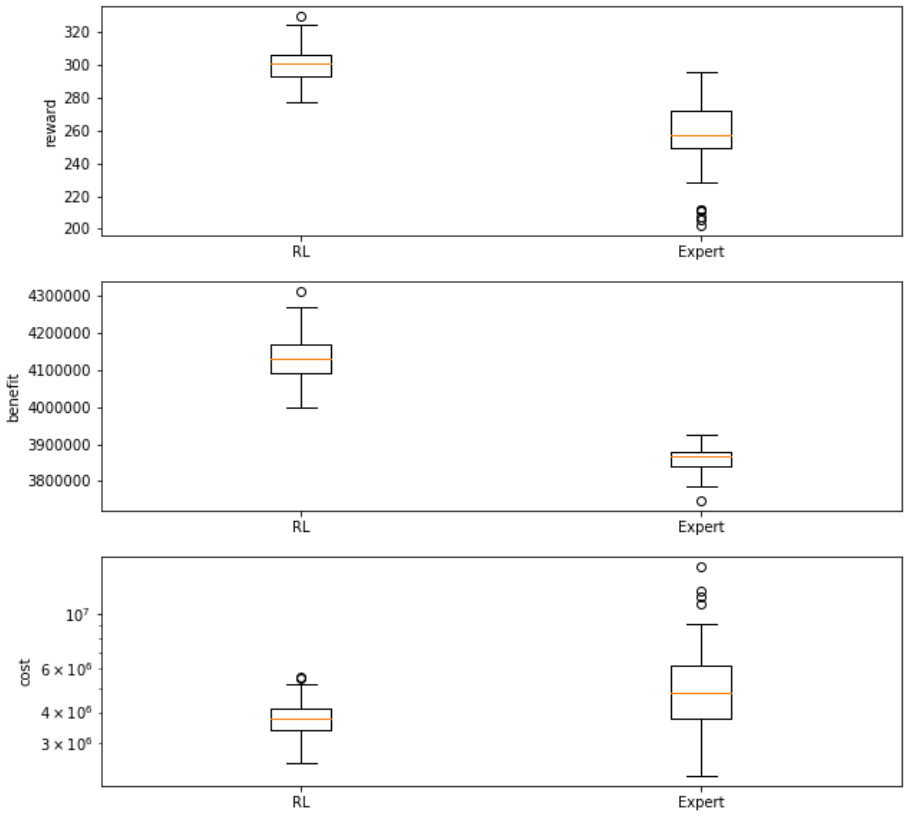


Figure 3.15: Comparison of the performance between the intelligent agent and rule-based expert policies for 100 evaluation episodes across: *reward*, *benefit* and *cost* (from top to bottom)

(capacity in bits allocated to the links controlled by the RL node). Furthermore, the DQL policies provide the highest reward, outperforming all non-RL policies, finding an adequate balance between benefit and cost. Also, the amount of information received by the DSN for the RL policies is obviously lower than the one obtained for the reference policy, which allocates the maximum capacity to all links at all times, but at a significantly lower cost. In conclusion, the RL solution offers better performance than the expert policy and finds an adequate balance between the number of bits successfully received the DSN and the capacity (or data rates) allocated to the RL node links. All this, while also controlling the buffer utilization of all nodes in the network, and therefore minimizing memory overflows and

back-propagation. In particular, the RL agent is able to completely avoid memory overflows in steady-state traffic conditions. However, during transient periods, the RL node experiences small intervals of memory overflow. Future research will relax the assumptions from Section 3.4 and experiment with a wider range of parameters. Controlling the buffer utilization is crucial for DTN networks, since they assume there is enough memory available to store and forward data as bundles arrive to the different nodes in the system. Finally, a key result is that the RL agent is not only able to autonomously manage the DTN in steady-state traffic conditions, but also dynamically adapt to varying traffic flows. Moreover, the intelligent agent is able to selectively drop bundles with different priorities (e.g., depending on the transmitter or the data flow type) when the node is fully congested.

Despite the good results obtained in this first attempt to manage DTN nodes using RL, more research is necessary to improve the RL agent and more thoroughly compare it against state-of-the-art strategies. Since the performance of RL was assessed for two particular scenarios, the next step would be to train the RL agent with a richer and more varied set of scenarios to characterize its robustness to other traffic conditions and network topologies. Also, despite Deep Q-Learning (DQL) working efficiently in this problem, other RL algorithms such as Actor-Critic, Trust-Region Policy Optimization (TRPO) or Proximal policy Optimization (PPO) could be implemented, and their performance compared against DQL. Finally, the ultimate goal is to add this AI layer into the Interplanetary Overlay Network (ION), JPL's implementation of the DTN protocol stack for spacecraft.

4. CONCLUSION

4.1 Summary

This thesis used AI and machine learning techniques to assist in the design and operation of Distributed Spacecraft Missions (DSMs). The first part of this thesis utilizes Global Optimization techniques to provide an efficient way during Pre-Phase A and Phase A design studies to deal with the combinatorial explosion of feasible Earth Observing multi-satellite designs when considering non-symmetrical constellation configurations. The second part of this thesis uses Machine Learning techniques to reduce the constant monitoring and ground operations required for the buffer management of Delay Tolerant Networks (DTNs).

