# POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering


Master's Degree Thesis

# Dynamic Tasking for Satellite Constellation: A Multi-Agent based approach using cooperative auctions

Supervisors

Prof. Luciano LAVAGNO

Dott. Riccardo MADERNA

Dott.sa Federica PAGANELLI

Candidate

Yithzak ALARCON

December 2022

# Abstract

The growing demand for ever-larger satellite constellations poses significant challenges in terms of automation and coordination capabilities. New advances in onboard automation enable satellites to make highly autonomous decisions, providing the basis for complex applications such as dynamic task allocation in a satellite constellation. By coordinating these satellites through a negotiation-based approach, it is possible to improve space mission return and enable a more efficient allocation of complex tasks.

This thesis paves the road toward dynamic task allocation using a multi-agent based architecture and cooperative auctions. A constellation composed of autonomous satellites is modeled as a multi-agent system and the problem of allocating new service requests is formulated accordingly. The model comprises satellite's capabilities, on-board resources, and various constraints. Then, an auction algorithm is designed and implemented to enable a cooperation mechanism between satellites and efficiently distribute the incoming requests to speed up on-ground planning. Each agent bids on the announced service requests by combining the contributions of several bidding terms, each one modeling a specific motivator for the satellite (e.g., resource availability or the presence of conflicts). A set of test cases has been designed to ensure the code's integrity and the correctness of the auction execution.

A scalability analysis is presented in order to assess the model's performance for an increasing number of satellites interacting dynamically in a constellation. In addition, a complexity analysis is developed to evaluate the time complexity in an environment with many agents, each of which presents its own time-varying characteristics during the auction. Finally, a sensitivity analysis allows fine-tuning the scaling parameters of the functions of each bidding term, as well as the weights that each term should have in the overall bid, with the purpose of evaluating the influence and impact of each component on the overall performance.

**Keywords:** Satellite constellation tasking, Multi-Agent systems, Cooperative auctions, Dynamic task allocation

# Acknowledgements

First of all, I want to thank all my family, my parents and my brother, their unconditional support throughout this process has been fundamental to be where I found myself today. Thank you because you always had a word of motivation towards me and gave me that encouragement to overcome all the difficulties.

I would like to thank Prof. Lavagno for his blind trust and his availability whenever I needed him.

A special thanks to the entire AIKO team, especially Riccardo and Federica, thank you for allowing me to participate in this exciting project, for guiding me every day, for correcting me, and for letting me take part in every single discussion.

Thanks to my friends: Sara, Cesar, Pedro, Daniel, Mayra, Sergio, and Migue who have always supported me and helped me on this journey since I arrived in Torino and before. A special thanks to Sarita, "Mi gordis", because we went through all this adventure of the master's degree together since the bachelor's, and to Cesar, "Mi compa", for always giving me a hand every time I needed it.

*"Todo tiene su final."*
*Héctor Lavoe*

# Table of Contents

# List of Tables

# List of Figures

# Acronyms

**GTU**
Ground Tasking Unit

**MiRAGE**
Mission Replanning through Autonomous Goal Generation

**LEO**
Low Earth Orbit

**GEO**
Geostationary Earth Orbit

**MILP**
Mixed Integer Linear Program

**SoE**
Sequence of Events

**DP**
Dynamic Programming

**MDP**
Markov Decision Process

**Dec-POMDP**
Decentralized Partially-Observable Markov Decision Process

**MMDP**
Multi-agent Markov Decision Process

**MSS**

Multi-Satellite System

**SVM**

Support Vector Machine

**JD**

Julian Dates

**ESA**

European Space Agency

**SAR**

Synthetic Aperture Radar

# Chapter 1

# Introduction

The aerospace industry has been in a growing trend of developing new technologies for onboard process automation, intending to bring a better cost-effective performance guarantee. Missions that involve constant contact with ground are plagued with inefficiencies such as latency, disruptions in communication quality due to external factors, and **limited visibility windows**; In particular, the latter, as depicted in Figure 1.1, is a major constraint as the satellite status cannot be monitored once outside the contact window, which is why extensive research has been conducted on the automation of space missions due to the inability to respond and adapt to unforeseen events.

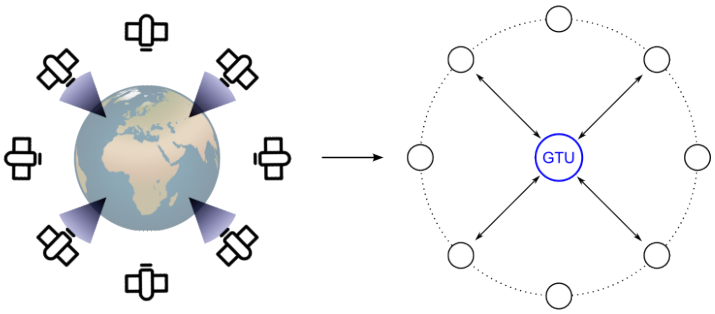



**Figure 1.1:** General scenario of a constellation of satellites with limited visibility windows

This thesis has been developed in collaboration with the new space company AIKO, which has been developing several products to provide satellites with the ability to make autonomous decisions, plan on-board tasks and operate in coordination with other nodes in the constellation to achieve more complex goals. One of these products, orbital_OLIVER, allows satellites to operate autonomously by analyzing data obtained from the environment and generating an optimal schedule.

Highly autonomous and intelligent satellites have the potential to significantly increase the return of space missions. On the other hand, a change in paradigm in how missions are operated is needed to realize these benefits. This is even more true once intelligent satellites are operated as part of a constellation. In this case, the need to coordinate these agents to achieve optimal task allocation in a constellation arises, which motivates the dynamic constellation tasking project that has been tackled by the present thesis.

This project investigates a new approach to constellation tasking to provide a better alternative to classic scheduling techniques by exploiting cooperation and dynamic decision-making capabilities of intelligent satellites, enabled by orbital_OLIVER. A new mathematical formulation based on negotiation is introduced, that takes in consideration the communication constraints, the limited availability of on-board resources, and sparse contact windows with ground mission control.

## 1.1 Starting point and thesis objectives

The starting point of this work is the orbital_OLIVER product [1], an artificial intelligence-based software that supports and augments space missions by making satellites independent from ground control [2]. This product gives the satellites the possibility to take autonomous decisions and plan their own schedule, making a central planning structure on ground no longer needed. Instead, the need for a process to assign tasks and to coordinate a constellation of agents that are autonomous, cooperative, and intelligent arises. The main objectives of the thesis are as follows:

1. To provide a well-descriptive formulation, modeling and scenario definition of the problem of dynamic task allocation in a satellite constellation with interacting intelligent agents (i.e., satellites integrated with orbital_OLIVER).

2. To propose a feasible solution to the problem of dynamic task allocation in a satellite constellation. In the space environment, satellites are constantly subjected to unforeseen events, state changes, and extreme conditions. Therefore, a tool that considers these conditions and circumstances is needed, also allowing satellites to coordinate and collaborate with each other to maximize the expected utility and achieve more complex goals. A theoretical framework based on agents' capabilities, multi-agent system modeling, and cooperative auctions is defined. This aims to achieve an approach that could improve the performance reported by the metrics (e.g., time complexity or bid evolution) presented in this work.

3. To present and discuss the outcomes of various simulations executed by the prototype implementation developed in collaboration with AIKO. The software is capable of running multiple routines simulating randomized instance of the problem based on predefined scenarios with the purpose of providing a preliminary assessment of the performance and scalability of the proposed solution, and to identify a roadmap for future development. Notably, the application is agnostic to a specific scenario, but can be configured to test the behavior of the developed.

## 1.2 Thesis structure

The structure of this thesis is organized as follows:

- **The second chapter** gives an overview of the state of the art focusing on classic and autonomous space mission operations, and multi-agent systems.
- **The third chapter** comprises the definition of the problem, the description of the different components involved, the conditions and constraints of the scenario, and the proposed mathematical modelling.
- **The fourth chapter** presents the implementation of the model, the description of the different features of the built software and the different mechanisms of the software framework.
- **The fifth chapter** shows the results of the different simulations and experiments to analyze the auction outcomes and performance.
- **The last chapter** draws the conclusions of the thesis comparing the general objectives with the experimental results. In addition, possible paths for future work and research is presented.

# Chapter 2

# State Of The Art

This chapter aims to give an overview of the state of the art of related fundamental topics before going into detail about the problem definition and architecture design. Thus, a literature survey regarding classic mission operations, autonomous constellation operations, and a multi-agent system theoretical basis are introduced.

## 2.1 Classic space mission operations

It is important to review the operation of classic space missions and identify the main characteristics and limitations this mechanism may have. This will also help us to define the environmental conditions that our model will face and give us an abstraction of the fundamental blocks that should compose the system. Especially since our main focus will be autonomous constellations operations and, therefore, it becomes necessary to establish operating conditions that would remain fixed regardless of whether the constellation is autonomous or not.

### 2.1.1 Types of satellite orbits

Two main types of mission operations are relevant for our case since we are modeling a scenario of a satellite constellation:

1. **Low Earth Orbit (LEO)**: The main focus of our work is oriented toward this type of operational concept; satellites orbit the earth at an altitude of approximately ∼500 km and with an **orbital period** of **∼90 min** [3], this last point is important when we need to simulate the sequences of contacts a satellite may have during its orbit. Also, it is important to highlight that this type of mission operation presents short contacts with ground because of the low orbit characteristic, therefore, only about **5 contacts per satellite**

with a duration of between 8 and 10 minutes are possible in a day with 1 or 2 ground stations. [3]

As an example, we can observe in Figure 2.1 the trajectory tracking of a GRACE satellite showing that only a few contacts are possible.

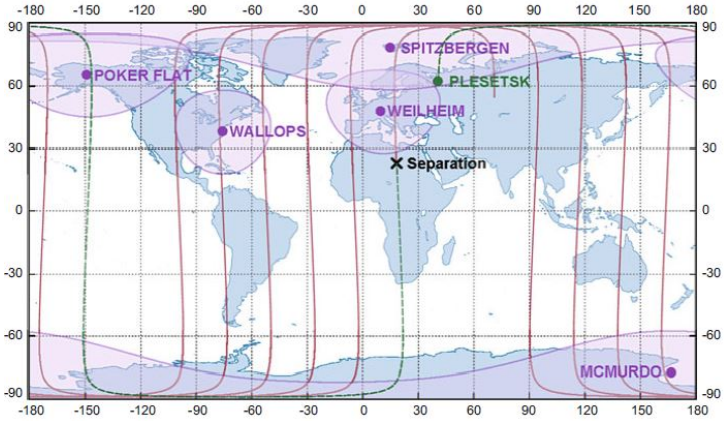



**Figure 2.1:** Ground tracking of GRACE satellite after launch (*green line* is the track before first acquisition over Weilheim) [3]

2. **Geostationary Earth Orbit (GEO)**: This type of operational concept, although not our main focus, is also important to consider as it has certain advantages over LEO; one of its main advantages is that it offers 24 hours of global coverage (except for some polar regions) [4], it means that orbit maneuvers, payload, and operations can be uploaded in real-time [3]. But, one of its main disadvantages is the time delay for a transmission to reach its destination, and also GEO satellites systems present more complexity due to the natural distance of the orbit from ground. As an example, we could observe in Figure 2.2 the global coverage range of a GEO satellite, and, of course, compare with the coverage amount in a LEO orbit shown before in Figure 2.1.

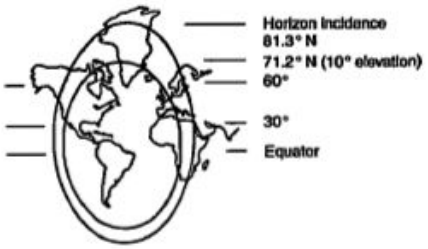



**Figure 2.2:** Coverage of one satellite in GEO orbit [4]

### 2.1.2 Planning problem

Although our methodology is based on the autonomy of a constellation of satellites acting as intelligent agents, the main purpose of this chapter is to give the reader a basis to compare the traditional approaches and the new methodology that will be developed throughout this work.

Standard mission planning prepares all the relevant activities (e.g., tasking scheduling, on-ground planning management, timeline generation) that happen during the mission, on board as well as on ground [3], this is no longer needed because of the high autonomy provided by orbital_OLIVER, but it is important to study the main blocks that compose a classic mission planning system control center, described as follows:

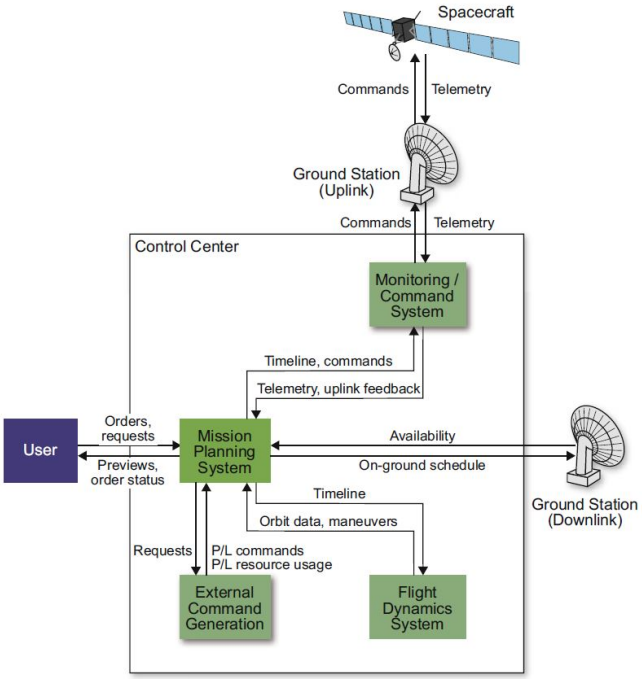



**Figure 2.3:** Classic mission planning system interfaces [3]

- **Mission planning system:** It is the central module that contains the planning model that is responsible for creating the different timelines depending on the type of mission and objects to be scheduled, i.e., specific instructions to be transmitted to the satellite to be followed. This central module has to be fed with the ground station availabilities, orbit and maneuver information, and payload configuration updates. [3]

- **Flight dynamics system:** This module contains real-time data of the computed orbit trajectory, maneuver execution, and attitude control. In some cases, this module could compute the command lists to be sent to the central mission planning system. [3]
- **Monitoring/Command system:** This module receives real-time telemetry from the spacecraft, and if the mission planning system does not generate a command sequence by itself, then this interface is needed to generate and transfer the list of commands to be executed by the satellite. It also received information about the status of the spacecraft and received feedback on whether they were uplinked and whether the commands were executed on board. [3]
- **External command generation:** This external module assists the mission planning system by generating lists of commands required after the different timelines have been generated, including additional information such as the required resource usage.

### 2.1.3 Classic constellation tasking

There has been extensive research and development in the creation of algorithms and methodologies for the scheduling problem for a large constellation of satellites. When there are no intelligent agents on board in a constellation, this produces a highly complex problem with multiple variables and constraints to take into account from ground, such as: the trajectory of the satellites, what will be the optimal sequence of events (SoE) to be executed for each one of them, the contacts that each satellite will have depending on the number of available ground stations, and the evolution of resource levels over time.

Several approaches have been studied and applied to variants of this problem, for instance, Augenstein et al. [5] formulated a Mixed Integer Linear Program (MILP) to solve a variation of the problem considering the imaging and contact allocation problem in a constellation of about 12 satellites, this solution optimally solves each local satellite imaging scheduling problem but also takes into consideration human operator intervention in the scheduling process; moreover, Boerkoel et al. [6] design a solution adding Dynamic Programming (DP) to the MILP estimates of the cumulative payloads and priorities obtained between contact opportunities, following a greedy approach and giving relevant behavior data of a satellites fleet. Shah et al. [7] also tackle this same variation by augmenting the problem to a constellation of over 100 satellites and 30 ground stations using a formulation involving MILPs. These works, however, provide optimal solutions for variations of a scheduling problem, and by not considering a scenario with autonomous agents in a constellation, it is possible to constrain the task allocation problem and assume

full knowledge of the satellite states. Other approaches have been studied using heuristic algorithms such as Ant Colony Optimization, Simulated Annealing, and genetic algorithms. [8], [9], [10]

One modern solution using a Reinforcement Learning approach is developed by Herrmann et al. [11], basically they modeled the constellation scheduling problem as a Markov Decision Process (MDP) that allows the agents to take action-rewarded decisions and deal with the uncertain environment changes having established policies if the problem is modeled as a decentralized partially observable Markov Decision Process (Dec-POMDP), however, the complexity to obtain an optimal solution for a finite-horizon is high, then they propose a Multi-agent Markov Decision Process (MMDP) that require full observability of the environment and also have a complex computational requirement but they reduced it by tackling the problem with a single Markov agent for training purposes and then providing the agents with a collective belief state, this formulation was considered as well for the definition of our tasking problem, however, we decided to keep the *Partially-Observable* characteristic for our problem definition and the *Multi-Agent* modeling that will be further developed in next chapters.

## 2.2 Autonomous constellation operations

With the recent growth in satellite deployment due to reduced component costs and ease of monitoring through software, there is a growing interest in automating a large constellation of satellites to perform increasingly complex tasks. At the moment, there are about 2000+ active satellites in orbit encouraging a growing demand for an autonomous satellite constellation that would bring about cost reductions by decoupling the normal operation of a constellation tasking allocations with ground monitoring [12], which would go from relying on human operators on ground supervising a huge number of satellites and intervening in their tasking processes, to a few supervisors monitoring the status of the constellation while it autonomously allocates tasks and returns feedback reports.

The demand for autonomous operation is increasing as a result of a shift towards a "mass-production" of small, standardized, modular, and distributed spacecraft sub-systems, resulting in an inevitable paradigm change in how satellites constellations are coordinated, designed, and operated. With an increasing need to operate the entire system as a whole, automation will provide significant benefits such as realizability, redundancy, and, as previously stated, cost efficiency. Figure 2.4 depicts the increasing demand over time for satellite mass production and deployment plans. However, this paradigm shift in the way constellations operate brings with it high-complexity challenges such as basic communication tasks, ground resource allocation, coordination management and anomaly probabilities. [13]

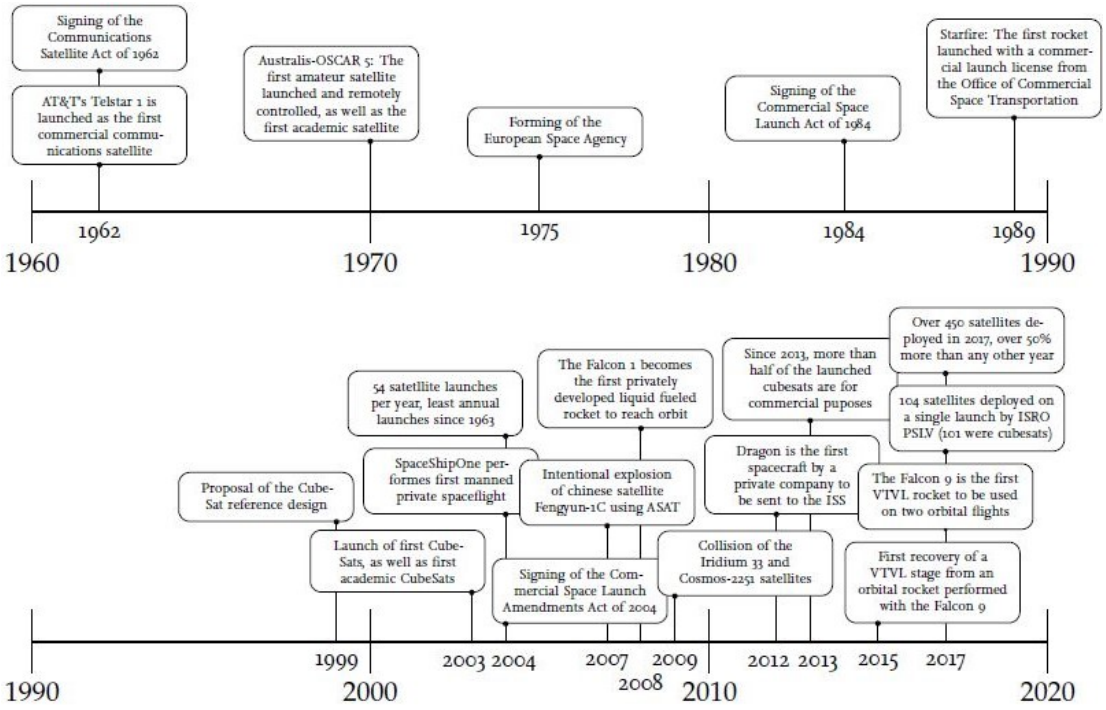



**Figure 2.4:** Timeline evolution of space stakeholders and significant events [13]

### 2.2.1 Onboard automation

An important point in the development of an autonomous constellation of satellites is the availability of agents with decision-making capabilities onboard, this allows us to simplify the mission planning process on ground, and to better take into account several difficulties and uncertainties inherent to the environment in which they operate, in our case, the highly hostile space.

To address the task scheduling problem on board, several methodologies have been developed that study in sufficient detail the considerations and requirements to be taken for a space mission. Castano et al. [14] developed a case study to highlight the necessary changes that would allow operators and scientists to manage an autonomous spacecraft. They designed a common model to understand, reconstruct, and explain the decisions made onboard and the state of the spacecraft. Even though the main focus of this work is interplanetary orbits, which is outside the scope of our work, the presented concept of operations remains relevant since one of the major difficulties of dealing with autonomous agents is understanding the decisions taken by them and the outcomes produced. As illustrated in Figure 2.5, a critical point to address when operating highly autonomous satellites, e.g endowed with orbital_OLIVER is that, given the same input (i.e., a timed list of

tasks to execute), one could obtain many nominal outcomes (i.e., not all and only the planned tasks will be executed by the satellite).

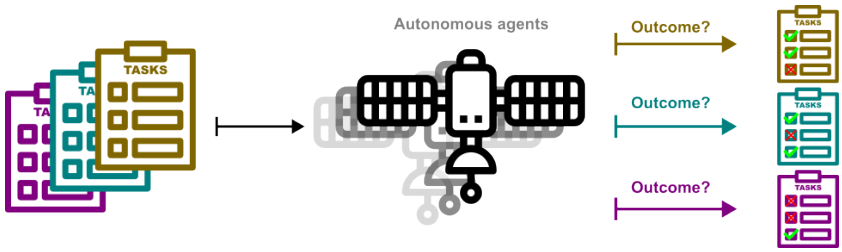



**Figure 2.5:** Pictorial diagram of task allocation between an autonomous constellation with unknown outcome

Another approach followed by Zheng et al. [15], is modeling a multi-satellite system with a game-theoretical based approach, assuming each satellite is a self-interested unique unit. They also proposed various negotiation mechanisms to cooperate in a distributed and decentralized system structure, this formulation would present certain restrictions, for instance, a negotiation procedure where agents act as self-interested only starts if the whole group is available to cooperate, another aspect is also that satellites will not always be able to communicate with each other all the time, therefore, they present other negotiation formulations for a centralized and decentralized structure, different formulations for different topologies of multi-satellite systems (MSS).

### 2.2.2 orbital_OLIVER

This chapter aims to describe the onboard autonomy product developed by the aerospace company AIKO, which this thesis has been developed in collaboration with. The orbital_OLIVER product is our main starting point and a fundamental block in our methodology since it enables a set of autonomy capabilities onboard satellites that have an impact on the behavior of an autonomous constellation of satellites.

This product is an intelligent onboard software that supports and augments space missions by making satellites highly independent from ground control [1]. This software operates by analyzing data from the satellite and its operating environment to generate and execute the optimal schedule of different activities; consists of three fundamental pillars [2]:

1. Onboard data processing
2. Operations planning
3. Autonomous control

Based on this, it comprises a simple cognitive architecture of complex space systems, providing satellites with the ability to [2]:

a) Sense the environment and its status (through onboard data processing)

b) Plan tasks according to acquired or inferred knowledge (operations planning)

c) Self-manoeuvre in the orbital environment (autonomous control)

As we can see, this formulation allows us to assume intelligent and autonomous satellites, capable of making decisions on their own and responding to unforeseen situations, as well as planning their on-board schedule. These assumptions will be used in the definition of our problem and the design of our methodology and architecture.

Figure 2.6, shows a level diagram illustrating the planning manager services contained in orbital_OLIVER. In which it is triggered by pending goals to be processed as a sequence of tasks and subtasks to generate an optimal set of mission goals. The optimal set of goals is obtained considering the current state of the agent, external events and timelines, and visibility windows. This framework enables considering both causal and temporal constraints, limited resources, and different state variables during the optimal and flexible plan definition. [2]

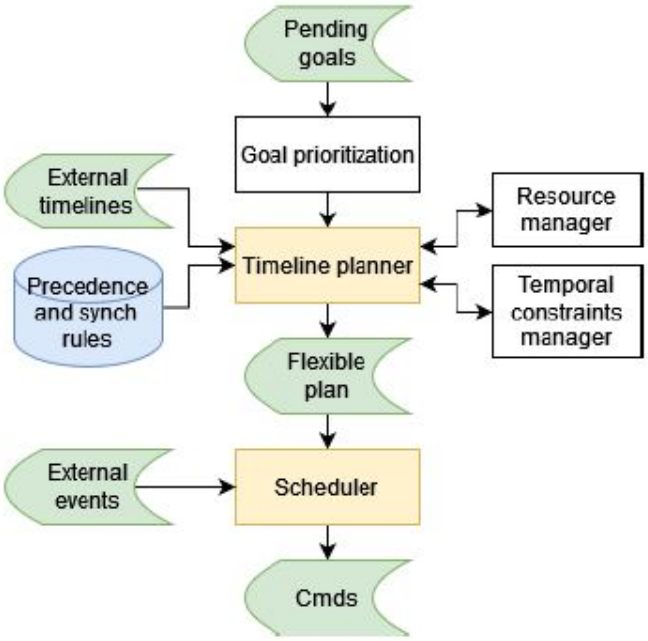



**Figure 2.6:** High-Level diagram of the Planning Manager Service [2]

A key point of this architecture is that the planning system allows OLIVER to be fully configurable to adapt to any kind of mission scenario, irrespective of the mission concept.

Previously, we could observe different methods that either took a variation of the onboard scheduling problem considering a certain number of antennas, ground stations, and satellites; or focused on considering for the constellation a type of onboard services such as imagery or another type. This is the reason why the approach presented by this software creates a total paradigm shift; since as previously seen it could involve complete scenarios with multi-satellite systems (MSS), varying levels of resources, payloads, and different contact windows; considerations that in the literature only resulted in partial solutions to a constrained-problem, or novel methods whose computational complexity tended to increase if the system presented more and more interacting elements.

Finally, another highly relevant feature is that the planning system is able to handle different types of resources to ensure the feasibility of the produced mission plan [2], which allows agents to reason about resource consumption in real-time and exploit them more efficiently. This has been considered in our methodology, designing a structure able to contain heterogeneous agents interacting and cooperating in an autonomous constellation.

### 2.2.3 Centralized/Decentralized planning

Several topologies could be found to obtain a solution to the task allocation problem in a constellation of satellites, this chapter aims to give an overview of the literature review for the main advantages and disadvantages of a centralized and a decentralized approach.

The centralized approach is widely used, which typically requires a central agent who is in charge of coordinating the other agents and proceeding to perform the task allocation, this is the approach followed in this thesis. More specifically, Zheng et al. [15] mention that one kind of centralized proposal for task allocation is the so-called centralized auction, this kind of methodology presents the advantage that the central auctioneer does not need to fully maintain tracking of the other members, just process the objective function obtained from each member. However, in their work, a different approach independent of regular contact with ground was needed since their working environment is deep space missions, and therefore having regular communication with ground control unit is not feasible. Even though, in our case, a centralized auction is an approach that fulfills our needs because it is possible to have regular contact windows in LEO. It is also important to mention that centralized proposals tend to be vulnerable to inter-system failures.

Other centralized approaches are also available, for instance, in robotic scenarios, Jose et al. [16] used a centralized genetic algorithm with greedy initialization for

task allocation in multi-robot systems, but concerning a few amounts of robots, otherwise, the convergence of the solution increases exponentially. Yao et al. [17] proposed a partially centralized approach, taking a group of autonomous satellites and approximating a solution based on a master-slave approach, and designing a Support Vector Machine (SVM) based selector to improve the dynamic response of the system but relying heavily on stable communications between the group of cooperating satellites.

Concerning decentralized proposals, these have a great advantage in the high robustness that can tolerate low-level failures in the system [15], however, many proposals using this methodology, or require that all agents are willing to cooperate to obtain a solution, or cannot provide guarantees of a short term allocation of the system and high success rate. Choi [18] proposed two decentralized auction-based approaches, the consensus-based auction algorithm and consensus-based bundle algorithm for a fleet of autonomous mobile robots, although, the convergence of consensus algorithms could take a relevant amount of time and require the transmission of a large amount of data.

Although our methodology is partially centralized, the bidding formulation that we will develop in the next chapters is based on a decentralized framework as proposed in Walsh et al. [19] and Atkinson [20], who develop a state-of-the-art of task allocation using a market protocol and market auctions, where the auctioneer acts as a central system receiving and evaluating each bid in the fleet.

## 2.3 Multi-Agent Systems

The purpose of this chapter is to explain the framework on which our methodology is based. The definition of an agent, its characteristics, and different methods of cooperation and negotiation will be introduced, as well as different approaches in the literature using this framework in different applications.

A primary notion of multi-agent systems and their main constituents (i.e., Agents) is provided: «Multiagent systems are systems composed of multiple interacting computing elements, known as *agents*» [21]

### 2.3.1 Agents definition

«An *agent* is a computer system that is *situated* in some *environment*, and that is capable of *autonomous* action in this environment in order to meet its design objectives.» [21]

Figure 2.7 gives an abstract view of an agent interacting with its environment. Once it receives a sensory input from it, an output action is produced and sent. The main issue that an agent faces is deciding which of its actions should perform

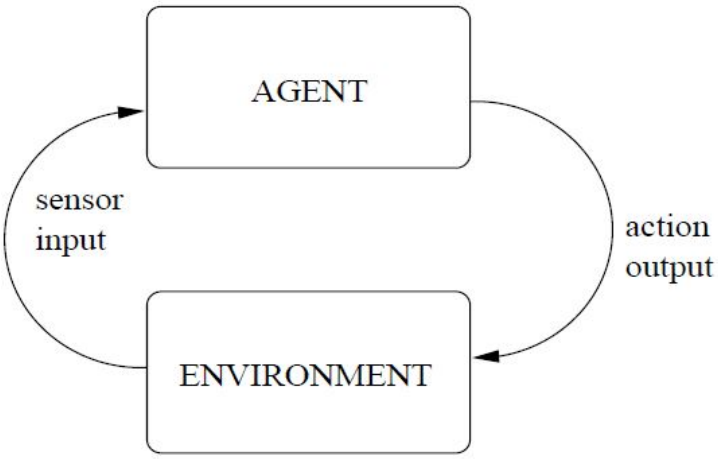



**Figure 2.7:** An agent receives a stimulus from the environment and produces an output action in response [21]

in order to best satisfy its design objectives. [21]

Besides the type of environment in which the agent finds itself, these typically have a set of available actions to execute based on the stimulus coming from this environment. An agent, by definition, has the ability to transition between states in response to different stimuli. Our methodology is not designed based on state transitions; however, as software entities, the introduction of programming logical conditions inevitably leads to logical states through which different entities may pass depending on whether the implemented conditions are met. As a result, the following chapters will provide a review of the various types of agents and their interactions with the environment.

### Intelligent agents

In principle, an intelligent agent has the ability to receive a stimulus from the environment, process this stimulus and autonomously elaborate an action that allows it to satisfy its design goals. An intelligent agent typically possesses the following attributes: [22]

- **Reactivity:** Intelligent agents can perceive their environment and respond to changes in them in order to achieve their design goals.
- **Pro-activeness:** Intelligent agents could exhibit goal-directed behavior by taking the initiative to meet their design objectives.
- **Social ability:** Intelligent agents can interact with other agents in order to achieve their design goals.

An agent is not required to possess all of these characteristics to be considered intelligent. In our research, we created two types of agents with a subset of these properties. The first, the central agent, is reactive in the sense that it must respond to environmental stimuli in order to notify others that an event has been initiated. It also possesses social skills in order to cooperate with the other agents to determine which of them best satisfies a previously established task.

The agents in charge of modeling the satellite state, on the other hand, should have all three characteristics. They require reactivity because they must respond to changes in the satellite's resources after interacting with the environment and thus update their own attributes. Social abilities to interact and collaborate with the central agent and other agents bidding for a specific task. Finally, because these agents are goal-oriented, pro-activeness is required in order to generate an offer that reflects its ability to satisfy a given request.

### Types of Intelligent Agents

It is necessary to describe how diverse agents would convey out the decision-making process that we previously referred to as action. Four different classes of agents could be considered for this:

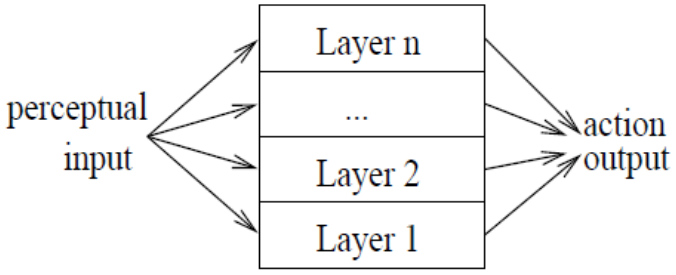
| **Type of agent** | **Decision-making process** |
|---|---|
| Logic-based agents | Logical deduction is employed to make decisions. |
| Reactive agents | Decision-making is carried out through some form of direct mapping from situation to action. |
| Belief-desire-intention agents | Decision making is based on the manipulation of data structures that represent the agent's beliefs, desires, and intentions. |
| *Layered architectures* | Decision making is realized via various software layers, each of which is more-or-less explicitly reasoning about the environment at different levels of abstraction. |

**Table 2.1:** Comparative between different types of agents and their decision-making process [21]

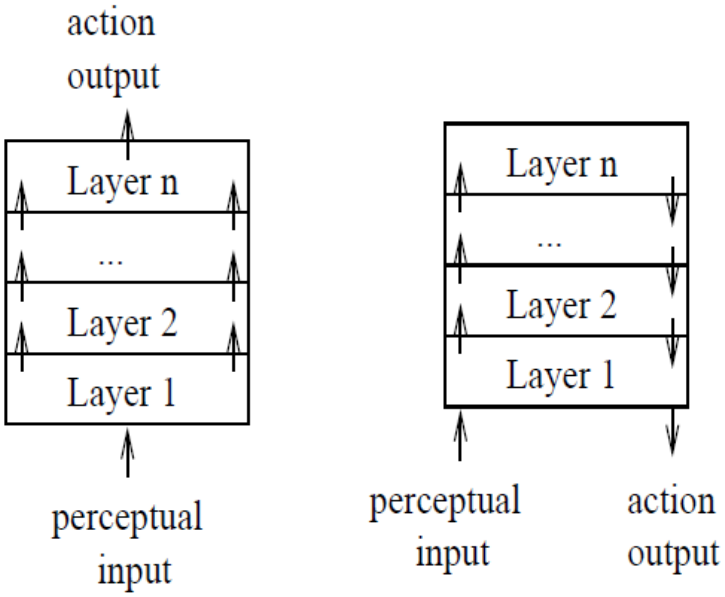
Our primary focus is on agents with **layered architectures**, since it is required software objects that can receive data from the environment and reason about it at various levels of abstraction based on a previously defined formulation. This concept leads to a class of architectures in which the several subsystems are organized into a hierarchy of interacting layers. [21] Two types of control flow can be identified in this type of architecture:

1. **Horizontal Layering:** Each software layer is directly linked to the sensory input and action output. In effect, each layer acts as an agent, generating suggestions for what action to take. (Figure 2.8a)

2. **Vertical Layering:** At most one layer handles both sensory input and action output. (Figure 2.8b)




**(a)** Horizontal Layering




**(b)** Vertical Layering

**Figure 2.8:** Information and control flows in layered agent architecture [22]

Regardless of the fact that the horizontal layering architecture refers to each layer as a different type of agent that processes the input and generates a reasoning output based on it. This is the type of architecture on which our model is based, since having a multi-layered structure would allow the agent to generate a numerical value based on the combination of several reasoning functions that allow it to process an output indicating the willingness to fulfill a given task.