Chapter 1 discussed the transition in the last few decades from monolithic to distributed architectures for space exploration. In this new scenario, the complexity of designing DSMs beyond symmetrical Walker constellations motivated the development of computationally efficient tradespace exploration tools to search through a vast and usually understudied constellation design space. This chapter explored an expert human-centric approach based on design heuristics gained through experience, which helps to quickly identify promising candidate designs but offers limited explorative capabilities to evaluate more than a handful of different design alternatives. This chapter also discussed another approach extensively used in the literature which does not require as many resources in the form of expert designers with years of experience, or a physical space to perform tradespace exploration for the design of new DSMs. This other approach consists of using Multi-Objective Evolutionary Algorithms (MOEAs), which tend to be computationally inefficient since they do not have available prior knowledge about the problem. To overcome this challenge, Chapter 1 improved the computational efficiency of MOEAs by proposing new evolutionary formulations to tackle the exploration of constellations for Earth observation DSMs including asymmetrical hybrid

configurations. This was done under the premise that choosing the right evolutionary formulation helps increasing the computational efficiency, since the problem formulation is at least as critical in driving performance as the details of the optimization algorithm used itself.

This introductory chapter also examined the added complexity in the operational phase of DSMs, especially focusing on space communications. We discussed the need of DTN protocols to enable robust communications in environments in which data transmission experiences long delays and constant connectivity disruptions between network nodes. This chapter motivated the need of developing novel intelligent tools to help automate some aspects of the decision making in the DTN technology due to the current persistent need of continuous ground operations on these networks. In order to accomplish that, Chapter 1 provided the necessary background on reinforcement learning, which was later used in the thesis to design and train a controller to manage DTN node buffers.

Chapter 2 developed novel evolutionary formulations in the form of chromosomes and operators to explore a vast region of the tradespace of non-walker hybrid constellation designs. These new evolutionary formulations were developed during the creation of a constellation design tool led and owned by NASA Goddard Spaceflight Center, which is used during the early design stages of Earth Observing missions. This tool serves the purpose of efficiently exploring large tradespaces of constellation designs, with the ultimate goal of performing feasibility studies, trade studies and what-if analyses. By means of these new formulations and the computational power of MOEAs, promising designs were identified in a wide variety of multi-satellite coverage problems including the observation of symmetrical, asymmetrical, connected and disconnected regions of interest, and large and small instrument Field Of Views (FOV). The proposed formulations were evaluated and compared against other existing formulations in the literature, including the ones that just focus on simple and symmetrical configurations such as the Walker pattern, and other more general formulations that consider all satellite's orbital parameters as independent optimization vari-

ables.

Chapter 3 introduced a new methodology that uses Deep Reinforcement Learning to smartly and autonomously manage DTN nodes. This chapter of the thesis starts by designing a model-free optimal controller able to maximize network performance while anticipating failures and avoiding memory overflows. Then, in order to assess the potential of using RL in DTN management in DSMs, an intelligent agent was trained on a realistic lunar mission scenario and its performance benchmarked against other non-RL based policies, including a rule-based policy designed by a DTN management expert. Finally, Chapter 3 also explored the performance of DQL in another scenario with a substantially different network topology and traffic characteristics. This new scenario consisted of a Walker constellation, formed by 24 satellites distributed across 3 planes, observing floods in 5 global regions over a simulation time of 6 hours. This scenario incorporated realistic astrodynamics models into the mobility of the different network nodes and priorities were added to the different DTN bundles to allow the intelligent agent to selectively drop packets of information when nodes get saturated in memory. In order to benchmark the effectiveness of using DQL in this Earth observation mission scenario, the performance of the trained intelligent agent was also compared to other non-RL decision making strategies, including random and expert ruled-based policies.

4.2 Main Contributions

The main contributions and findings of this thesis are summarized below:

- The proposed hybrid constellation evolutionary formulations (chromosomes and operators) allow to explore efficiently a previously understudied constellation design space, since most works in the literature solely focus on searching through the space of simple and symmetrical configurations such as the Walker pattern.

- The proposed evolutionary formulations, which allow to explore hybrid and heteroge-

neous constellations, were shown to outperform in the form of higher convergence rate and final hypervolume than a state of the art general formulation which treats all satellite variables as independent continuous variables in an application case study with disjoint areas of interest.

- No previous work had provided an evolutionary formulation with similar level of performance (convergence rate and hypervolume) that adapts to the nature of the coverage problem in hand (connected/disconnected regions of interest and wide/small FOV). The proposed solution allows the mission designer to consider a general tradespace of multiple sub-constellations (Walker, Heterogeneous Planes, SSO Train, string-of-pearls, etc.) and the algorithm itself finds what combinations of these sub-constellations work best in each coverage scenario.

- The work conducted in this thesis allowed the creation of a constellation design tool owned by NASA Goddard Spaceflight Center, which is used during Pre-Phase A and Phase A of Earth Observing missions and which will be released open source within the next year. This tool serves the purpose of efficiently exploring large tradespaces of constellation designs, with the ultimate goal of performing feasibility studies, trade studies and what-if analyses.

- This thesis proposed for the first time using a model-free control strategy based on Reinforcement Learning to autonomously manage memory buffers in a DTN node. Moreover, the proposed system model of a downlink to Earth that can be generally applied to any future distributed space mission to the Moon, Mars and beyond.

- This thesis showed that the methodology proposed to smartly manage DTN nodes using RL is robust/adaptive to dynamic traffic volumes and works successfully in different traffic conditions and network topologies. The efficiency of the approach presented was validated in two different scenarios: (1) a realistic lunar mission scenario

that models a Moon-to-Earth network with 46 nodes and two activities: A human outpost on the lunar surface and a mining operation on the inside of a crater; and (2) an Earth Observation mission scenario which consisted of a Walker constellation, formed by 24 satellites distributed across 3 planes, observing floods in 5 global regions.

- The use of RL within the management of DTN greatly improved network performance, optimizing the amount of successfully delivered bundles, minimizing allocation of resources to the different links while avoiding failures in the form of memory overflows. In fact, the RL policies were shown to outperform other more traditional non-RL approaches used in network management in both lunar and Earth observing scenarios.