Weiss [22] points out that the horizontal layers compete with each other to generate suggestive actions, posing a risk to the agent's behavior because it will be incoherent. Our architecture employs a weighted combination of layers to generate a single value describing satisfiability. Furthermore, the central agent mediates the interaction of multiple agents, while each of them is modeled to fully cooperate.

### Environment

An environment is the domain in which an agent can perform a set of available actions that could change the state of the environment, described as follows:

$$Ac = \{\alpha, \alpha', ...\} \tag{2.1}$$

An environment may be in any of a finite set $E$ of discrete, instantaneous states [21]:

$$E = \{e, e', ...\} \tag{2.2}$$

Then, an event run consists of an environment of continuous state changes caused by actions performed by the agents present in it, as described below:

$$r : e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} e_2 \xrightarrow{\alpha_2} e_3 \xrightarrow{\alpha_3} \cdots \xrightarrow{\alpha_{u-1}} e_u \tag{2.3}$$

However, whilst the agents could change the state of the environment through its available set of actions, the state of the agents can be altered depending on the changes perceived from the environment.

An environment could have different properties, which are classified as follows:

- *Accessible vs. Inaccessible*: An accessible environment is one in which the agent has complete, accurate, and up-to-date information about the state of the environment. The great majority of moderately complex environments are inaccessible. [22] Our scenario is focused on real-time conditions of outer space in low earth orbit, therefore, our environment is inaccessible as satellites agents cannot have with full certainty and completeness all the information that surrounds it.

- *Deterministic vs. Non-deterministic*: In a deterministic environment, any action has a single guaranteed effect. There is no uncertainty about the state that will be achieved as a result of an action. [22] The outer space

in which satellites interact is a hostile environment and, therefore, a non-deterministic one. Complex formulations are modeled in order to have a minimum abstraction to allow a prototype interact with the multiple factors and variables encountered in this type of environment.

- *Episodic vs. Non-episodic*: An agent in an episodic environment can decide what action to take based purely on the current episode. It does not need to consider the interactions between this and future episodes. [22] Our model considers the evolution of the attributes present in the satellite (e.g., the evolution of the resources or the contact windows in its orbit) during a single auction, and thus this last one could be considered an episode in itself. As a result, satellite models make decisions based on current events during the auction and the evolution of various attributes, as previously mentioned.

- *Static vs. Dynamic*: A static environment is one that can be assumed to remain unchanged except for the agent's actions. A dynamic environment is one that has other processes running on it and thus changes in ways that the agent cannot control. [22] With this definition, it can be deduced that the outer space in which the satellites are immersed is a dynamic environment since several variables influence it beyond the control of the agents.

### 2.3.2 Multi-agent society

#### Multi-agent Environment

When dealing with multi-agent societies, the environment should be designed in such a way that the agents can operate effectively and interact productively among themselves.

Typically, a multi-agent environment has the following characteristics: [23]

1. Multi-agent environments provide an infrastructure specifying communication and interaction protocols.

2. Multi-agent environments are typically open and have no centralized designer.

3. Multi-agent environments contain agents that are autonomous and distributed, and may be self-interested or cooperative

In our case, a multi-layered software architecture enables the use of multiple object-oriented agents, each of which can update its attributes based on the input received and thus process an output in a fully cooperative environment.

### Agents communication

An agent has explicitly represented knowledge as well as a mechanism for acting on or drawing conclusions from that knowledge. It is also assumed that an agent is capable of communicating. This ability is a combination of perception (message reception) and action (the sending of messages). These may be the only perceptual and acting abilities of a purely computer-based agent. [23]

- **Coordination**

  As established in Huhns [23], agents communicate in order to improve their own performance or the society/system in which they exist. It should be noted that depending on whether the agents are goal-based or not, the goals may or may not be explicitly known to them. Communication allows agents to coordinate their actions and behavior, resulting in more coherent systems.

  Coordination is a necessary feature when dealing with a multi-agent society or an environment where multiple interactions between agents take place. This allows to improve the overall performance of the system while at the same time improving the performance of each individual component, more efficient use of available resources, and coherence in the system's behavior.

  There are two ways in which agents can coordinate themselves:

  1. Cooperation: Coordination between non-antagonistic agents (i.e., agents that are not self-interested, do not compete with other agents, and cooperate to achieve a better performance together). Typically, to cooperate successfully each agent maintains a model of the other agents so that future interactions can be modeled.
  2. Negotiation: Coordination between competitive or purely self-interested agents.

  A pictorial representation of the previous concepts of cooperation and negotiation is shown in the Figure 2.9. It can be observed that cooperation can be achieved through planning, which can be centralized or decentralized. As previously discussed, centralized planning requires the presence of a central agent in charge of coordinating the activities of the other agents, whereas in decentralized planning, communication among them is essential because they must coordinate in the best way possible to achieve a common goal. Forcing them not to be self-interested.

  On the other hand, negotiation happens through competition, where agents tend to be antagonistic and self-interested, allowing a greater focus on self-improvement to achieve the best possible task satisfaction.

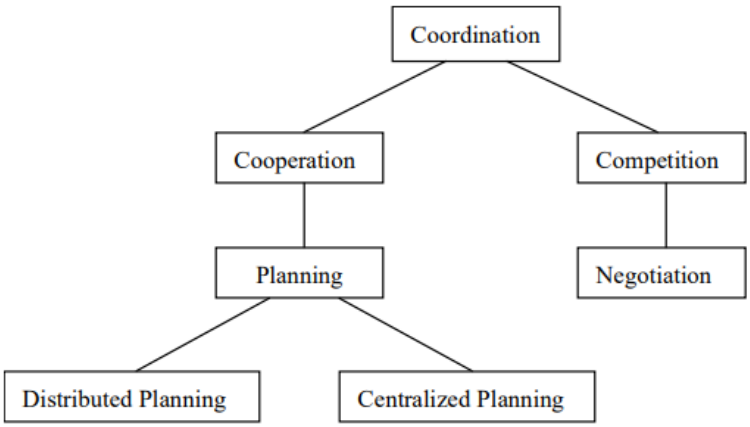



**Figure 2.9:** Different ways in which agents can coordinate their behavior and activities. [23]

- **Message type**

  A communication network allows an agent to send and receive messages. Agents must have the ability to receive information through messages, enabling them to establish a dialogue with other agents. Messages can be of various types, as defined below.

| Communicative Action | Illocutionary Force | Expected Result |
|---|---|---|
| Request | Request | |
| Refusal | Inform | Acceptance |
| Offer/Bid | Inform | Acceptance |
| Proposal | Inform | Acceptance |

**Table 2.2:** Interagent Message types [23]

In table 2.2, the different types of messages that for our model an agent could use to communicate can be seen. It should be noted that depending on the agent's role in society, whether as a central agent or as a bidding agent, the types of messages available will differ. For central agents, it is not necessary to offer any type of bid since their role is to compare bids and accept them and, based on this, inform the awarding agent. Likewise, bidding agents can have basically all types of messages available but they are not expected to elaborate on proposals but only to communicate their acceptance. The following subchapter elaborate further on this type of interactions of Announce-Offer/Bid-Award network, commonly called a contract-net.

- **Agents interaction**

  There are different mechanisms commonly used to distribute tasks: [23]

  1. Market mechanisms: By generalized agreement or mutual selection, tasks are assigned to agents (analogous to pricing commodities)
  2. Multiagent planning: Task assignment is handled by planning agents.
  3. Organizational structure: Agents have distinct responsibilities for specific tasks.
  4. Contract-Net: Announce, bid, and award cycles.

  Our model for task distribution among cooperative agents is based on the *Contract-Net protocol*.

  **Contract-Net protocol:** «The contract net protocol is an interaction protocol for cooperative problem solving among agents. It is modeled on the contracting mechanism used by businesses to govern the exchange of goods and services. The contract net provides a solution for the so-called *connection problem*: finding an appropriate agent to work on a given task.» [23]

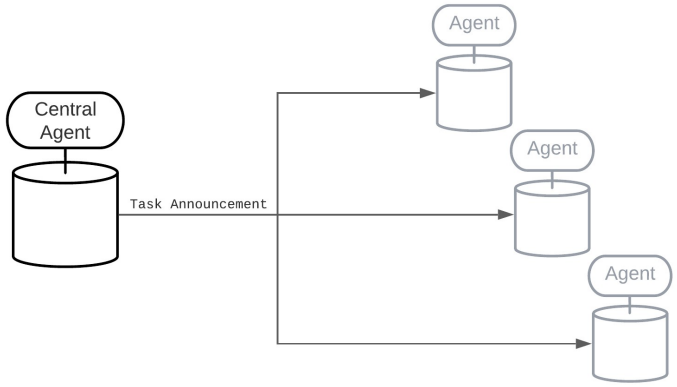
  A manager is an agent who wants a task solved; similarly to our model, this agent is called "Auctioneer". From this central agent perspective, the process is the following:

  1. Announce a task that needs to be performed.
  2. Receive and evaluate the various bids from potential available agents.
  3. Award an agent who might bid the highest.
  4. Receive satisfaction of the task coming from the award agent.

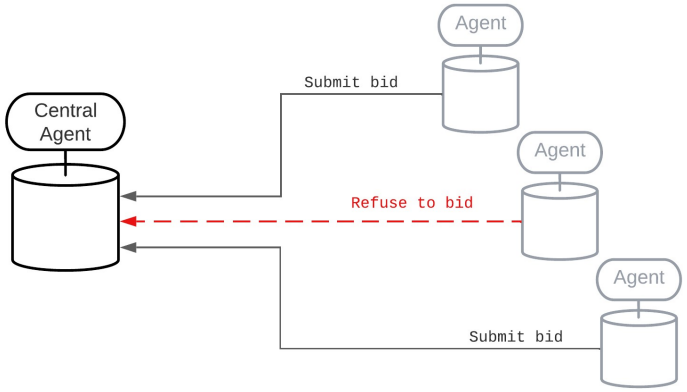
  On the other side, the process from a contractor's perspective; similarly, in our model known as "Proxy" is:

  1. Receive the task announcement.
  2. Reasoning and evaluate my capability to satisfy the request.
  3. Generate a bid based on my willingness to satisfaction.
  4. Respond or refuse to bid.
  5. Perform the task if my bid is accepted.
  6. Give feedback to the central agent about the results obtained.

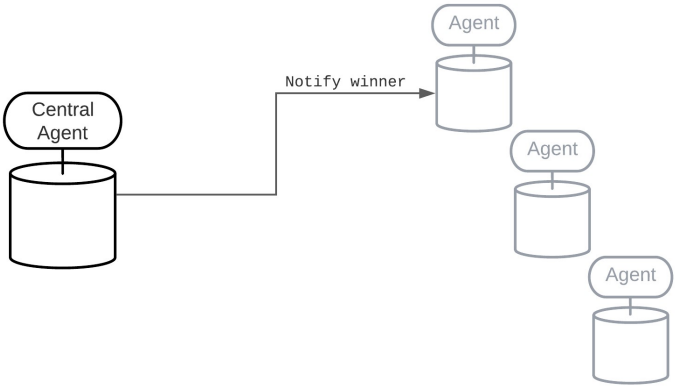
  A pictorial view of the above-described process can be seen in Figure 2.10.




**(a)** Announcement of the existence of a pending task




**(b)** Submission of the corresponding bid (or refusal)




**(c)** Notification to the awarded agent

**Figure 2.10:** Basic steps in the distribution of a task in a Contract-Net

### Auctions

«An auction consists of an auctioneer and potential bidders. Auctions are usually discussed in situations where the auctioneer wants to sell an item and get the highest possible payment for it while the bidders want to acquire the item at the lowest possible price.» [23]

In the same way, when using multiple auctions for the distribution of tasks to different agents, it could be assumed with this previous concept that the bidding agents try to obtain an announced request with the lowest possible bid. But, on the contrary, the modeling done has the principle of forcing the bidding agents to cooperate with each other in order to leave the best qualified satellite, while at the same time having each of them bid a value that reflects as close to reality as possible their ability to satisfy it.

In contrast to voting, where the outcome binds all agents, the outcome of an auction is usually a deal between two agents: the auctioneer and one bidder. Furthermore, in voting, the protocol designer is assumed to want to improve the social good, whereas in auctions, the auctioneer is assumed to want to maximize his own profit. [23] This concept takes relevance in our model since the central agent (i.e., Auctioneer) is modeled in order to award tasks to the bidders with the highest offer, thus configuring an auction setting described as follows.

- **Auction settings**

  There are various auction settings available depending on how an agent's value of the item is determined. [22]

  1. Private-value auctions: The value of the good depends only on the agent's own preferences.
  2. Common-value auctions: An agent's value of an item depends entirely on other agents' values of it, which are identical to the agent's by symmetry of this criterion.
  3. *Correlated-value auctions*: An agent's value depends partly on its own preferences and partly on others' values. A negotiation within a contracting setting fulfills this criterion. An agent may handle a task itself in which case the agent's local concerns define the cost of handling the task. [21]

  This last auction configuration is part of our task distribution process, with the difference that the bidding agents do not know the bids of the other ones and once it is assigned, they cannot renegotiate with other agents, since this functionality is performed by the central agent. With the auction configuration, the protocol by which an auction will be carried out can be defined, as described below.

- **Auction protocols**

  There are different protocols by which an auction can be conducted, depending on how the items are offered and the capabilities of the bidders.

  Each bidder in the *English* auction (first-price open-cry) is free to raise his bid. The auction ends when no bidder is willing to raise his bid any further, and the highest bidder wins the item at the price he bid. An agent's strategy consists of a series of bids based on his private value, prior estimates of other bidders' valuations, and past bids of others. [22]

  In the *first-price sealed-bid* auction, each bidder submits one bid without knowing the others' bids. The highest bidder wins the item and pays the amount of his bid. [22]

  Our auction model is based on first-price sealed-bid because the agents do not know what the others are bidding, and the one who manages to bid more wins (e.g., those satellite models that are able to bid the most indicate that they have a better willingness to satisfy a task).

Several instances of auction applications in different scenarios can be found in the literature. Sandholm [24] developed a bidding/auction decision process based on the Contract-Net interaction method. This enabled him to perform a classification of different tasks and the coordination of their allocation in a multi-agent environment, as well as to develop marginal cost functions on which agents can rely on a criterion.

Huberman [25] further developed a solution for the distribution of thermal resources in a building with a market-based system. His work presents a computerized auction in which double-blind agents participate by buying thermal units proportional to the amount of bid they offered. A central computer moderates the entire auction, acting as the central agent or auctioneer and enabling an equitable temperature distribution while imposing minimal costs on the actuators.

# Chapter 3

# Problem definition

## 3.1 Scenario description

The availability of advanced automation onboard satellites enables a significant change in paradigm in operating large and complex constellations. This project explores feasible solutions to the dynamic constellation tasking problem with the aim to fully exploit onboard re-planning capabilities to dramatically simplify and speed-up on-ground planning. This is done using multi-agent-based architectures and cooperative auctions. [26]

The ground system collects service requests coming from clients. When a satellite comes into contact with a ground station, a subset of the pending requests is assigned to it, depending on the satellite's capabilities and availability. No detailed planning is performed on the ground, but each satellite schedules its own operations.

Further research can extend the framework to allow reassignment of unsatisfied requests to other satellites, generation of new requests directly onboard the satellites, as well as satellite-to-satellite communication and request (re)assignments.

### 3.1.1 Scenario components

An essential part of the problem description is the definition of the different components that comprise it. This would help to understand the different factors that are involved in the task distribution mechanism in a satellite constellation. It considers the structure from the ground control unit to the satellite's operation modes and capabilities.

- Proxy: A formal definition can be described as an intermediate mechanism that provides a connection between two parts. In computer science, a proxy server is an application that acts as an intermediary between a client requesting a

resource and the server. Likewise, in our context, the 'Proxy' is the on-ground satellite model. It acts on behalf of the real satellite.

- Auctioneer: This is the label given to the central agent, who coordinates the auction process and therefore the bidding agents. In principle, it is in charge of announcing the items up for auction, collecting the different bids, determining the winner of the auction, and notifying the awarded proxy of the tasks assigned to them. It essentially coordinates the flow of information and the organization of the auction.

  Figure 3.1 shows a representative view of the interface between proxies and auctioneer in a satellite constellation, as well as the interaction of the ground tasking unit when a request arrives from a client. It also depicts the contact with the real satellites that triggers the tasking process.

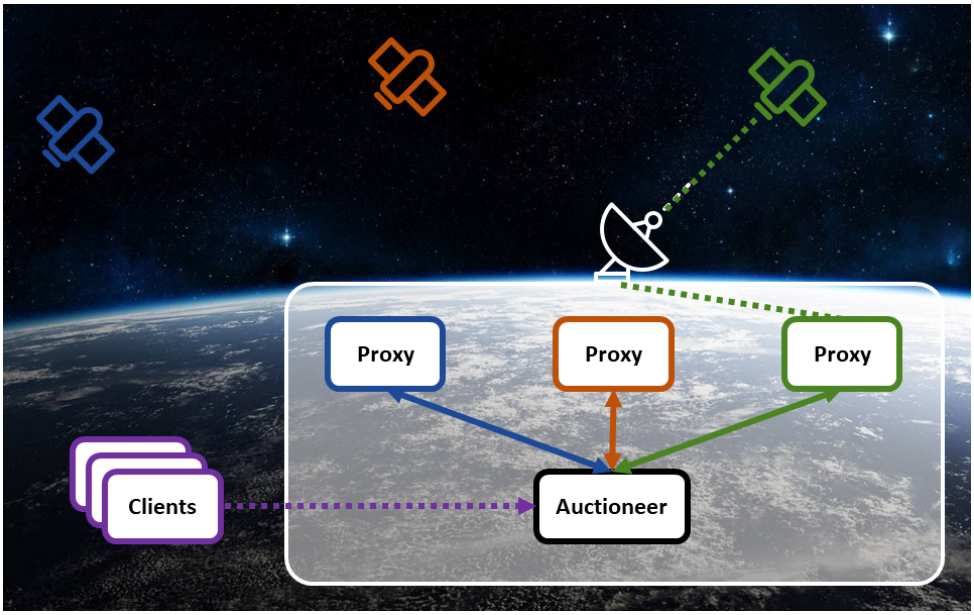



**Figure 3.1:** Pictorial view of the considered scenario and proposed system architecture.

In Figure 3.2, a pictorial depiction of the roles of the Auctioneer and the Proxies within the ground tasking unit can be seen. The Auctioneer should be able to communicate with proxies that represent satellites that are not in contact with ground (the black ones), as well as communicate with proxies that represent satellites that are in contact with ground (the colored ones), and update their different attributes with the incoming information.

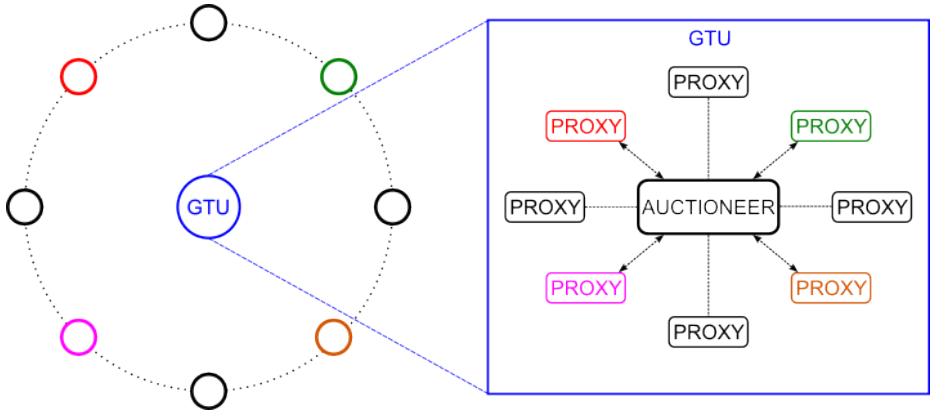



**Figure 3.2:** Pictorial representation of an Auctioneer-Proxies system in the ground tasking unit of a constellation of satellites

- Services: These are defined as the different activities that a satellite can conduct depending on its own characteristics. Naturally, satellites can be classified depending on the on-board services they can deliver. For instance, there are different types of services that can be offered to the client, such as imaging in the colour of infrared spectrum, acquisitions using SAR payloads or many others. Additionally, there are specific ones related to satellite maintenance and contact with ground control unit, such as solar recharging and antenna downlink services. A pictographic representation of service types as images can be seen in Figure 3.3. A request for image acquisition over a specific region is shown on the left, and the processed result for the client is shown on the right.

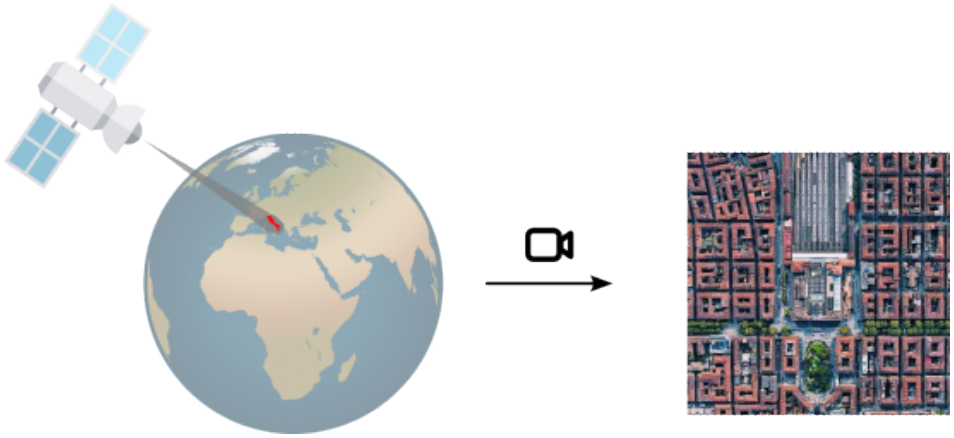


**Figure 3.3:** Pictorial representation of satellite image acquisition

- Payloads: This term is defined by the European Space Agency (ESA) as those elements of the spacecraft that are specifically dedicated to producing mission data and relaying that data back to Earth. Consequently, for each of the services mentioned before, a number of payloads are required to execute them. For instance, recalling Figure 3.3 of image acquisition, to perform such a service, a satellite with an optical payload (i.e., High/Low resolution camera) is required. In addition, the antenna payload is required to proceed with the data downlink. Similarly, on-board resources are required for the proper execution of the service; thus, solar panels to recharge the batteries are required.

### 3.1.2 Scenario constraints

The introduction of different constraints present in the scenario is required, and therefore, they should be considered when designing the task distribution architecture.

- Visibility windows: This type of constraint stems from the orbital path of the satellite. It occurs because the downlink antennas on ground have a limited range of coverage, creating gaps during the orbit where it is not possible to communicate with the satellite. This type of condition is a major constraint for the system to be modeled, since it must be considered when designing the architecture of task distribution in the constellation. A representative view can be seen in Figure 1, where dark areas can be observed in which satellite communication with the ground station is not possible; likewise, there are already established ranges during the orbit in which the satellite comes into contact with the ground stations.

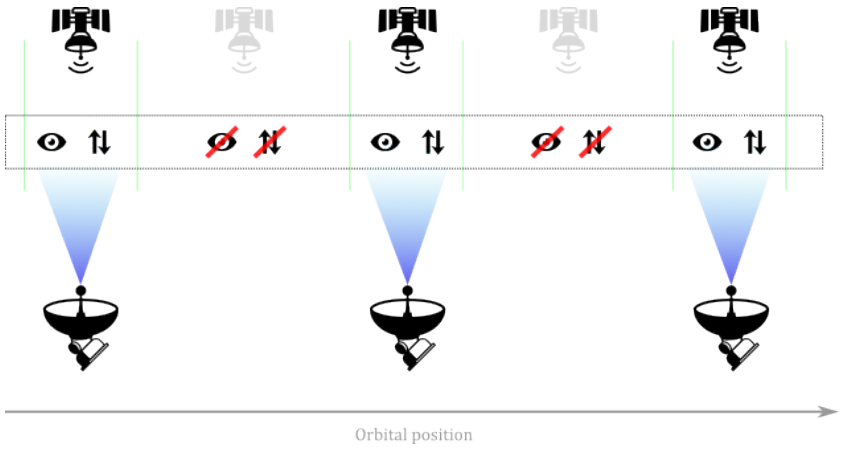



**Figure 3.4:** Pictorial representation of visibility windows

- Recharge windows: Solar recharge windows are those intervals during the orbit in which the satellite can receive solar energy, allowing it to recharge the batteries onboard the spacecraft. It can be carried out concurrently with the execution of other tasks, as it does not conflict with them. The introduction of this attribute and its updating in the architecture are necessary because it directly affects satellite's resources, its performance, and its correct operation.

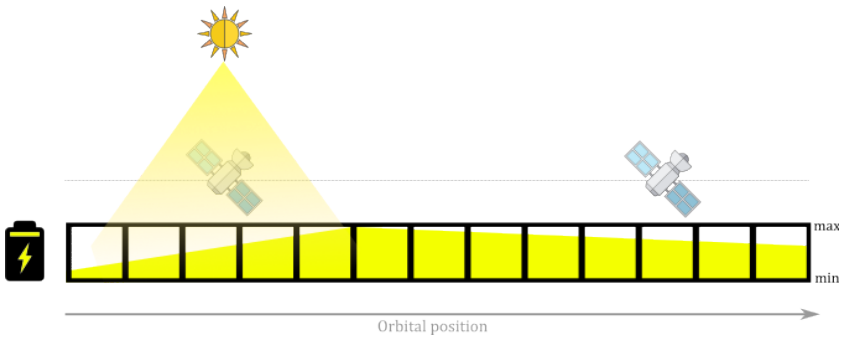



**Figure 3.5:** Pictorial representation of recharge windows

## 3.2 Problem description

The following Section provides a detailed definition of the problem.

### 3.2.1 Definition of the system components

Let be $\mathcal{CS} = (S, G, K, C)$ the *Constellation System* composed of:

- a set of satellites $S = \{s_i\}$
- a set of ground stations $G = \{g_i\}$
- a set of offered services $K = \{k_i\}$.
- a *Contact Plan* $C$, being the set of contact windows $C = \{c_i\}$ over an horizon $t_C$ computed from the satellites' orbits and the location of ground stations. Each contact window $c = (t_s, t_e, s, g)$, $c \in C$, describes a time interval starting at $t_s$ and concluding at $t_e < t_C$ during which satellite $s$ exchanges data via ground station $g$.

Let also be $\mathcal{GU} = (a, P)$ the *Ground Tasking Unit* composed of:

- an *Auctioneer* agent $a$

- a set of *Proxies* agents $P = \{p_i\}$ such that $|P| = |S|$ and $p_i$ is a Proxy for $s_i, \forall i = 1, \ldots, |S|$

The definition of Auctioneer and Proxy agents is given in Section 3.2.2.

The Ground Tasking Unit receives client requests and satellite status updates as inputs. The Clients' requests can be received at any time and are defined as a tuple $r = (ID_{req}, ID_{cli}, k, ROI, t^{sub}, t^{exp}, \alpha)$ where:

- $ID_{req}$ is a unique request ID
- $ID_{cli}$ is the client ID

  These attributes are part of the request's own nomenclature, from the implementation point of view their datatype can be a numeric integer or a string.

- $k \in K$ is the requested service: This attribute contains relevant information about the request and the type of service requested; it is a vector that can vary between different services such as optical, spot, strip and combined acquisition.
- $ROI$ is the geographic area of interest – whose exact definition depends on the service type
- $t^{sub}$ is the submission date
- $t^{exp}$ is the expiring date – i.e., the time limit to satisfy the request

  These attributes are normally expressed in Julian Date (JD) format, which is a numerical datatype format with a range of $[0, \infty]$, that allows for the quantification of time differences as well as the numerical representation of a given date. They are responsible for informing the date on which the request was submitted or will expire.

- $\alpha \in [0, 1]$ is the priority level: This attribute numerically quantifies the client's priority for the request. This present a numerical datatype whose maximum value '1' represents the highest priority.

A client request is said to be satisfied after a satellite in the constellation has successfully executed the requested service and the corresponding data have been successfully transmitted to the ground and back to the client. In this work, the transmission time from the Ground Unit to the Client is assumed to be negligible.

Satellite status updates are received according to the Contact Plan. Specifically, satellite $s^*$ communicates its status $\sigma_{s^*,t}$ at the beginning of each contact – i.e., $c \in C : s = s^* \Rightarrow \sigma_{s^*,t_s}$. Satellite status updates are defined as a tuple with the following elements:

- arrival time
- satellite telemetry
- pending assigned requests

These data are used to update the status of the associated Proxy , whose definition is given in the following. The Ground Tasking Unit answers to the reception of each satellite status update with a set $R^*$ of requests assigned to the communicating satellite.

### 3.2.2 Definition of agents within the Ground Tasking Unit

The Auctioneer is a tuple $a = (R, P)$ where:

- $R = \{r_i\}$ is the set of unassigned client requests
- $P = \{p_j\}$ is the set of proxies with which it communicates

Each satellite Proxy is a tuple $p_j = (s_j, R_j, L_j, K_j, \Pi_j, \mathcal{K}_j, \Delta_j) \in P$ composed of:

- the satellite $s_j \in S$ which the agent is Proxy of
- a set $R_j$ of already assigned pending requests
- a set of resource level arrays $L_j = \left\{ l_j^\rho = \left[ l_{j,t_0}^\rho, l_{j,t_1}^\rho, \ldots, l_{j,t_N}^\rho \right] \right\}$ where each $l_j^\rho$ indicates the predicted resources levels for the satellite's resource $\rho$ at discrete future time instants $t_0, t_1, \ldots, t_N$. More detail on the modelling of on-board resources is provided in section 4.3.
- a set $K_j \subseteq K$ of services that satellite $s_p$ is currently able to perform
- a set $\Pi_j$ of payloads available onboard the satellite to provide services
- a function $\mathcal{K}_j : K_j \to \mathcal{P}(\Pi_j)$ that maps each service $k \in K_j$ to the subset of payloads required to perform the service
- a function $\Delta_j : K_j \to \mathbb{R}$ that maps each service $k \in K_j$ to the amount of time required to perform the service

### 3.2.3 Client-System interaction

The Client-System interaction is described by the sequence diagrams in Figure 3.6. Clients' requests can arrive at any time and are received by the Auctioneer, which places them into the list of pending unassigned requests. When a request is fulfilled, the Auctioneer responds to the appropriate Client with the acquired data. Otherwise, if the request is still unsatisfied upon expiration of the request deadline, the Auctioneer sends a notification to the appropriate Client.

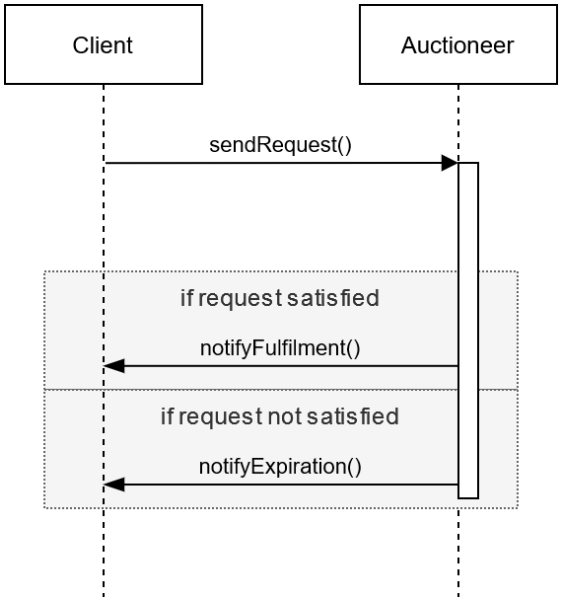



**Figure 3.6:** Collection of new requests coming from clients

### 3.2.4 Satellite-System interaction

The Satellite-System interaction is described by the sequence diagrams in Figure 3.7. Satellite-system interaction is triggered whenever a satellite $s_i \in S$ enters a visibility window and communicates with the ground station. Satellite status updates are received by the Ground Tasking Unit, which starts the task assignment process that proceeds according to the following steps:

1. **Proxies update**. Each $p \in P$ is updated; Proxy $p_i$ corresponding to the visible satellite $s_i$ is updated from real data contained in the received status message, while the remaining Proxies are updated through simulation according to their expected evolution.

2. **Announcement**. The Auctioneer broadcasts a message to all the Proxies containing a set $R_a^k \in R$ of pending requests that must be assigned during the current (k-th) auction round.

3. **Bid generation**. [27] Each Proxy $p_j$ generates a bid $b_{j_i}$ for each request $r_i$ to identify its capability to satisfy $r_i$.

4. **Bid submission**. Each Proxy sends its bids to the Auctioneer.

5. **Requests assignment**. The Auctioneer processes the bids and assigns a subset $\hat{R}^k \in R_a^k$ of requests to the winning Proxies.

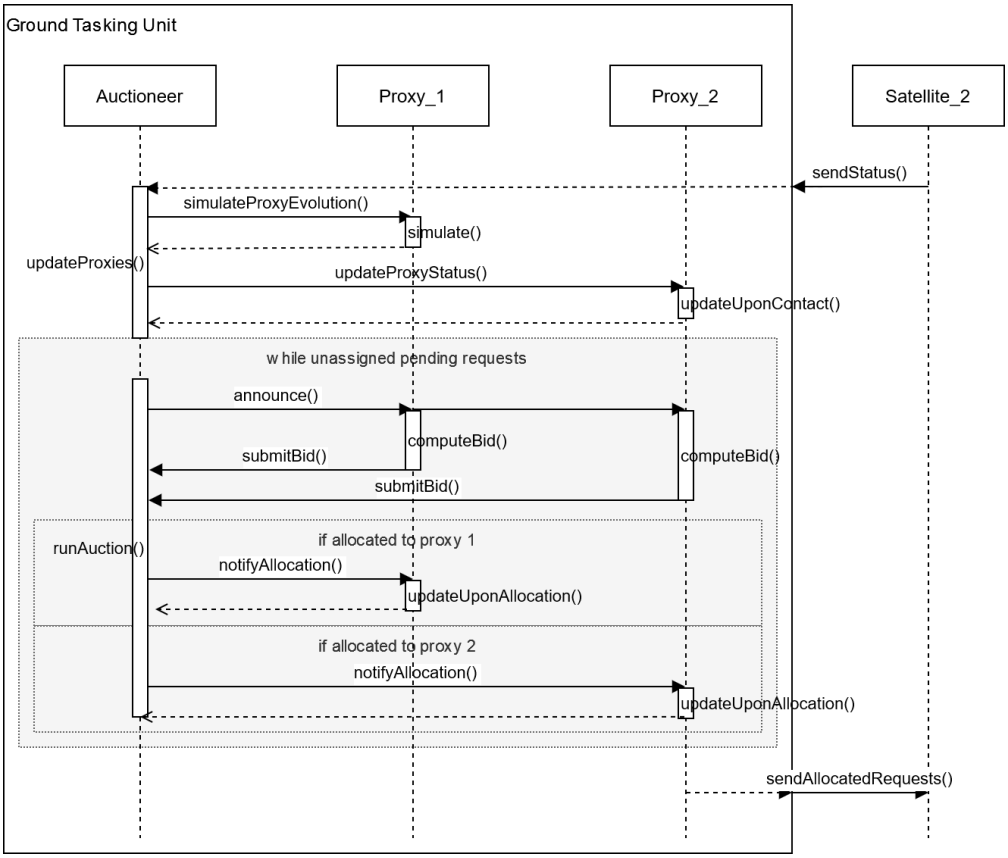



**Figure 3.7:** Auction process triggered by a new satellite contact

6. **Back to 2**. If there are still unassigned pending requests, a new auction round starts.