4.3 Limitations and Future work

There are many opportunities for future work in the area of tradespace exploration strategies, and in particular MOEAs, for the design of DSMs. First, this thesis only considered domain independent operators to generate new solutions or offspring from an archive of parent chromosomes. In the last few years, there has been a lot of advances in Adaptive Operator Selection (AOS) strategies and Knowledge-Driven Optimization (KDO). The former applies machine learning methods during the optimization process to efficiently select operators and the latter extracts in real-time design rules that define relationships between design variables that are common in high-quality solutions. Then these rules or design principles are used to create heuristics in the form of new operators during the optimization in order to converge to the good regions of the tradespace faster. While this thesis has focused on developing efficient problem formulations for the constellation and orbit design of DSMs and not as much on the evolutionary algorithm used itself, the next step could involve developing a new KDO/AOS framework that not only uses the evolutionary formulations proposed in this thesis but also leverage the power of more intelligent evolutionary algorithms. In other words, a KDO algorithm could run a data mining algorithm every now and then to discover

features in the high quality solutions in the current Pareto Front and use these features to create new operators to generate new chromosome designs. For instance, the data mining algorithm could discover that, for a particular problem, the high-quality solutions contain a hybrid walker configuration formed by a high inclination Walker constellation and a low inclination one. Or it could be that the solutions in the Pareto front mix satellites at high and low altitudes through a heterogeneous plane configuration. Then, the MOEA could use this information to create knowledge-dependent operators that modify chromosome solutions that do not contain these design features and evolve them to new designs that do meet this criteria.

Another slight improvement could be made to the hybrid chromosome proposed in this work. While the heterogeneous plane sub-constellation chromosome (which could potentially be contained inside the hybrid chromosome) was designed as a variable length chromosome, the hybrid chromosome as a whole was not. In fact, when considering the space of hybrid constellations formed by N constellation sub-types, the way the hybrid chromosome unselects a certain constellation sub-type to explore a hybrid design with $m < N$ sub-constellations is by assigning a 0 to the number of satellites variable. As discussed in Chapter 2, this is a source of chromosome redundancy that can lead to inefficiencies and slow down the convergence rate during the search of high-quality designs. A good challenge to solve in the future would be to create a variable-length hybrid chromosome with the respective set of operators and hopefully obtain an improved version of the evolutionary formulations presented in this thesis.

There are also many opportunities for future work in the area of communications operations in DSMs, specifically in the use of Artificial Intelligence to generate intelligent strategies to optimize the management of DTN nodes. So far, the intelligent DTN management policies developed in this thesis through RL are currently being used in the Jet Propulsion Laboratory for benchmarking purposes. The reason why the proposed methodology is still

not planned to be launched in space is due to the lack of interpretability of machine learning models. As of today, Neural Networks (NN) are still seen as a big black box and it is hard to understand their behavior. Ultimately, the intelligent agents presented in this work take decisions to optimize the network performance of DTNs, avoiding failures and memories overflows based on decisions or actions dictated by a well-trained NN. But, what if, by launching this NN to space, the system encountered a very unlikely situation not visited during the training of the NN that therefore led to an unrecognized state by the NN, which in turn took a catastrophic decision that led to system failure? This poses a very big problem in the application of AI and ML in space applications and that is why, despite offering worse performance than AI solutions, big space agencies such as NASA still choose non-AI based (such as rule or heuristics based decision making strategies) that are more interpretable and offer lower risk of system failure. Further research in better understanding machine learning models would lead to a higher trust by experts and space agencies in intelligent systems based on AI, which would increase their presence in space.

Another issue that could arise when implementing the RL methodology proposed in this thesis is the availability for an intelligent centralized controller to observe the complete state of the network. In the Lunar scenario, it was observed that, unless the intelligent DTN node had state information about the memories of all neighbor nodes, their decisions tended to be greedy to remain at a good memory regime but congesting the other nodes of the network. Similarly, in the Earth Observing satellite constellation scenario, it was assumed that the centralized ground controller had information about all radios and buffer utilization state of all network entities. Unquestionably, this could represent a challenge in the space environment, specially in deep space exploration missions, where distances are vast, communications delays are long and interruptions between transceiver and receiver are frequent. Therefore, an opportunity for future work would be to incorporate uncertainty to the state representation and implement an efficient RL based partially observable control system. In addition, de-

spite Deep Q-Learning (DQL) was shown to work very well when managing DTNs, other RL algorithms such as Actor-Critic, Trust-Region Policy Optimization (TRPO) or Proximal policy Optimization (PPO) could be studied, and their performance compared against DQL.

Finally, the DTN management methods developed and presented in this thesis were based on a medium fidelity Python-based simulator for DTN developed at JPL and called DtnSim. This simulation environment allowed us to simulate the transmission of traffic flows between the different nodes of the system as well as to accurately represent the memory state of all entities forming the system. While this simulator allowed to investigate for the first time into the applicability of RL to manage buffers in DTN nodes, the next step involves adding this AI layer into the Interplanetary Overlay Network (ION), JPL's implementation of the DTN protocol stack for spacecraft. Once this is done, some hardware testing needs to be completed in order to increase the Technology Readiness Level (TRL) of the software solution presented in this thesis.

REFERENCES

[1] D. Selva and D. Krejci, "A survey and assessment of the capabilities of cubesats for earth observation," *Acta Astronautica*, vol. 74, pp. 50–68, 2012.

[2] I. Del Portillo, B. G. Cameron, and E. F. Crawley, "A technical comparison of three low earth orbit satellite constellation systems to provide global broadband," *Acta Astronautica*, vol. 159, pp. 123–135, 2019.

[3] R. Sandau, "Status and trends of small satellite missions for earth observation," *Acta Astronautica*, vol. 66, no. 1-2, pp. 1–12, 2010.