7. **Requests transmission**. Once the assignment process has been completed, a message is sent to satellite $s_i$ before the end of the visibility window, containing the new requests assigned to it during the auction.

In the following, detail about relevant steps is given.

### Announcement

At the beginning of the k-th auction round [28], the Auctioneer announces to all the Proxies a set of pending requests $R_a^k \subseteq R, |R_a^k| = m$, where $m$ is the announcement size. The Auctioneer can select the requests to put in the announcement according to several strategies – such as simple FIFO, ordered by priority, by expiring date, or more complex metrics. The announcement size also impacts the system behavior.

### Bid generation

Upon reception of announcement $R_a^k = \{r_1, r_2, \ldots, r_m\}$, each Proxy $p_j$ generates a bid $b_{j,i} \in [0,1]\ \forall r_i \in R_a^k$. The bid expresses the capability of Proxy $p_j$ to satisfy the request $r_i$ timely and efficiently. Bids are computed independently for each request in the announcement.

Let $t_s^{min} = \min_{c \in C}\{t_s : s = s_j\}$ be the start time of the earliest contact window for Proxy $p_j$ and let $t_s^{min} < t_{ROI_i}^j < t_i^{exp}$ be the time to target– i.e., the minimum time required by $s_j$ to reach $ROI_i$ of request $r_i$, provided that it lies between the first time to contact and the expiring date of the request. This is obtained by propagating the orbit of $s_i$ with dedicated tools. Then, each bid is evaluated as follows: Specifically, each bid is evaluated as follows:

$$b_{j,i} = \begin{cases} 0 & \text{if } k_i \notin K_j \wedge \nexists t_{ROI_i}^j \\ \sum_{n=1}^{N_b} w_{j,n}\gamma_n & \text{otherwise} \end{cases} \tag{3.1}$$

That is: the Proxy bids 0 if it is unable to provide the requested service, otherwise it bids a convex combination [29] of several bidding terms $\gamma_n \in [0,1]$ that account for a specific aspects to be included in the overall bid. The tuple $W_j = (w_{j,1}, w_{j,2}, \ldots, w_{j,N_b})$ is called the bidding strategy of Proxy $p_j$, with $w_{j,n} \geq 0,\ n = 1, \ldots, N_b$ and $\sum_{n=1}^{N_b} w_{j,n} = 1$ as per definition of convex combination.

A description of the bidding terms considered for this problem is provided in the following:

- **Pending requests ($\gamma_q$).** The role of this term is to balance the load among the satellites in the constellation. The bid decreases as the number of already assigned requests increases, specifically:

$$\gamma_q = \operatorname{sech}\left(\zeta_q |R_j|\right) \tag{3.2}$$

  where, $\zeta_q \in (0, \infty)$ is a non-dimensional scaling factor to improve the sensitivity of the hyperbolic secant function.

- **Request priority ($\gamma_p$).** The role of this term is to favor the assignment of high-priority requests so that they are assigned before lower-priority ones. This term accounts only for the characteristics of the request, namely its priority and expiring date, while being independent of the Proxy status. As such, the evaluation is equal for all Proxies. The bid increases with the priority level and the approach of the expiring date. Specifically:

$$\gamma_p = \alpha_i \cdot \operatorname{sech}\left(\zeta_p (t_i^{exp} - t_0)\right) \tag{3.3}$$

  where, $\zeta_p \in (0, \infty)$ is a non-dimensional scaling factor to improve the sensitivity of the hyperbolic secant function and $t_0$ is the time at which the auction is run.

- **Request satisfaction time ($\gamma_s$).** The role of this term is to favor the assignment of requests to the satellites that can satisfy them the earliest possible. The bid is higher the shorter the time required for the satellite to execute the requested service and transmit back the data. This is evaluated by considering the best-case scenario, i.e., the first available opportunity. Specifically:

$$\gamma_s = \operatorname{sech}\left(\zeta_c \left[(t_s^* - t_0) - t_s^{min}\right]\right) \tag{3.4}$$

  where,

  - $\zeta_c \in (0, \infty)$ is a non-dimensional scaling factor to improve the sensitivity of the hyperbolic secant function and improve sensitivity
  - $t_s^{min} = \min_{c \in C} \{t_s : s = s_j\}$ is the start time of the earliest contact window for Proxy $p_j$. This is used to improve the sensitivity of the hyperbolic secant function by de-biasing the input
  - $t^j_{ROI_i}$ is the time to target – i.e., the time required by $s_j$ to reach $ROI_i$ of request $r_i$. This is obtained by propagating the orbit of $s_i$ with dedicated tools.
  - $t_s^* = \min_{c \in C} \{t_s : s = s_j \wedge t_s > t_{ROI_i}\}$ is the start time of the earliest contact window for Proxy $p_j$ after the time to target

- **Satellite availability ($\gamma_a$).** The role of this term is to account for possible conflicts of $r_i$ with any $r \in R_j$ or with any $c \in C$. A request is in conflict with a ground contact if they overlap in time. For the purpose of evaluating the bid, a conflicting contact can be seen as a highest-priority task that cannot be delayed. Two requests are in conflict if they ask for the same payload at the same time. When conflicting requests are assigned to the same satellite, it results in a delay in the satisfaction of all but one of the conflicting requests. Therefore, the higher the priority of the requests likely to be delayed by the addition of $r_i$, the lower the bid. Specifically:

$$\gamma_a = \begin{cases} 1 & \text{if } \tilde{R}_{j,i} = \emptyset \wedge \nexists \tilde{c} \\ \dfrac{\prod\limits_{r_n \in \tilde{R}_{j,i} \cup r_i} (1-\alpha_n)}{1 - \max\limits_{r_n \in \tilde{R}_{j,i} \cup r_i} \{\alpha_n\}} & \text{if } \tilde{R}_{j,i} \neq \emptyset \wedge \nexists \tilde{c} \\ \prod\limits_{r_n \in \tilde{R}_{j,i} \cup r_i} (1 - \alpha_n) & \text{otherwise} \end{cases} \tag{3.5}$$

  where $\tilde{c} \in C : [t_{ROI_i}, t_{ROI_i} + \Delta_j(k_i)] \cap [t_s, t_e] \neq \emptyset$ is a ground contact conflicting with $r_i$ where $\tilde{R}_{j,i} \subseteq R_j$ is the set of pending client requests already assigned

to $p_j$ that are in conflict with $r_i$, which is computed as:

$$\begin{aligned}\tilde{R}_{j,i} = \{r_n : r_n \in R_j \wedge \\ \mathcal{K}_j(k_i) \cap \mathcal{K}_j(k_n) \neq \emptyset \wedge \\ [t_{ROI_i}, t_{ROI_i} + \Delta_j(k_i)] \cap [t_{ROI_n}, t_{ROI_n} + \Delta_j(k_n)] \neq \emptyset\}\end{aligned}$$

- **Satellite resources ($\gamma_l$).** The role of this term is to prevent resource over-consumption in the satellite in order to improve the likelihood of being able to perform the assigned services when expected. The bid is lower the lower the minimum level reached by the current bottleneck resource, specifically:

$$\gamma_l = \min_{l_j^\rho \in L_j,\ t=t_{ROI_i},\ldots,t_N} \left\{ \frac{l_{j,t}^\rho}{\hat{l}_j^\rho} \right\} \tag{3.6}$$

  being $\hat{l}_j^\rho$ the upper bound of the resource level for resource $\rho$ and the time to target $t_{ROI_i}$ is considered since $r_i$ impacts the resource level only after its execution.

**Request assignment:**

Once the Auctioneer has collected all the bids from the different Proxies that have submitted them, it proceeds to evaluate them to select the winners of the k-th auction round. To perform this process, a hierarchical selection is made as described below:

1. Having all Proxies $[p_1, \ldots, p_j] \in P$ that generates a bid $b_{a,i}^j$ for each request from the set of announced ones $r_{a,i} \in R_a^k$. Each awarded Proxy per announced request $P_{j,i}^w$ are those that submit the highest bids such that $b_i^w = \max_{r=r_1,\ldots,r_i}\{b_1^i, \ldots, b_j^i\}$
2. Once it is established the set of awarded proxies $P_{j,i}^w$ per each announced requests, the Auctioneer takes the assignment size h and select from the set of announced requests $R_a^k$ the top $h$-th with the highest bids (i.e., ranked by bid size).

Upon each Proxy has allocated their corresponding assigned requests, each of them proceeds to perform an update of its attributes and resources due to the expected consumption required by the insertion of a new request before starting the new auction round. This allows to have updated information for further new rounds of auctions. Further details of this update and tracking process are described in section 4.3.

# Chapter 4

# Architecture

This chapter aims to describe in-depth all the software architecture carried out to create a framework that allows the experimentation of different scenarios in the context of dynamic tasking in a constellation of satellites.

This architecture for dynamic task distribution in a multi-agent system relies on a high-level multi-layered object-oriented programming. This architecture allows each agent to be modeled as a software object with different attributes that can execute several methods.

Figure 4.1 depicts one of the fundamental interactions between the auctioneer and the proxies. It should be noted that this is not the only possible interaction, as discussed in the subsequent sub-chapters. A message announced by the auctioneer and a proposal made by the proxies in response to the announcement can be seen; this type of communication has its own software-based structure.

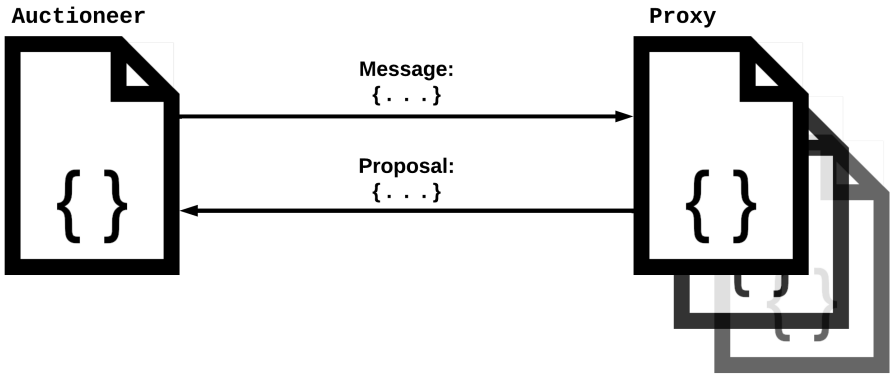



**Figure 4.1:** General overview of the basic interaction between auctioneer and proxy objects

## 4.1 Requests/Message Structure

An essential part of the interactions between agents are the messages, their format, and their structure. In our model, the requests constitute the basis of the messages and are the means of communication between Auctioneer and Proxies, since they are the minimum piece of information to be exchanged.

The requests are contained into a data structure that enables the collection and access of elements of various data types as a hash map structure. As a result, the request structure is coded as a dictionary with multiple fields, as illustrated in Figure 4.2. As stated in the problem definition, all of the enlisted fields contain the relevant information that the proxies require to process and compute their respective bids. This structure also enables the auctioneer to communicate with them and describe the clients' needs.

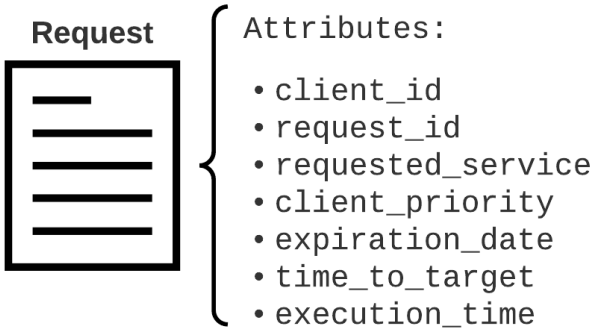



**Figure 4.2:** Different attributes of a request

The auctioneer and proxy classes contain different public and private methods that allow processing the information, reasoning based on it, and generating an output. These methods have access to the information contained in a request for the fulfillment of a specific role within the class hierarchy organization. A further description of the structure of these classes and their methods is described in more detail below.

## 4.2 Auctioneer class

The Auctioneer plays the role of central agent and coordinator among the bidding agents. In an auction, it is in charge of announcing the requests up for auction and collecting the different bids proposed by the different Proxies, updating each of them with the incoming information when a satellite comes into contact with a ground station (recalling the sequence diagram in Figure 3.7), and communicating with the award proxy of the auction.

To generate such an architecture that allows the agent to possess the capabilities mentioned before, a `class` file is created using Python programming language that enables the initialization of the agent with its various attributes as well as several software public and private methods that will allow the Auctioneer to coordinate different proxies throughout the auction.

**Class structure flow process**

1. **Objects and self attributes initialization:** The Auctioneer class initialize its multiple attributes (e.g., list of proxies objects, list of incoming requests, assignment size, announcement size, when the auction starts, and when the auction ends) that allow it to contain relevant information to update the status of the proxies and to store the auction's own data.

2. **Auction mode configuration:** Dedicated methods have been developed to configure the announcement size and the assignment size, whose values are previously fixed in the simulation routine to be executed. These allow to configure the mode in which the auction will be conducted; the default setting is a single-item auction.

3. **Proxies update:** During initialization, the auctioneer updates its attributes with a list of proxies available for bidding, enabling the inter-object interaction; a graphical representation is shown in Figure 4.3. Different methods have been implemented to update the information contained in the proxies' attributes, i.e., the size of the announcements and assignment has to be transmitted to the proxies communicating the modality of the auction to be conducted.

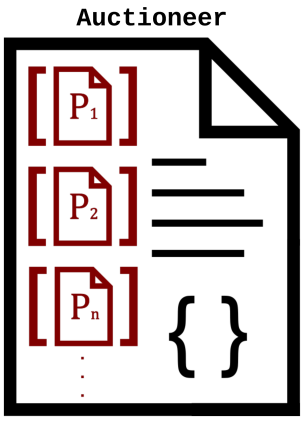



**Figure 4.3:** Pictorial representation of Auctioneer class linked to Proxies class objects through its attributes, allowing it to communicate with them

Another update modality has been implemented in which the Auctioneer can trigger the Proxies' random update class method, allowing random available services, payloads, and consumption maps to be passed within the attributes of each available Proxy. It has been programmed to produce reliable and unbiased results. Another purpose is to investigate the effect of a fixed parameter on the entire auction while the others are randomized.

Proxies' reset class methods triggered by the Auctioneer are also developed when multiple simulation routines are to be performed. It is necessary to reset the attributes and temporal variables of the Proxies before starting a new simulation in order to avoid mismatching information from different simulations and compromising data reliability.

4. **Auction triggering and pending request assignment process:** The definition of these methods allows to trigger the start of the auction. As shown in Figure 4.4, it has a conditional statement to continue running the auction as long as there are pending requests on the Auctioneer's list to be assigned.

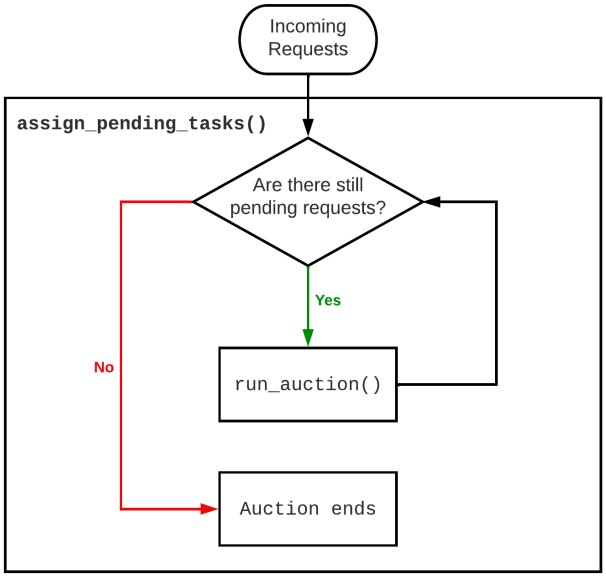



**Figure 4.4:** Flowchart of the internal process for the triggering of the auction

By triggering the request assignment method, it calls the corresponding methods for the creation of the announcement, the collection of the different bids from the available proxies, their evaluation, and the notification of the winner of the auction; a graphical representation of this process is depicted in Figure 4.5. A detailed description of each of these steps is described below.

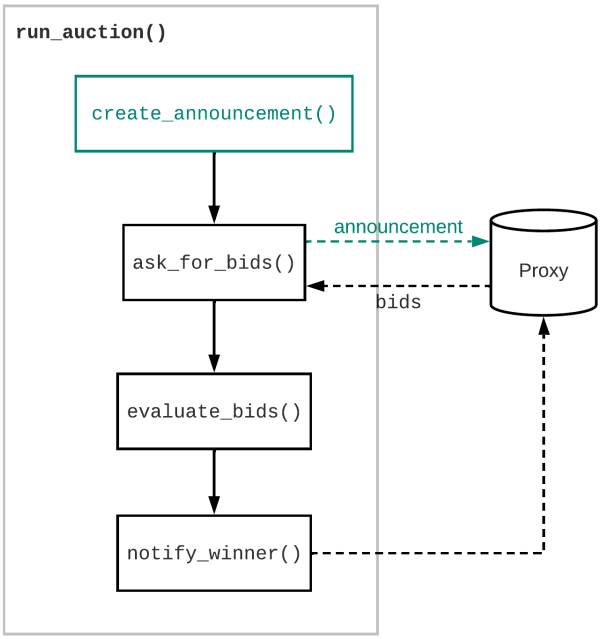



**Figure 4.5:** Flowchart of the internal process of the request assignment method

5. **Announcement creation and bid processing:** A dedicated class method creates the announcements to be sent to the proxies for evaluation. This method has to consider the announcement size previously set and generate a list of requests to be processed by the proxies.

   After the announcement has been successfully sent to the available proxies and they have already computed their respective bids, then the auctioneer operates as a collector. It goes through the entire list of available proxies, asking them to compute their bid for the given announcement, and collects the bid that each proxy offers for it. It returns an array with all the bids collected corresponding to each proxy and each announced request.

6. **Bid evaluation process:** The evaluation process has been designed to determine which proxy offered the highest bid, as well as take the assignment size value into consideration and select the best proxies from among this group of highest bidders as many as the assignment size dictates. This process is depicted graphically in Figure 4.6.