[4] S. S. Board, N. R. Council, *et al.*, *The role of small satellites in NASA and NOAA Earth Observation Programs*. National Academies Press, 2000.

[5] L. Di, K. Moe, and T. L. van Zyl, "Earth observation sensor web: An overview," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 3, no. 4, pp. 415–417, 2010.

[6] J. Le Moigne, "Distributed spacecraft missions (dsm) technology development at nasa goddard space flight center," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 293–296, IEEE, 2018.

[7] M. Drusch, U. Del Bello, S. Carlier, O. Colin, V. Fernandez, F. Gascon, B. Hoersch, C. Isola, P. Laberinti, P. Martimort, *et al.*, "Sentinel-2: Esa's optical high-resolution mission for gmes operational services," *Remote sensing of Environment*, vol. 120, pp. 25–36, 2012.

[8] D. Poursanidis and N. Chrysoulakis, "Remote sensing, natural hazards and the contribution of esa sentinels missions," *Remote Sensing Applications: Society and Environment*, vol. 6, pp. 25–38, 2017.

[9] J. M. Carreiras, S. Quegan, T. Le Toan, D. H. T. Minh, S. S. Saatchi, N. Carvalhais, M. Reichstein, and K. Scipal, "Coverage of high biomass forests by the esa biomass mission under defense restrictions," *Remote Sensing of Environment*, vol. 196, pp. 154–162, 2017.

[10] M. G. O'Neill and A. L. Weigel, "Assessing fractionated spacecraft value propositions for earth imaging space missions," *Journal of Spacecraft and Rockets*, vol. 48, no. 6, pp. 974–986, 2011.

[11] J. Le Moigne, J. C. Adams, and S. Nag, "A new taxonomy for distributed spacecraft missions," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 872–883, 2020.

[12] J. Puig-Suari, C. Turner, and W. Ahlgren, "Development of the standard cubesat deployer and a cubesat class picosatellite," in *2001 IEEE aerospace conference proceedings (Cat. No. 01TH8542)*, vol. 1, pp. 1–347, IEEE, 2001.

[13] J. Singer, G. Norris, and J. J. Pelfrey, "Enabling science and deep space exploration through space launch system secondary payload opportunities," in *14th International Conference on Space Operations*, p. 2585, 2016.

[14] N. Crisp, K. Smith, and P. Hollingsworth, "Small satellite launch to leo: a review of current and future launch systems," *Transactions of the Japan Society for Aeronautical and Space Sciences, Aerospace Technology Japan*, vol. 12, no. ists29, pp. Tf_39–Tf_47, 2014.

[15] J. Crusan and C. Galica, "Nasa's cubesat launch initiative: Enabling broad access to space," *Acta Astronautica*, vol. 157, pp. 51–60, 2019.

[16] A. Marinan, A. Nicholas, and K. Cahoy, "Ad hoc cubesat constellations: Secondary launch coverage and distribution," in *2013 IEEE aerospace conference*, pp. 1–15, IEEE, 2013.

[17] C. S. Ruf, S. Gleason, Z. Jelenak, S. Katzberg, A. Ridley, R. Rose, J. Scherrer, and V. Zavorotny, "The cygnss nanosatellite constellation hurricane mission," in *2012 IEEE International Geoscience and Remote Sensing Symposium*, pp. 214–216, IEEE, 2012.

[18] W. J. Blackwell, S. Braun, R. Bennartz, C. Velden, M. DeMaria, R. Atlas, J. Dunion, F. Marks, R. Rogers, B. Annane, *et al.*, "An overview of the tropics nasa earth venture mission," *Quarterly Journal of the Royal Meteorological Society*, vol. 144, pp. 16–26, 2018.

[19] E. Gill, P. Sundaramoorthy, J. Bouwmeester, B. Zandbergen, and R. Reinhard, "Formation flying within a constellation of nano-satellites: The qb50 mission," *Acta Astronautica*, vol. 82, no. 1, pp. 110–117, 2013.

[20] "Planet labs." `https://www.planet.com/`, 2021.

[21] C. Boshuizen, J. Mason, P. Klupar, and S. Spanhake, "Results from the planet labs flock constellation," 2014.

[22] "Capella space." `https://www.capellaspace.com/`, 2021.

[23] "One web." `https://www.oneweb.world/`, 2021.

[24] "Ses." `https://www.ses.com/`, 2021.

[25] M. Dwyer, D. Selva, I. del Portillo, M. Sanchez-Net, B. Cameron, Z. Szajnfarber, and E. Crawley, "Exploring the trade-offs of aggregated versus disaggregated architectures for environmental monitoring in low-earth orbit," in *AIAA SPACE 2014 Conference and Exposition*, p. 4416, 2014.

[26] D. Selva, B. G. Cameron, and E. F. Crawley, "Rule-based system architecting of earth observing systems: earth science decadal survey," *Journal of Spacecraft and Rockets*, vol. 51, no. 5, pp. 1505–1521, 2014.

[27] J. G. Walker, "Circular orbit patterns providing continuous whole earth coverage," tech. rep., ROYAL AIRCRAFT ESTABLISHMENT FARNBOROUGH (UNITED KINGDOM), 1970.

[28] P. G. Buzzi and D. Selva, "Evolutionary formulations for design of heterogeneous earth observing constellations," in *2020 IEEE Aerospace Conference*, pp. 1–10, 2020.

[29] H. W. Lee, S. Shimizu, S. Yoshikawa, and K. Ho, "Satellite constellation design methodology for optimal regional coverage," *arXiv preprint arXiv:1910.00672*, 2019.

[30] C. A. C. Coello, G. B. Lamont, D. A. Van Veldhuizen, *et al.*, *Evolutionary algorithms for solving multi-objective problems*, vol. 5. Springer, 2007.

[31] J. R. Wertz, D. F. Everett, and J. J. Puschell, *Space Mission Engineering: The New SMAD*. Space Technology Library, Microcosm Press, 2011.

[32] P. G. Buzzi, D. Selva, N. Hitomi, and W. J. Blackwell, "Assessment of constellation designs for earth observation: Application to the TROPICS mission," *Acta Astronautica*, vol. 161, pp. 166–182, aug 2019.

[33] V. L. Foreman, J. Le Moigne, and O. De Weck, "A survey of cost estimating methodologies for distributed spacecraft missions," in *AIAA SPACE 2016*, p. 5245, 2016.

[34] P. Stadter, A. Chacos, R. Heins, G. Moore, E. Olsen, and M. Asher, "Confluence of navigation, communication, and control in distributed spacecraft systems," in *2001 IEEE Aerospace Conference Proceedings (Cat. No. 01TH8542)*, vol. 2, pp. 2–563, IEEE, 2001.

[35] S. Nag, A. Aguilar, R. Akbar, A. Azemati, J. Frank, R. Levinson, A. Li, M. Moghaddam, V. Ravindra, and D. Selva, "D-shield: Distributed spacecraft with heuristic intelligence to enable logistical decisions," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, 2020.

[36] D. Cellucci, N. B. Cramer, and J. D. Frank, "Distributed spacecraft autonomy," in *ASCEND 2020*, p. 4232, 2020.

[37] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 27–34, 2003.

[38] E. Harkavy and M. S. Net, "Utilizing reinforcement learning to autonomously mange buffers in a delay tolerant network node," in *2020 IEEE Aerospace Conference*, pp. 1–8, IEEE, 2020.

[39] P. Garcia Buzzi, D. Selva, and M. Sanchez-Net, "Exploring reinforcement learning for autonomous delay tolerant network management," in *2020 AIAA ASCEND*, AIAA, 2020.

[40] E. Crawley, B. Cameron, and D. Selva, *System architecture: strategy and product development for complex systems*. Prentice Hall Press, 2015.

[41] A. M. Ross and D. E. Hastings, "11.4. 3 the tradespace exploration paradigm," in *INCOSE international Symposium*, vol. 15, pp. 1706–1718, Wiley Online Library, 2005.

[42] J. Le Moigne, P. Dabney, O. De Weck, V. Foreman, P. Grogan, M. Holland, S. Hughes, and S. Nag, "Tradespace analysis tool for designing constellations (TAT-C)," *International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 2017-July, pp. 1181–1184, 2017.

[43] D. A. Van Veldhuizen and G. B. Lamont, "Evolutionary computation and convergence to a pareto front," in *Late breaking papers at the genetic programming 1998 conference*, pp. 221–228, Citeseer, 1998.

[44] M.-J. Deutsch and J. S. Nichols, "Advanced approach to concept and design studies for space missions," *Astrophysics and Space Science*, vol. 273, no. 1, pp. 201–206, 2000.

[45] C. Iwata, S. Infeld, J. M. Bracken, M. McGuire, C. McQuirck, A. Kisdi, J. Murphy, B. Cole, and P. Zarifian, "Model-based systems engineering in concurrent engineering centers," in *AIAA SPACE 2015 Conference and Exposition*, p. 4437, 2015.

[46] R. E. Oberto, E. Nilsen, R. Cohen, R. Wheeler, P. DeFlono, and C. Borden, "The nasa exploration design team: blueprint for a new design paradigm," in *2005 IEEE Aerospace Conference*, pp. 4398–4405, IEEE, 2005.

[47] J. K. Ziemer, J. Ervin, and J. Lang, "Exploring mission concepts with the jpl innovation foundry a-team," in *AIAA Space 2013 Conference and Exposition*, p. 5431, 2013.

[48] W. Mason, V. Coverstone-Carroll, and J. Hartmann, "Optimal earth orbiting satellite constellations via a pareto genetic algorithm," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, p. 4381, 1998.

[49] G. Confessore, M. Di Gennaro, and S. Ricciardelli, "A genetic algorithm to design satellite constellations for regional coverage," in *Operations Research Proceedings*, pp. 35–41, Springer, 2001.

[50] M. Asvial, R. Tafazolli, and B. G. Evans, "Non-GEO satellite constellation design with satellite diversity using genetic algorithm," *20th AIAA International Communication Satellite Systems Conference and Exhibit*, no. May 2002, pp. 1–7, 2002.

[51] W. R. Whittecar and M. P. Ferringer, "Global coverage constellation design exploration using evolutionary algorithms," *AIAA/AAS Astrodynamics Specialist Conference 2014*, no. August, pp. 1–20, 2014.

[52] I. Meziane-Tani, G. Métris, G. Lion, A. Deschamps, F. T. Bendimerad, and M. Bekhti, "Optimization of small satellite constellation design for continuous mutual regional coverage with multi-objective genetic algorithm," *International Journal of Computational Intelligence Systems*, vol. 9, no. 4, pp. 627–637, 2016.

[53] T. Savitri, Y. Kim, S. Jo, and H. Bang, "Satellite Constellation Orbit Design Optimization with Combined Genetic Algorithm and Semianalytical Approach," *International Journal of Aerospace Engineering*, vol. 2017, no. May, 2017.

[54] S. W. Paek, S. Kim, and O. de Weck, "Optimization of reconfigurable satellite constellations using simulated annealing and genetic algorithm," *Sensors (Switzerland)*, vol. 19, no. 4, 2019.

[55] C. Han, S. Bai, S. Zhang, X. Wang, and X. Wang, "Visibility optimization of satellite constellations using a hybrid method," *Acta Astronautica*, no. January, pp. 0–1, 2019.