7. **Auction winners' notification:** Once the best proxies have been selected, a method is activated to notify each of them with the corresponding assigned request and the information contained in it. This also keeps track of those requests that were not assigned to any proxy.


**Figure 4.6:** Bid evaluation process with respect to announced requests and assignment size

In this process, it is possible that no proxy is available to bid on a request, in which case this is in charge of separating these unassigned requests into a group of unfeasible requests, in order to avoid mixing them with those that were not assigned due to the assignment size and may return to the Auctioneer's pending requests list.

Algorithm 1 assembles a pseudo-code of the Auctioneer class's structure, recalling all the class methods processes described above, as well as their interactions with the class attributes. The first part of the class definition comprises the initialization of the Auctioneer class attributes. The second part is composed of the definition of the class methods that can be invoked externally to the class definition. And finally, the definition of the class methods that are requested for the management of the auction.

**Algorithm 1** Auctioneer class definition

```
1: procedure CLASS: AUCTIONEER(P, R, A_s, A_h, assign_size, announce_size)
2:     ▷ P is the list of proxies participating in the auction
3:     ▷ R is the list of incoming requests from clients
4:     ▷ A_s Date in which auction start
5:     ▷ A_h Planning horizon for the current auction
                                  ▷ Initialization by default (Single-Item Auction)
6:     announce_size ← 1
7:     assign_size ← 1
8:     ▷ Update proxies with new information
9:     while length(R) > 0 do                                  ▷ Trigger the auction
10:        run_auction()
11:    end while
12:    ▷ Private methods definition
13:    procedure RUN_AUCTION(definition):
14:        ▷ Create announcement
15:        announcement ← create_announcement(announce_size)
16:        ▷ Broadcast announcement and ask for bids
17:        bids ← ask_for_bids(announcement)
18:        ▷ Evaluate bids
19:        winners ← evaluate_bids(bids)
20:        ▷ Notify winner of the auction
21:        notify_winner(winners)
22:    end procedure
23: end procedure
```

## 4.3 Proxy class

The Proxy class embodies the structure of the bidding agents; they can receive new information from the auctioneer to update their attributes, compute bids for announced requests, and allocate requests assigned by the auctioneer to be executed by the satellite.

This class, such as the Auctioneer class, is constituted of class methods that the Auctioneer can call to compute a bid when an `announcement` is to be auctioned. At the same time, these internal methods invoke other class methods that handle receiving, processing, and updating incoming data.

**Class structure flow process**

1. **Proxy attributes initialization and update:** The Proxy class initializes multiple attributes and receives information from the Auctioneer (e.g., proxy identifier code, auction bidding strategy, contact window plan, resource consumption map, services and payloads available, and recharge windows plan) that contains relevant data to update the proxy's status in order to compute the bid and allocate a potential new auctioned request.

   (a) **Resource tracking and saturation:** The introduction of dedicated methods for resource level tracking and resource saturation phenomena became necessary. Resource tracking is designed to keep the level of resources in the satellite model updated when new requests are assigned to it; this allows the model to be closer to real-time conditions. This tracking takes the values provided in the consumption map in order to determine how much is consumed by each requested service, as well as the recharge and contact windows.

   As stated in the problem definition, the resource saturation phenomenon occurs because the satellite has reached its maximum on-board capacity. For instance, if the battery is nearing its maximum capacity during a solar recharge window, a procedure to handle this as well as the consumption due to task execution needs to be implemented. This should enable tracking the levels of the resources while they are below the maximum capacity levels; once they exceed these levels, the profile of saturated resources is computed by the changes in slope in the profile of unsaturated resources and by adding the value of the saturated resource profile in the previous time instant. The implementation is based on the formula shown in Equation (4.1), where:

$$\bar{l}^{\rho}_{j,t} = \min\left[\hat{l}^{\rho}_{j},\ \bar{l}^{\rho}_{j,t-1} + (l^{\rho}_{j,t} - l^{\rho}_{j,t-1})\right] \tag{4.1}$$

- $\bar{l}^{\rho}_{j,t}$ is the saturated value of resource $\rho$ from proxy $j$ at instant time $t$.
- $\hat{l}^{\rho}_{j}$ is the upper bound level of resource $\rho$ from proxy $j$.
- $\bar{l}^{\rho}_{j,t-1}$ is the saturated value of resource $\rho$ from proxy $j$ at previous instant time $t-1$.
- $l^{\rho}_{j,t}$, $l^{\rho}_{j,t-1}$ are the non-saturated values of resource $\rho$ from proxy $j$ at current time instant $t$ and previous time instant $t-1$, correspondingly.

An instance of the evolution of resources over the duration of an auction of different types of satellites (i.e., a SAR and OPTICAL satellite) is shown in figure 1. Two resources per satellite type: battery and memory, can be observed. Moreover, each of them presents the evolution of two profiles in the plot: a saturated profile and an unsaturated profile. When a new request is added to the proxy, the consumption of these resources is applied to the unsaturated profile, and thus the saturated profile is computed as previously established by using Equation (4.1).

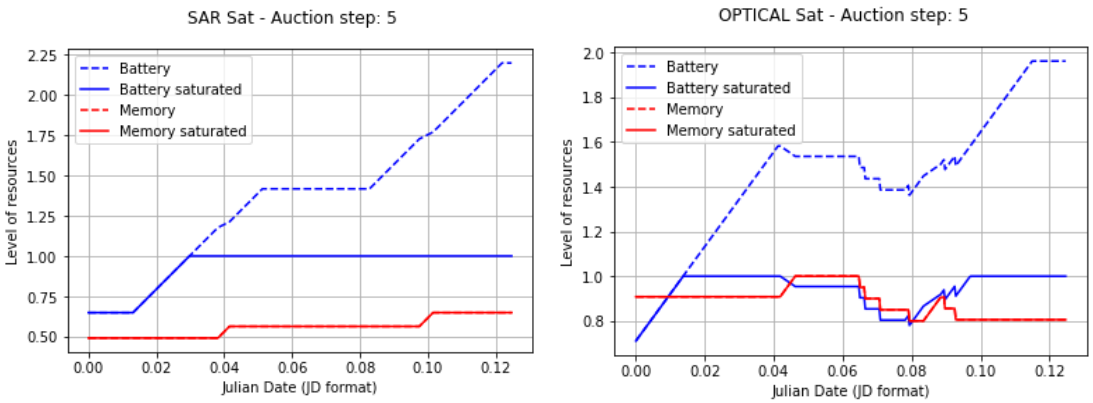



**Figure 4.7:** Evolution of resources level during an auction for a SAR and an OPTICAL satellite type

2. **Bid computation:** Once the proxy's attributes have been updated and there is an incoming request to be processed, the latter is sent to all of the computing functions of each term that comprise the final offer by evaluating a specific aspect (e.g., level of current resources or request expiration time), as described in the previous chapter. Once each term generates an output that reflects the level of satisfaction based on the conditions encountered by the satellite, the final bid is computed as the sum of all contributions multiplied by the corresponding weights established by the agent's bidding strategy and sent to the auctioneer for evaluation. Figure 4.8 depicts a graphical representation of this process.

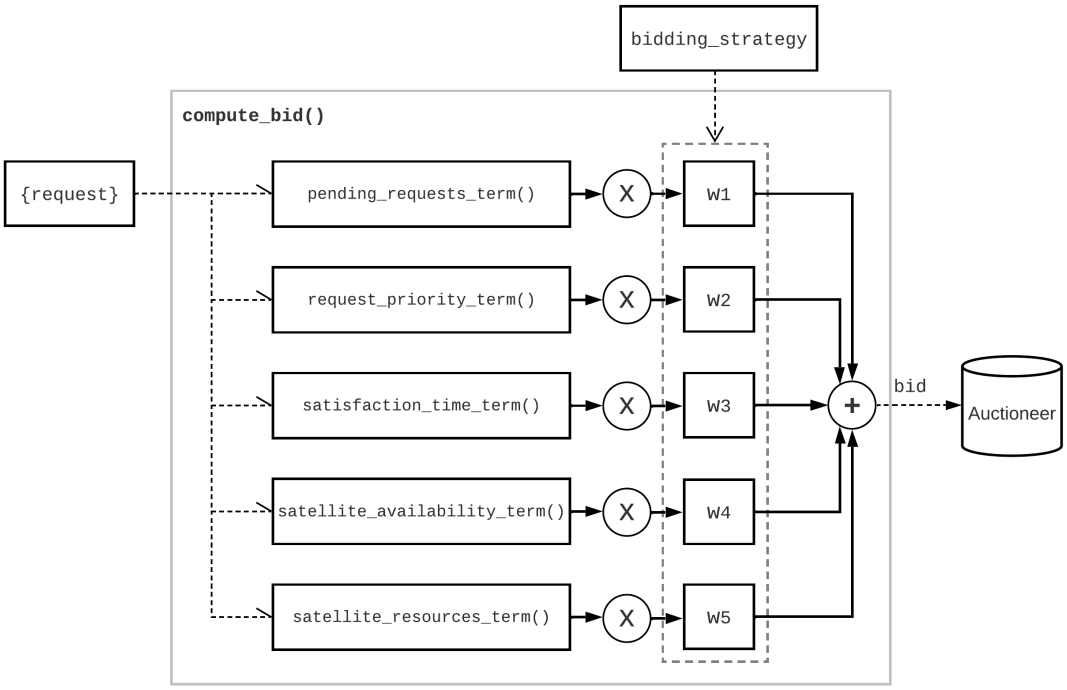



**Figure 4.8:** Bid computation internal process flowchart

3. **Assigned request allocation:** After the auctioneer has evaluated the different proxy bids and notified the winner of the auction, the assigned proxy proceeds to add it to the list of pending requests. Within this method, an update of the satellite resource levels is performed as well as a saturation of these due to the insertion of the new request.

Algorithm 2 assembles a pseudo-code of the Proxy class's structure, recalling all the class methods processes described above, as well as their interactions with the proxy class attributes. The first part of the class definition comprises the initialization of the Proxy class attributes, the update of the resources level and the saturation process of these. The second part is composed of the definition of the class methods that can be invoked externally to the class definition (i.e., the compute bid and add request methods that can be called by the Auctioneer). And finally, the definition of the class methods that are requested for the computation of the different bid terms.

**Algorithm 2** Proxy class definition

1: **procedure** CLASS: PROXY(ID, `biddingStrategy`, $C, \mathcal{K}, K, \Pi, L, R_w$)

▷ Attributes initialization

2: ▷ `biddingStrategy` is the bidding strategy containing all the weights
3: ▷ $C$ is the set of contact windows with ground
4: ▷ $\mathcal{K}_m$ is the consumption map
5: ▷ $K$ are the set of services available
6: ▷ $\Pi$ are the set of payloads available
7: ▷ $L$ are the level of resources
8: ▷ $R_w$ are the set of recharge windows
9: ▷ Updating proxies attributes with assigned requests
10: ▷ and given contact windows consumption
11: $L \leftarrow$ `resource_update()`
12: ▷ Generating saturated resource profile
13: $L \leftarrow$ `resource_saturation()`
14: ▷ Bid computation definition
15: **procedure** COMPUTE_BID(`definition`):
16: ▷ Saving weights from bidding strategy
17: $W \leftarrow [\texttt{biddingStrategy}]^T$
18: ▷ Bid generation
19: `Bid` $\leftarrow B \cdot W$
20: ▷ Sending final bid to auctioneer
21: `send_bid(bid)`
22: **end procedure**

▷ Bid terms computation

23: ▷ Pending requests term
24: $b_1 \leftarrow$ `pending_requests_term()`
25: ▷ Request priority term
26: $b_2 \leftarrow$ `request_priority_term()`
27: ▷ Satisfaction time term
28: $b_3 \leftarrow$ `satisfaction_time_term()`
29: ▷ Satellite availability term
30: $b_4 \leftarrow$ `satellite_availability_term()`
31: ▷ Satellite resources term
32: $b_5 \leftarrow$ `satellite_resources_term()`
33: ▷ Saving all the contributions
34: $B \leftarrow [b_1, b_2, b_3, b_4, b_5]$

## 4.4 Manual testing framework

A series of test cases are required to ensure the integrity of each of the previously defined methods, as well as the correct operation of the code. Each manual test case designed for each class is described as follows.

**Auctioneer class testing**

- Request addition: This test case checks that the auctioneer correctly appends incoming requests within the attributes. This is to ensure that the request is not added more than once or to ensure repeatability when multiple requests have to be added.
- Proxies set: Since it is necessary to add the list of proxies available for bidding in the auction, a test case is required to check that the specific proxies were correctly added by exploiting the identification code of each of them. At the same time, the auctioneer has other methods to remove proxies from its list, and therefore a check on this removal process should be performed.
- Auction execution: This test case checks, in a completely randomized auction, that the announcement and assignment sizes were correctly inherited. This is done by checking the number of requests that were sent to the proxies and those that were ultimately assigned. It also checks that the assigned requests are in the proxies to which they were assigned, again by exploiting the identifier code of both the requests and the proxies.

**Proxy class testing**

- Bidding terms tests:

  a) Pending requests term: This test case for the bid term that considers the number of assigned requests checks the proportionality of the output obtained with the reasoning function as multiple requests are added to the proxy.

  b) Request priority: Test case with the same objective as the previous one mentioned above, to check the proportionality of the reasoning function output when requests with different levels of client priority and temporal priority are introduced.

  c) Satisfaction time term: This test case was implemented by creating different contact windows for multiple proxies, in order to check the output of the satisfaction time reasoning function from the time to target of the request to the first available contact window of the satellite after it.

d) Satellite availability term: For this test case, a definition of multiple conflicting requests was developed to check the output (i.e., whether it is in conflict or not) of this method, from combinations of conflicts of service requests up to contact window conflicts with service requests.

e) Resource term: This test checks the correct estimation of the satellite resource evolution when generating the bid term by inserting requests for different services and consumption ranges. In addition to this, it also checks the creation of saturation profiles for each resource, since this is the basis for calculating the bid term.

## 4.5 Simulation routine

Running multiple simulations requires the design of a routine that can vary different parameters in order to obtain results that allow to analyze the effect of each one of them on the final performance. These routines contains multiple degrees of freedom allowing specify the whole scenario characteristics. These degrees of freedom are listed below:

- Announcement and assignment sizes
- Bidding strategy for Proxies
- Sensitivity parameters per bid term function
- Number of Proxies, incoming requests, available services, and among others.

The general structure of the simulation routine, as shown in Figure 4.9, includes an initial conditions phase in which all the auction, auctioneer, and proxy specifics are established. The second phase is in charge of varying the degrees of freedom seen above by running multiple simulations and generating data accordingly. Finally, all auction data is recorded for further processing and analysis.

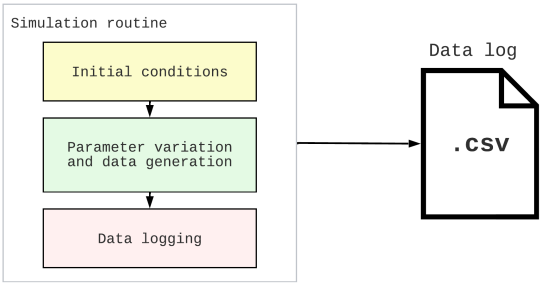



**Figure 4.9:** Simulation routine flow process

# Chapter 5

# Experimental Results

This chapter presents the results obtained by applying the developed architecture for dynamic tasking to a realistic satellite constellation scenario. Firstly, the benchmark scenario's set-up is described in Section 5.1. Secondly, a scalability analysis is developed in Section 5.2, where the results and time complexity analysis for different experiment settings are presented. Finally, Section 5.3 presents a sensitivity analysis performed on a variety of experimental setups to examine the effects of different bidding strategies of the Proxies on the final task assignment.

## 5.1 Scenario set-up

The reference scenario considered for the generation of results and their subsequent analysis is an Earth Observation (EO) constellation in Low Earth Orbit (LEO) with available optical and Synthetic Aperture Radar (SAR) payloads. The following parameters have been set for its execution, as follows:

- **Orbital period:** For this scenario, a orbital period of $\sim$90 min is considered, an average duration for a satellite in low earth orbit (LEO).
- **Planning horizon:** A planning horizon duration has been established up to $\sim$2 orbits, about 200 minutes.
- **Resource resolution:** The projection of the resource consumption is performed with a resolution of 400 points over the planning horizon, that is one point every 30 seconds.
- **Resource initialization:** For each proxy, the initialization of the resource levels is set to a random value between $[0.2; 1]$. By having a non-zero minimum value, it is expected that the satellite will keep a certain percentage of resources either for maintenance or to avoid complete depletion.

- **Requests expiration interval:** The expiration time for requests that need a SAR payload has been set up to a maximum of 2 days. For other types of requests the expiration time is set up to 1 day.

- **Auction repetition:** Each scenario parametrization tested is run three times with randomized initialization in order to improve the reliability of data.

The consumption map for each satellite activity (which comprise payload services, downlink, and recharge) is detailed in Table 5.1. The percentage consumption is given for the battery and memory resources, as well as their duration in minutes. Memory consumption values for payload service requests were extrapolated from typical sizes for Optical/SAR images on real satellites. On the other hand, the battery consumption values were extrapolated based on how many acquisitions the satellite was expected to make within a single recharge cycle, depending on common payload duty cycle. Thus, it can be seen that the consumption is low for optical images because it is expected to acquire many of these, in contrast to being very high for services that require SAR-type payload.