[56] D. Yan, P. You, C. Liu, S. Yong, and D. Guan, "Constellation multi-objective optimization design based on QoS and network stability in LEO satellite broadband networks," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 3, pp. 1260–1283, 2019.

[57] L. Agresti, S. Corpino, P. F. Stesina, P. D. B. Tridon, P. L. Savioli, and A. D'Ottavio, *Multi Objective Genetic Algorithm Optimization Of a Cubesats Constellation To Ensure Communication Services During Natural Disasters Over Europe*. PhD thesis, Politecnico di Torino, 2019.

[58] X. Xu, Y. Han, J. Luo, J. Wickert, and M. Asgarimehr, "Seeking optimal GNSS radio occultation constellations using evolutionary algorithms," *Remote Sensing*, vol. 11, no. 5, pp. 1–17, 2019.

[59] F. Keyvani and S. H. Torabi, "Design and simulation of regional navigation constellation with optimized mean DOP based on hybrid GEO and IGSO satellites," *International Journal of Aviation, Aeronautics, and Aerospace*, vol. 6, no. 2, 2019.

[60] C. D. Jilla and D. W. Miller, "Multi-objective, multidisciplinary design optimization methodology for distributed satellite systems," *Journal of Spacecraft and Rockets*, vol. 41, no. 1, pp. 39–50, 2004.

[61] S.-w. Cheng, L.-c. Shen, and J. Chen, "Optimization of regional coverage reconnaissance satellite constellation by improved nsga-ii algorithm," in *2008 International Conference on Intelligent Computation Technology and Automation (ICICTA)*, vol. 1, pp. 660–664, IEEE, 2008.

[62] M. P. Ferringer and D. B. Spencer, "Satellite constellation design tradeoffs using multiple-objective evolutionary computation," *Journal of Spacecraft and Rockets*, vol. 43, no. 6, pp. 1404–1411, 2006.

[63] N. Hitomi, H. Bang, and D. Selva, "Extracting and applying knowledge with adaptive knowledge-driven optimization to architect an earth observing satellite system," in *AIAA Information Systems-AIAA Infotech@ Aerospace*, p. 0794, 2017.

[64] N. Hitomi and D. Selva, "Constellation optimization using an evolutionary algorithm with a variable-length chromosome," *IEEE Aerospace Conference Proceedings*, vol. 2018-March, pp. 1–12, 2018.

[65] D. Selva, "Knowledge-intensive global optimization of earth observing system architectures: a climate-centric case study," in *Sensors, Systems, and Next-Generation Satellites XVIII*, vol. 9241, p. 92411S, International Society for Optics and Photonics, 2014.

[66] M. P. Ferringer, R. S. Clifton, and T. G. Thompson, "Efficient and accurate evolutionary multi-objective optimization paradigms for satellite constellation design," *Journal*

of Spacecraft and Rockets, vol. 44, no. 3, pp. 682–691, 2007.

[67] C. W. Ahn, *Advances in evolutionary algorithms*. Springer, 2006.

[68] M. Ferringer, M. DiPrinzio, T. Thompson, K. Hanifen, and P. Reed, "A framework for the discovery of passive-control, minimum energy satellite constellations," in *AIAA/AAS Astrodynamics Specialist Conference*, p. 4158, 2014.

[69] J. Puig-Suari, G. Zohar, and K. Leveque, "Deployment of cubesat constellations utilizing current launch opportunities," 2013.

[70] M. Fakoor, M. Bakhtiari, and M. Soleymani, "Optimal design of the satellite constellation arrangement reconfiguration process," *Advances in Space Research*, vol. 58, no. 3, pp. 372–386, 2016.

[71] C. L. Leonard, W. M. Hollister, and E. V. Bergmann, "Orbital formationkeeping with differential drag," *Journal of Guidance, Control, and Dynamics*, vol. 12, no. 1, pp. 108–113, 1989.

[72] R. Bevilacqua and M. Romano, "Rendezvous maneuvers of multiple spacecraft using differential drag under j2 perturbation," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 6, pp. 1595–1607, 2008.

[73] B. S. Kumar, A. Ng, K. Yoshihara, and A. De Ruiter, "Differential drag as a means of spacecraft formation control," in *Aerospace Conference, 2007 IEEE*, pp. 1–9, 2007.

[74] S. Varma and K. D. Kumar, "Multiple satellite formation flying using differential aerodynamic drag," *Journal of Spacecraft and Rockets*, vol. 49, no. 2, pp. 325–336, 2012.

[75] M. Horsley, S. Nikolaev, and A. Pertica, "Small satellite rendezvous using differential lift and drag," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 2, pp. 445–453, 2013.

[76] O. Ben-Yaacov and P. Gurfil, "Long-term cluster flight of multiple satellites using differential drag," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 6, pp. 1731–1740, 2013.

[77] T. Finley, D. Rose, K. Nave, W. Wells, J. Redfern, R. Rose, and C. Ruf, "Techniques for leo constellation deployment and phasing utilizing differential aerodynamic drag," *Advances in the Astronautical Sciences*, vol. 150, pp. 1397–1411, 2014.

[78] I.-S. Chang, "Space launch vehicle reliability," *Crosslink*, pp. 22–32, 2005.

[79] T. Ely, R. Anderson, Y. Bar-Sever, D. Bell, J. Guinn, M. Jah, P. Kallemeyn, E. Levene, L. Romans, and S. Wu, "Mars network constellation design drivers and strategies," in *AAS/AIAA Astrodynamics Specialist Conference*, p. 301, 1999.

[80] E. Lansard, E. Frayssinhes, and J.-L. Palmade, "Global design of satellite constellations: a multi-criteria performance comparison of classical walker patterns and new design patterns," *Acta Astronautica*, vol. 42, no. 9, pp. 555–564, 1998.