In this scenario, the consumption for payload service requests is provided as cumulative values referring to the overall battery and memory needed to perform a single acquisition. Instead, the consumption values for solar recharges and antenna downlink are given as consumption rates per simulation step to account for the variability in the activity duration (i.e., the overall consumption of these activities is equal to the consumption rate times the activity duration).

| **Service** | **Payload** | **Battery [%]** | **Memory [%]** | **Duration [min]** |
|---|---|---|---|---|
| Image | Optical | -5 | -5 | 1 min |
| Spot | SAR | -30 | -5 | 3 min |
| Strip | SAR | -50 | -10 | 5 min |
| Combined Acquisition | Optical + SAR | -20 | -3 | 2.5 min |
| Downlink | Antenna | -0.3333 | 0.6666 | [5; 10] min |
| Recharge | Solar Panels | 0.6666 | 0 | [50; 60] min |

**Table 5.1:** Consumption map for each Service-Payload in the scenario

The experiments described in the following sections are performed considering a constellation with a uniform distribution of proxy types (i.e., for any experiment, approximately one third of the total proxies are optical type, another one third are SAR type, and one third present both payloads available), as can be seen in Table 5.2.

| Optical | SAR | Optical + SAR | Total |
|---|---|---|---|
| 2 | 2 | 1 | 5 |
| 9 | 8 | 8 | 25 |
| 17 | 17 | 16 | 50 |
| 25 | 25 | 25 | 75 |
| 34 | 33 | 33 | 100 |
| 42 | 42 | 41 | 125 |
| 50 | 50 | 50 | 150 |

**Table 5.2:** Constellation sizes and composition considered for scalability tests

The overall number of proxies in the constellation depends on the specific experiment that has to be performed. For the scalability analysis, an increasing number of proxies is considered following the values reported in Table 5.2 so as to retrieve meaningful information on the time complexity of the algorithm. Instead, for the sensitivity analysis presented in the following chapters, a single experiment is chosen (the one corresponding to the colored row in the Table), with a total number of proxies of 25, as mentioned before, maintaining a uniform distribution per satellite type.

Moreover, Table 5.3 describes the cases considered for the scalability tests concerning the number of requests to be assigned and their distribution by type of service requested. Then, the sensitivity analysis is performed on the colored one, an experiment with a high number of requests to be assigned. In the considered scenario, the distribution of pending requests is consistent with the constellation capability and the duty cycle of the different payloads. This follows from the assumption that the constellation is correctly sized with respect to the user demand.

When the auction starts, satellites might have already assigned requests from previous rounds. The number of already allocated requests per satellite type for this scenario is described as follows:

- **Optical Satellites:** They have a random number of allocated requests between [0-10], with the service requested being Image acquisition.
- **SAR Satellites:** They have a random number of allocated requests of [0,1], with the service requested being either Spot or Strip acquisition.
- **Optical-SAR Satellites:** They have the following number of allocated requests per service type.

  a) A random number of Optical allocated requests between [0-5].

| Image | Spot | Strip | C. Acquisition | Total |
|---|---|---|---|---|
| 101 | 7 | 5 | 2 | 115 |
| 501 | 37 | 26 | 11 | 575 |
| 1001 | 75 | 52 | 22 | 1150 |
| 1501 | 112 | 78 | 34 | 1725 |
| 2001 | 150 | 104 | 45 | 2300 |
| 2502 | 187 | 129 | 57 | 2875 |
| 3002 | 224 | 155 | 69 | 3450 |

**Table 5.3:** Number of requests to be assigned and distribution with respect to the type of requested service considered for scalability tests

b) A random number of SAR allocated requests between [0-1]; that can require a Spot, Strip or a Combined Acquisition service.

Table 5.4 presents the set-up of the scaling parameter values for each bid function already described in the problem definition. These correspond to the hyperbolic functions' sensitivity parameter for the terms of pending requests, priority of requests, and satisfaction time. These values were obtained by reasoning on the expected range of the hyperbolic functions inputs given the scenario characteristics (e.g. planning horizon, ground contact frequency).

| Parameter | Value |
|---|---|
| $\zeta_q$ | 0.1 |
| $\zeta_p$ | 15 |
| $\zeta_c$ | 30 |

**Table 5.4:** Sensitivity parameters for hyperbolic functions of bidding terms

## 5.2 Scalability analysis

The scalability analysis is perfomed in two different scenarios. The first one performs time complexity analysis for single-item auctions varying the constellation and pending request list sizes as shown in Tables 5.2 and 5.3. The second one performs time complexity analysis in multiple experiments varying the announcement size and assignment size.

### 5.2.1 Single-Item Auctions experiments

Figure 5.1 plots the evolution of time complexity for all combinations of experiments contained in the scenario description.

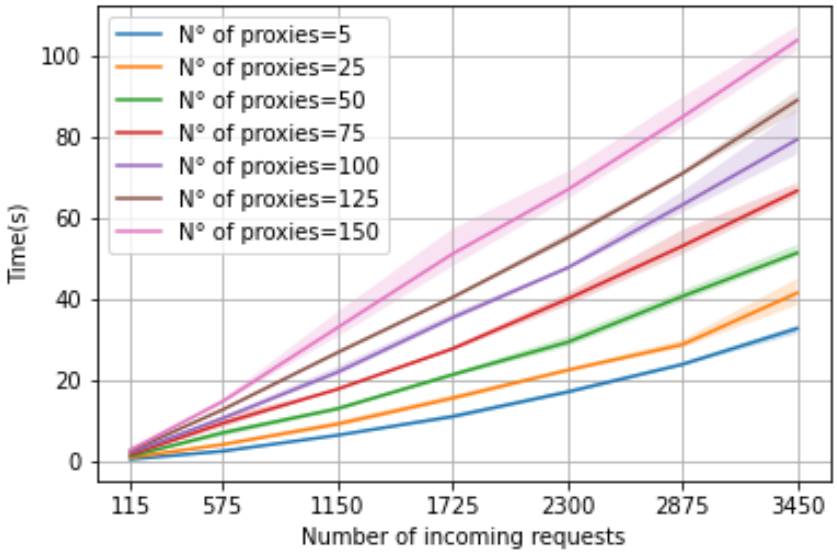



**Figure 5.1:** Average time complexity plot with maximum/minimum intervals - Single-Item Auctions

These results have been obtained using an off-the-shelf laptop with Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz and 12 GB of RAM. It can be observed that in case of single-item auctions, the time complexity exhibit almost a linear dependence. More specifically, the time complexity is situated in a type of exponential degree with such a low slope that for practical purposes it approximates a linear degree complexity, meaning that the model can be scaled without concern of obtaining an accelerated growth in complexity.

Taking instances from the graph, it can be seen that in an extreme case of: 3450 incoming requests and 150 proxies; the total average time spent by the auction is just over 100 seconds, less than 2 minutes.

### 5.2.2 Announcement/Assignment size experiments

Different experiments have been conducted to analyze the time complexity by varying the announcement and the assignment size. These are listed below along with their respective complexity analysis of the auction performance. The comparisons of time complexity for various experiments are primarily focused on the simulation case with 50 proxies and 575 incoming requests.

- **Announcement size: 50/Assignment size: 1**

  Figure 5.2 shows a simulation case with the highest ratio between announcement size and assignment size, considering that the assignment size takes the minimum possible value. As it can be observed in the plot, for the previously mentioned case study, the time complexity increased x20 times (∼100 sec) with respect to the complexity presented in the single-item auction (∼5 sec).

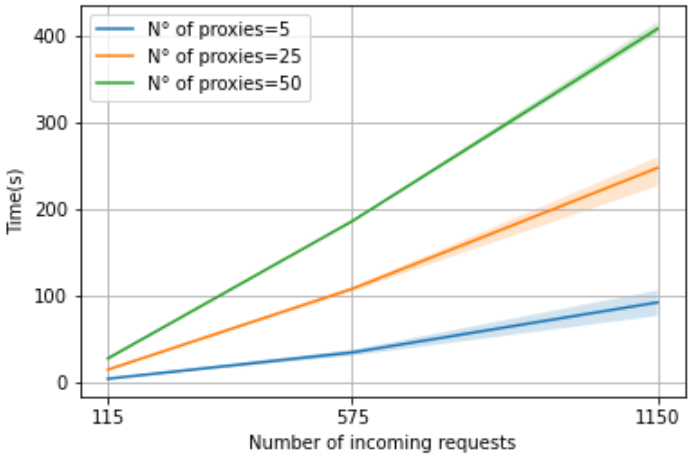



**Figure 5.2:** Average time complexity plot with maximum/minimum intervals - Announcement size: 50/Assignment size: 1

- **Announcement size: 50/Assignment size: 10**

  Figure 5.3 shows the time complexity for a smaller ratio than the previous simulation case. It can be seen that as the ratio between announcement/assignment decreases, the time complexity decreases. For instance, for the case study with 25 proxies and 575 incoming requests, an average time of about 10 seconds is obtained, x10 times less than the previous case and twice as long as an single-item auction.

- **Announcement size: 50/Assignment size: 50**

  Figure 5.4 shows the time complexity evolution results for an unitary ratio simulation case, as also presented in the single-item auctions (i.e., assignment and announcement have the same size).

  Better performance in terms of time complexity is provided with this announce/assign ratio, with an average time of about ∼3.5 seconds for the case of 25 proxies and 575 incoming requests. Specifically, it presents a 30% reduction with respect to an single-item auction.

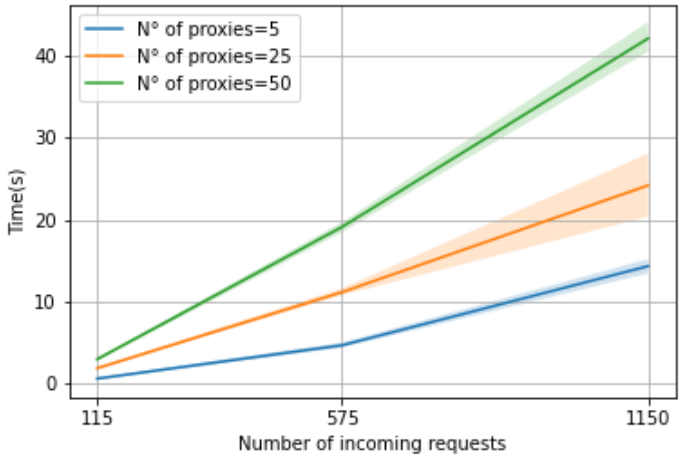



**Figure 5.3:** Average time complexity plot with maximum/minimum intervals - Announcement size: 50/Assignment size: 10

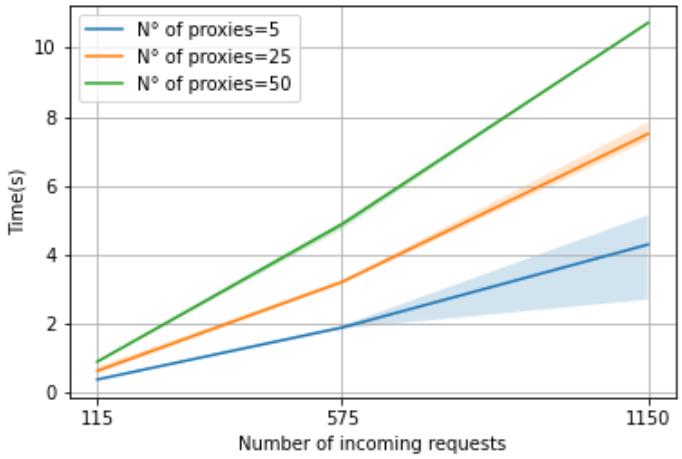



**Figure 5.4:** Average time complexity plot with maximum/minimum intervals - Announcement size: 50/Assignment size: 50

- **Announcement size: 100/Assignment size: 50**

  Figure 5.5 shows the time complexity results for a case with an announcement size twice that previous cases and with assignment size different from the minimum quantity.

  No major changes in complexity can be established between this case study and single-item auctions for the experiment with 25 proxies and 575 incoming requests, since both total average times are within 5 seconds. However, differences

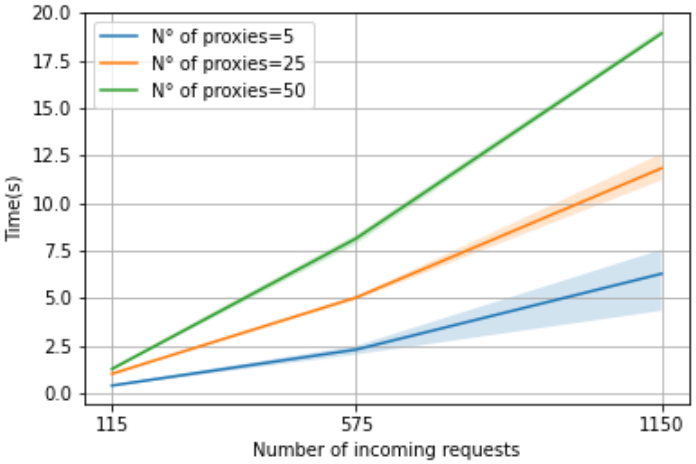



**Figure 5.5:** Average time complexity plot with maximum/minimum intervals - Announcement size: 100/Assignment size: 50

in time averages tend to be lower in this case with a larger number of proxies.

A better comparison can be made with the Tables 5.5 and 5.6. For instance, a x4-fold increase in the auction duration can be observed for the case of announcement size of 50 and assignment size of 1, by comparing the values of each table.

Table 5.5 summarizes the comparative analysis in terms of time complexity described previously for the experiment with 25 proxies and 575 incoming requests, classified by announcement and assignment size.

| Proxies | Requests | Announcement | Assignment | Time (s) |
|---|---|---|---|---|
| 25 | 575 | 1 | 1 | 5 |
| 25 | 575 | 50 | 1 | 100 |
| 25 | 575 | 50 | 10 | 12 |
| 25 | 575 | 50 | 50 | 3.5 |
| 25 | 575 | 100 | 50 | 5 |

**Table 5.5:** Time complexity comparative between different simulation cases

It is also worth noting that in the preceding simulations, the bidding strategy retained the same weight for all terms because it does not have major impact over time complexity; therefore, the complexity analysis is centered on the announcement and assignment size.

A further analysis could be conducted to determine a balance between the quality of the bids and the cases that present a large announcement/assignment

| Proxies | Requests | Announcement | Assignment | Time (s) |
|---|---|---|---|---|
| 50 | 1150 | 1 | 1 | 50 |
| 50 | 1150 | 50 | 1 | 400 |
| 50 | 1150 | 50 | 10 | 45 |
| 50 | 1150 | 50 | 50 | 15 |
| 50 | 1150 | 100 | 50 | 18 |

**Table 5.6:** Time complexity comparative between different simulation cases

size ratios. In this way, it could be determined that the use of such a high ratio is worthwhile since higher bids are generally chosen during the evolution of the auction. This is because the size of the assignment is related to the number of bids with the highest value for an item. Furthermore, because the time complexity remained approximately linear for the different experiments, deeper explorations are not required since the model already reflects an appropriate capacity to be scaled, which was one of the main objectives for the simulations performed. As a result, for the sensitivity analysis, all experiments are run in single-item auction mode, as it is the simplest case to conduct the sensitivity analysis and study the effects with respect to the bidding strategy. This also helps to remove the influence of multiple allocations from the auction outcome.

## 5.3 Sensitivity analysis

Each experiment is designed by changing the weight of a specific term that composes the total bid. Three experiments are analyzed in this section: an experiment with uniform distribution of weights, a set of null experiments setting one weight at a time equal to 0 (thus removing the bidding term from the final bid), and a set of predominant experiment setting a higher value to one of the weights.

### 5.3.1 Uniform weights experiment

In this experiment, each of the 5 bid terms has a corresponding weight of 0.2, thus having a homogeneous distribution.

Figure 5.6 shows the distribution of assigned requests before and after the auction filtered by satellite type. A reduction of the load balance can be observed after the auction.

Table 5.7 shows the numerical values of the mean and variance before and after the auction. It can be observed that with this weighting configuration high variance conditions are present, especially for satellites with optical payload, due to the fact

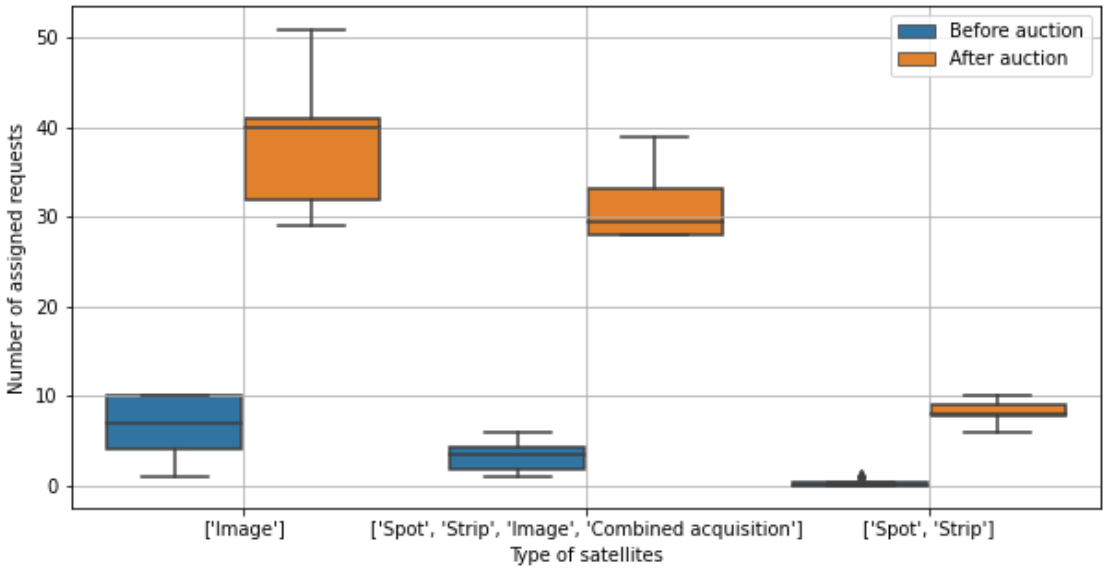



**Figure 5.6:** Box plot of load balance before/after auction per Satellite type - Uniform weights

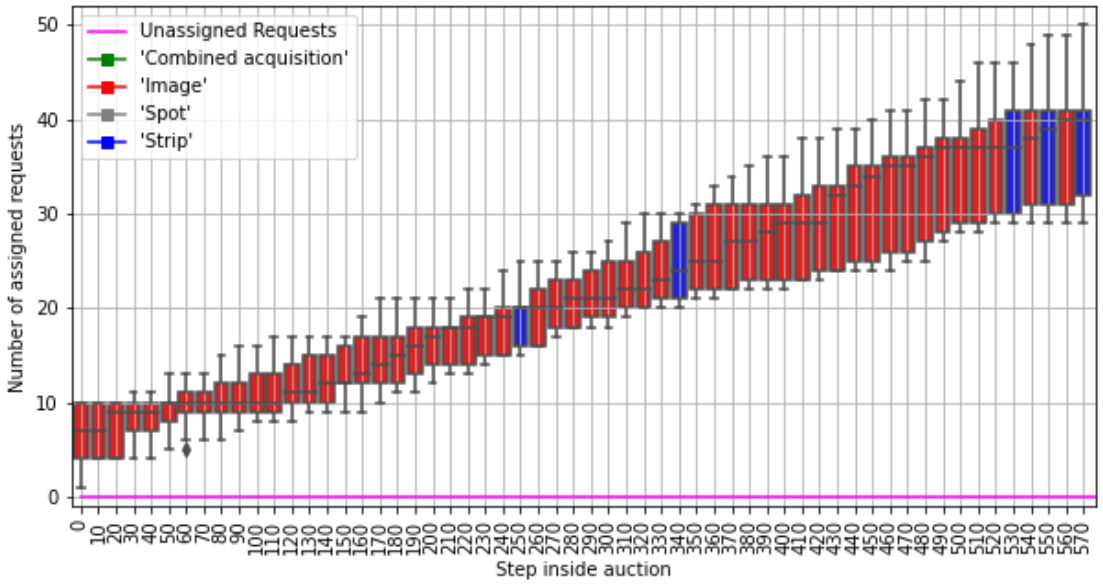
that in the distribution of incoming requests a large proportion of these require this type of service. On the other hand, for satellites with SAR type payloads, whose incoming requests are fewer, a lower variance is observed.