[81] J. Grosshans and M. F. Barschke, "Mission concept of a nanosatellite constellation for global wildfire monitoring," in *AIAA SPACE and Astronautics Forum and Exposition*, p. 5267, 2017.

[82] M. P. Ferringer, D. B. Spencer, and P. Reed, "Many-objective reconfiguration of operational satellite constellations with the large-cluster epsilon non-dominated sorting genetic algorithm-ii," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pp. 340–349, IEEE, 2009.

[83] G. F. Dubos and J. H. Saleh, "Comparative cost and utility analysis of monolith and fractionated spacecraft using failure and replacement markov models," *Acta Astronautica*, vol. 68, no. 1-2, pp. 172–184, 2011.

[84] J.-F. Castet and J. H. Saleh, "Interdependent multi-layer networks: Modeling and survivability analysis with applications to space-based networks," *PloS one*, vol. 8, no. 4, p. e60402, 2013.

[85] S. M. Hull, "End of mission considerations," Preprint GSFC.BOOK.7496.2012, NASA Goddard Space Flight Center, Greenbelt, MD, United States, 01 2013.

[86] D. M. Cornwell, "Nasa's optical communications program for 2017 and beyond," in *2017 IEEE International Conference on Space Optical Systems and Applications (ICSOS)*, pp. 10–14, IEEE, 2017.

[87] W. A. Imbriale and W. A. Imbriale, *Large antennas of the deep space network*. Wiley Online Library, 2003.

[88] C. F. Kwadrat, W. D. Horne, and B. L. Edwards, "Inter-satellite communications considerations and requirements for distributed spacecraft and formation flying systems," 2002.

[89] D. Baird, "Space communications: 7 things you need to know." `https://www.nasa.gov/feature/goddard/2020/space-communications-7-things-you-need-to-know`, 2021.

[90] L. J. Ippolito, "Radio propagation for space communications systems," *Proceedings of the IEEE*, vol. 69, no. 6, pp. 697–727, 1981.

[91] M. Toyoshima, "Trends in satellite communications and the role of optical free-space communications," *Journal of Optical Networking*, vol. 4, no. 6, pp. 300–311, 2005.

[92] A. Roy, T. Acharya, and S. DasBit, "Quality of service in delay tolerant networks: A survey," *Computer Networks*, vol. 130, pp. 121–133, 2018.

[93] R. S. Sutton, A. G. Barto, *et al.*, *Introduction to reinforcement learning*, vol. 135. MIT press Cambridge, 1998.

[94] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[95] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[96] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[97] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[98] K. Deb, A. Member, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm:NSGAII," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[99] L. Wang, Y. Wang, K. Chen, and H. Zhang, "Optimization of regional coverage reconnaissance satellite constellation by nsga-ii algorithm," in *2008 International Conference on Information and Automation*, pp. 1111–1116, IEEE, 2008.

[100] W. S. Adams and L. Rider, "Circular polar constellations providing continuous single or multiple coverage above a specified latitude," *Journal of the Astronautical Sciences*, vol. 35, pp. 155–192, June 1987.

[101] A. H. Ballard, "Rosette constellations of earth satellites," *IEEE Transactions on Aerospace and Electronic Systems*, no. 5, pp. 656–673, 1980.

[102] D. Mortari, M. P. Wilkins, and C. Bruccoleri, "The flower constellations," *Journal of Astronautical Sciences*, vol. 52, no. 1, pp. 107–127, 2004.

[103] K. Mott and J. Black, "Model-Based Heterogeneous Optimal Space Constellation Design," *2018 21st International Conference on Information Fusion, FUSION 2018*, pp. 602–609, 2018.

[104] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex systems*, vol. 9, no. 2, pp. 115–148, 1995.

[105] S. P. Hughes, R. H. Qureshi, S. D. Cooley, and J. J. Parker, "Verification and validation of the general mission analysis tool (gmat)," in *AIAA/AAS astrodynamics specialist conference*, p. 4151, 2014.

[106] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary computation*, vol. 10, no. 3, pp. 263–282, 2002.

[107] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer US, 2007.

[108] F. Biscani, D. Izzo, W. Jakob, GiacomoAcciarini, M. Märtens, M. C, A. Mereta, C. Kaldemeyer, S. Lyskov, S. Corlay, G. Acciarini, B. Pritchard, K. Manani, Acxz, J. Mabille, A. Huebl, M. López-Ibáñez, Jakirkham, J. Lee, Hulucc, Polygon, L. Č. Zajc, J. Adler, J. Travers, J. Jordan, I. Smirnov, H. Nguyen, F. Lema, E. O'Leary, and A. Mambrini, "esa/pagmo2: pagmo 2.11.4," sep 2019.

[109] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data mining and knowledge discovery*, vol. 8, no. 1, pp. 53–87, 2004.

[110] N. Hitomi, H. Bang, and D. Selva, "Adaptive Knowledge-Driven Optimization for Architecting a Distributed Satellite System," *Journal of Aerospace Information Systems*, vol. 15, pp. 485–500, jun 2018.

[111] N. Hitomi and D. Selva, "Incorporating expert knowledge into evolutionary algorithms with operators and constraints to design satellite systems," *Applied Soft Computing Journal*, vol. 66, no. May 2018, pp. 330–345, 2018.

[112] N. Hitomi and D. Selva, "A classification and comparison of credit assignment strategies in multiobjective adaptive operator selection," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 294–314, 2017.

[113] M. Sanchez net, "Welcome to dtnsim, python-based simulator for delay tolerant networking." `https://github.com/msancheznet/dtnsim`, 2019.

[114] J. Hu, R. Wang, X. Sun, Q. Yu, Z. Yang, and Q. Zhang, "Memory dynamics for dtn protocol in deep-space communications," *IEEE Aerospace and Electronic Systems Magazine*, vol. 29, no. 2, pp. 22–30, 2014.

[115] K. Scott and S. Burleigh, "Bundle protocol specification," 2007.

[116] M. Ramadas, S. Burleigh, S. Farrell, *et al.*, "Licklider transmission protocol-specification," *IETF request for comments RFC*, vol. 5326, 2008.

[117] W.-B. Pöttner, J. Morgenroth, S. Schildt, and L. Wolf, "Performance comparison of dtn bundle protocol implementations," in *Proceedings of the 6th ACM workshop on Challenged networks*, pp. 61–64, 2011.

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

[119] V. Mahendran, T. Praveen, and C. S. R. Murthy, "Buffer dimensioning of delay-tolerant network nodes-a large deviations approach," in *International Conference on Distributed Computing and Networking*, pp. 502–512, Springer, 2012.

[120] V. Mahendran and C. S. R. Murthy, "Buffer dimensioning of dtn replication-based routing nodes," *IEEE communications letters*, vol. 17, no. 1, pp. 123–126, 2012.

[121] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[122] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," in *2006 IEEE International Conference on Autonomic Computing*, pp. 65–73, IEEE, 2006.

[123] G. Tesauro *et al.*, "Online resource allocation using decompositional reinforcement learning," in *AAAI*, vol. 5, pp. 886–891, 2005.

[124] S. Abdallah and V. Lesser, "Learning the task allocation game," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pp. 850–857, 2006.

[125] T. Li, Z. Xu, J. Tang, and Y. Wang, "Model-free control for distributed stream data processing using deep reinforcement learning," *arXiv preprint arXiv:1803.01016*, 2018.

[126] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, pp. 50–56, 2016.

[127] W. Chen, Y. Xu, and X. Wu, "Deep reinforcement learning for multi-resource multi-machine job scheduling," *arXiv preprint arXiv:1711.07440*, 2017.

[128] Y. Yang, Y. Li, K. Li, S. Zhao, R. Chen, J. Wang, and S. Ci, "Decco: Deep-learning enabled coverage and capacity optimization for massive mimo systems," *IEEE Access*, vol. 6, pp. 23361–23371, 2018.

[129] Y. Yin, L. Wang, and E. Gelenbe, "Multi-layer neural networks for quality of service oriented server-state classification in cloud servers," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1623–1627, IEEE, 2017.

[130] L. Wang and E. Gelenbe, "Adaptive dispatching of tasks in the cloud," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 33–45, 2015.

[131] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," in *2006 IEEE International Conference on Autonomic Computing*, pp. 65–73, IEEE, 2006.

[132] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines." `https://github.com/hill-a/stable-baselines`, 2018.

[133] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov, "Openai baselines." `https://github.com/openai/baselines`, 2017.

[134] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

APPENDIX A

EXPERT POLICY

This appendix contains all the rules that conform the **expert** policy and determine what actions to take depending on the network state s. As mentioned in Section 3.4.3, since the memory utilization state variables are continuous, they were discretized to three possible values: LOW ($u < 50\%$), OKAY ($50\% < u < 80\%$) and HIGH ($u > 80\%$). Similarly, instead of using the 10 possible values of data rates, only two cases (and their negated values) are used to define these rules: MINIMUM and MAXIMUM. The exhaustive list of rules is the following:

1. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

2. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

3. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

4. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *decrease* Rb_{in}

5. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *decrease* Rb_{in}

6. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

7. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *decrease* Rb_{in}

8. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *decrease* Rb_{in}

9. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

10. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{out}

11. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *do nothing*

12. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

13. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{out}

14. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{in}

15. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

16. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *do nothing*

17. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{in}

18. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

19. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

20. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *do nothing*

21. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *drop packets*

22. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

23. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{in}

24. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{in}

25. **IF** (memory neighbor is LOW and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

26. **IF** (memory neighbor is LOW and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{in}

27. **IF** (memory neighbor is LOW and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{in}

28. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

29. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

30. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

31. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

32. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

33. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

34. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

35. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

36. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

37. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{out}

38. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *do nothing*

39. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

40. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{out}

41. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *do nothing*

42. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

43. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{out}

44. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *do nothing*

45. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

46. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

47. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *do nothing*

48. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *drop packets*

49. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

50. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *do nothing*

51. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{in}

52. **IF** (memory neighbor is OKAY and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

53. **IF** (memory neighbor is OKAY and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *do nothing*

54. **IF** (memory neighbor is OKAY and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{in}

55. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{in}

56. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{in}

57. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

58. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{in}

59. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{in}

60. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

61. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

62. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *do nothing*

63. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MINIMUM) **THEN** *increase* Rb_{out}

64. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{in}

65. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{in}

66. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

67. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{in}

68. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{in}

69. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

70. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *decrease* Rb_{out}

71. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *do nothing*

72. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is not MINIMUM nor MAXIMUM) **THEN** *increase* Rb_{out}

73. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *increase* Rb_{in}

74. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *increase* Rb_{in}

75. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MINIMUM and Rb_{out} is MAXIMUM) **THEN** *increase* Rb_{in}

76. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *increase* Rb_{in}

77. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *increase* Rb_{in}

78. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is not MINIMUM nor MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *increase* Rb_{in}

79. **IF** (memory neighbor is HIGH and memory RL node is LOW and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *decrease* Rb_{out}

80. **IF** (memory neighbor is HIGH and memory RL node is OKAY and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *do nothing*

81. **IF** (memory neighbor is HIGH and memory RL node is HIGH and Rb_{in} is MAXIMUM and Rb_{out} is MAXIMUM) **THEN** *drop packets*