| Satellite type | Average before | Variance before | Average after | Variance after |
|---|---|---|---|---|
| Optical | 6.333333 | 11.777778 | 38.444444 | 41.358025 |
| Optical+SAR | 3.250000 | 2.937500 | 31.125000 | 14.109375 |
| SAR | 0.250000 | 0.187500 | 8.125000 | 1.359375 |

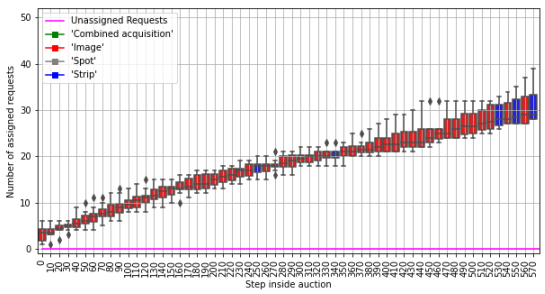
**Table 5.7:** Load balance comparison before/after auction per Satellite type - Uniform weights

Figure 5.7 shows the evolution of the load balance per satellite type. As expected, there is an increasing trend in the number of assigned requests as the auction evolves, as well as an increasing trend in the variance. The latter is more prominent in optical satellites. This type of graph helps to understand how the distribution of assigned requests evolves as the auction progresses by determining whether a high variance environment is being experimented among the proxies or, on the contrary, the auction distributes incoming requests in a balanced way.

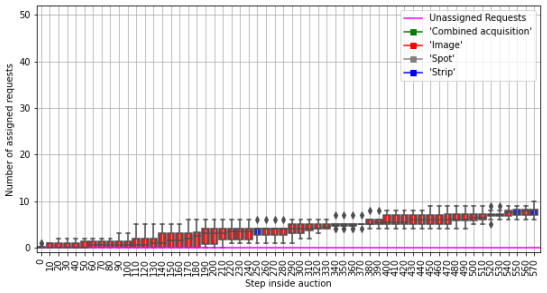
Figure 5.8 shows the evolution of the total bid during the entire auction, as well




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR



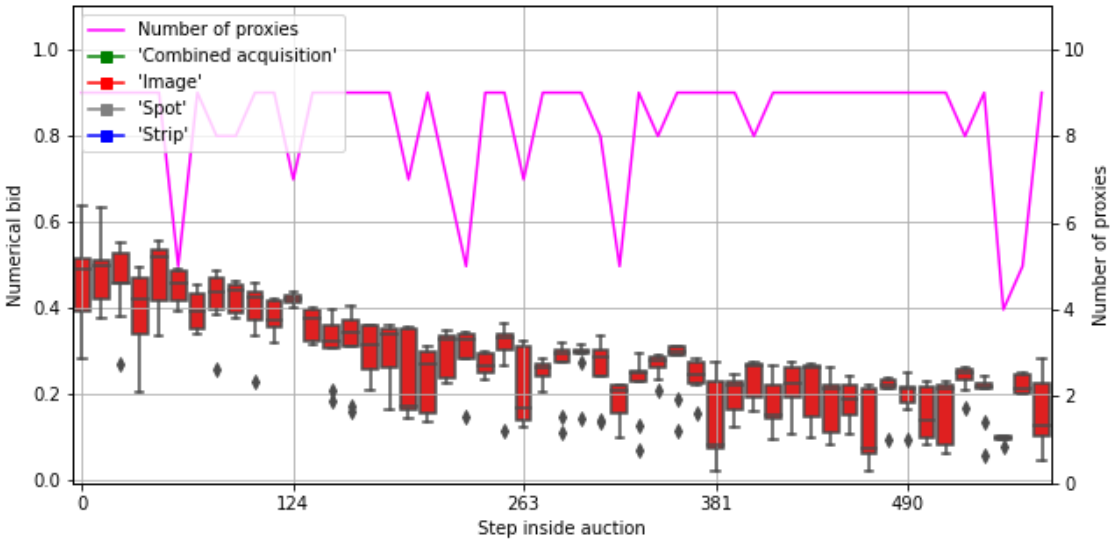

**(c)** Satellites types: SAR

**Figure 5.7:** Box plot of load balance evolution per satellite type - Uniform weights

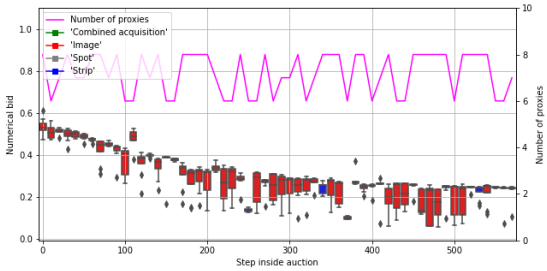
as the number of proxies participating in each step. The main noticeable feature of this result is the decreasing trend with a slow slope as the auction evolves. This decreasing trend is expected since as increasingly more requests are assigned, for instance, it is expected that the load term decrease within each auction step. Of course, for SAR satellites, due to the small number of requests they assume, this trend is not as pronounced.

This graph validates why there are no unassigned requests, as shown in Figure 5.7, because there are no instances where no proxies bid on a request, as shown in the line plot containing the number of proxies that bid at each step of the auction.

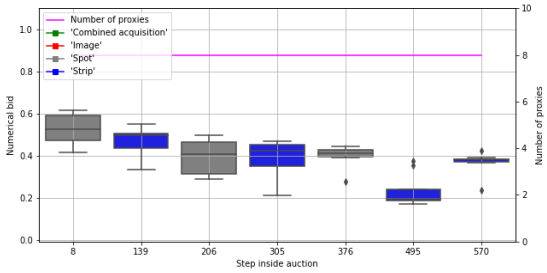
Figure 5.9 shows the evolution of the number of conflicts among the requests assigned to a Proxy over the number of requests assigned to that Proxy. As a note to consider, the maximum number of conflicts among the list of all requests to be




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR



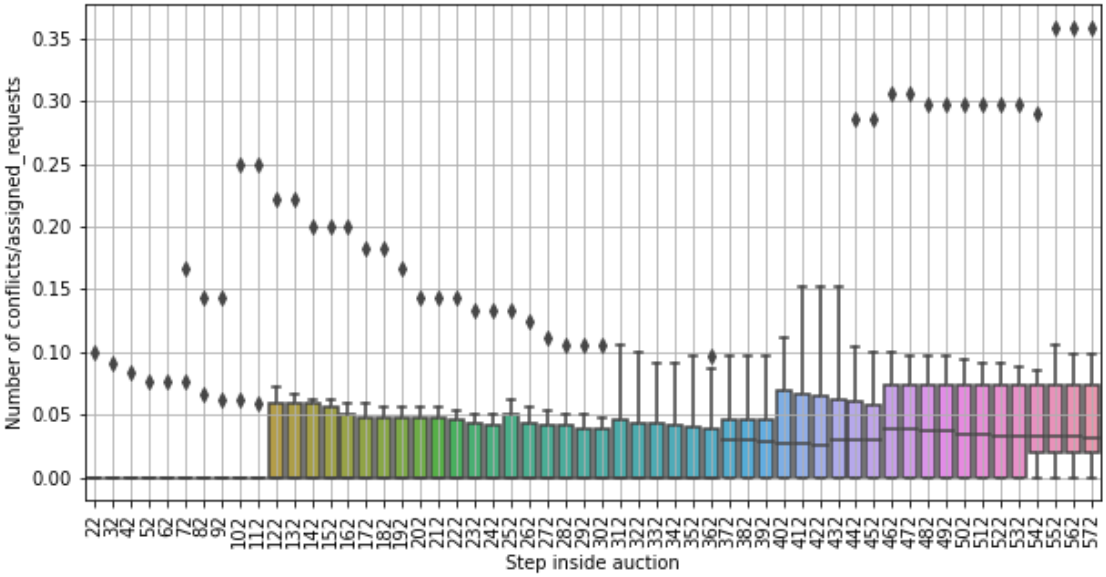

**(c)** Satellites types: SAR

**Figure 5.8:** Box plot of bid evolution per satellite type and number of Proxies bidding - Uniform weights

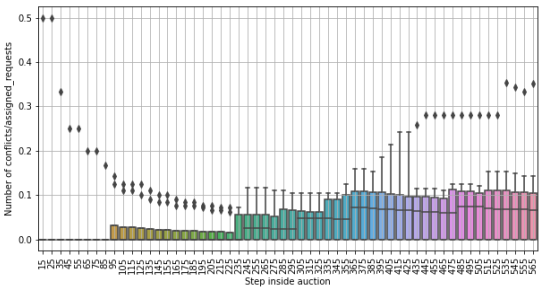
assigned with this configuration is 2.8434.

This graph shows two main points to highlight in the model with this weight configuration. The first is by looking at the maximum conflict ratio (i.e., for the worst-case Proxy) tends to decrease, especially in early stages; this means that the algorithm prefers to assign non-conflict tasks to that Proxy. The second by looking at the average conflicts, the trend is increasing, but slowly and stadily, meaning that conflicts tends to be evenly distributed among satellites.

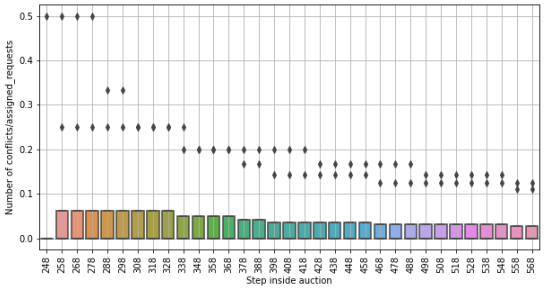
Furthermore, reaffirming what was previously established, is that the largest conflict ratio data that can be observed in the graphs does not exceed 0.35 in contrast to the maximum ratio of 2.8434, a similar behavior for the graphs of other types of satellites with this configuration. Therefore, for this configuration of weights, modeling conflicts allows an optimal reduction of conflicts among potentially assignable requests. Further verification for other weight configurations




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR




**(c)** Satellites types: SAR

**Figure 5.9:** Box plot of number of conflicts/maximum potential conflicts per satellite type - Uniform weights

will be reported.

Figures 5.10, 5.11 show the evolution of the different partial bid terms contained in the final bid for the present weight configuration of different satellites types. Two types of graphs are observed: the stacked area plot shows the contribution to the final bid by each term at each step of the auction, and the line plot shows the values of each term without multiplying by its respective weight. The first one helps to understand the percentage weight that each term has with respect to the final bid, and the second one helps to examine individually the evolution of each term. These plots correspond to a specific proxy, similar behavior is present between satellites of the same type.

It can be observed that the satellite availability term (conflict term) becomes

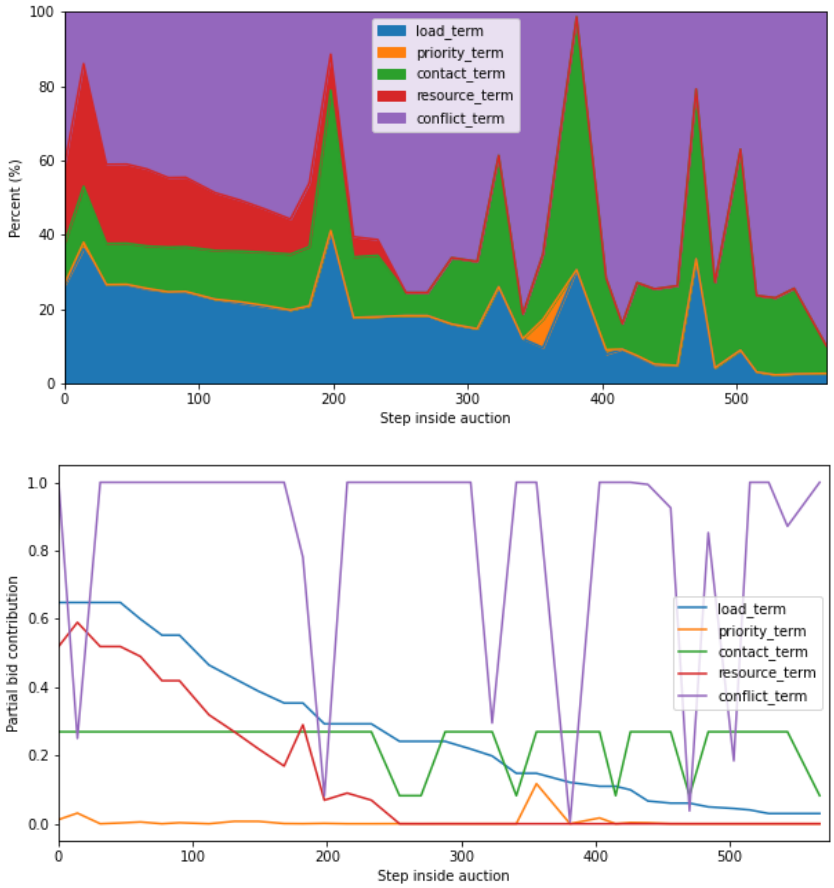
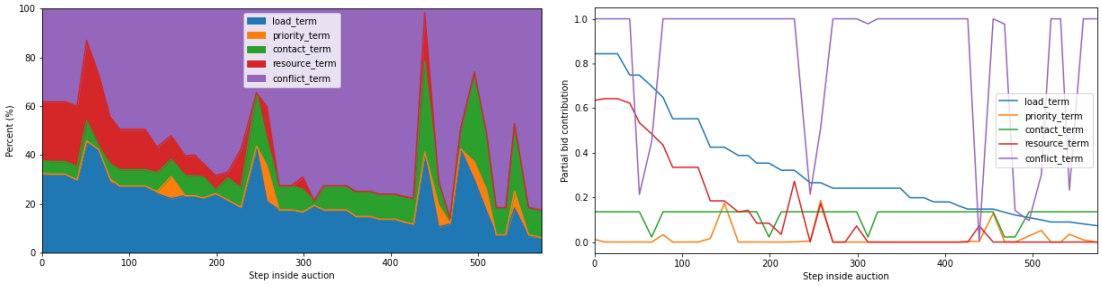


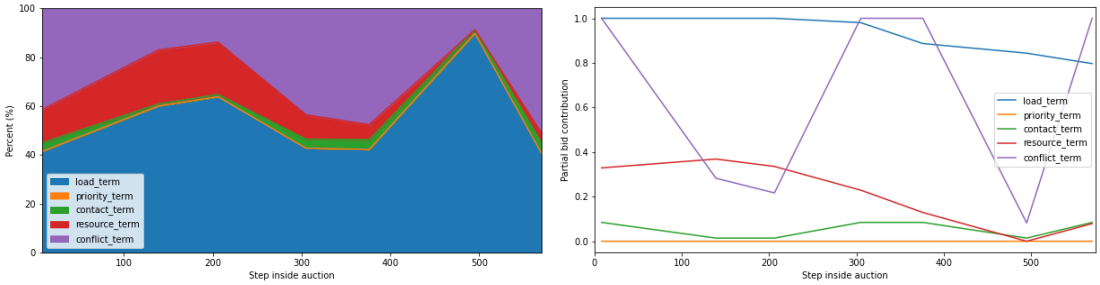

**Figure 5.10:** Stacked area plot and line plots per partial bid term of Optical satellite type - Uniform weights

more relevant as the auction evolves with the increasingly assignment of requests. This is due to the decrease in the contribution of satellite resources and pending requests term (load term), since with the allocation of new requests, a decreasing trend in these terms is expected. It should be noted that each term is independent of the step within the auction.

The line plots also confirms what was previously stated regarding the conflict management by the auction, since it can be observed just few drops in the conflict term instead it remains more or less constant (and higher) during the whole auction; therefore, the conflict management is performed such as the number of assigned requests increases, the probability of a new request to be in conflict increases. This behaviour can be seen from the highest frequency of conflict terms different from 1 as the auction progresses.




**(a)** Satellites types: Optical+SAR




**(b)** Satellites types: SAR

**Figure 5.11:** Stacked area plot and line plots per partial bid term of other satellite types - Uniform weights

Preliminarily, it can be observed that the term that models the satisfaction time (contact term) discretely takes two values as the auction evolves, which coincides with the conditions given for the scenario because, for a horizon of two orbits, 2/3 contacts per orbit were established; thus, the values in which this term range are expected. The priority term does not make a major contribution as the auction progresses since it does not depend on the satellite's own capabilities but on the characteristics of each request. Therefore, it is expected that the different proxies bidding for the same request will have the same value for this term, as observed in the priority term line plot for an Optical satellite and an Optical+SAR satellite, which have the same evolution profile for the priority term.

Finally, a resource depletion is presented, as shown by the satellite resource term, while at the same time the load term continues to decrease indicating the assignment of more requests. This behavior could happen because no constraint is introduced in the problem definition to avoid it. However, it can be handled by tuning the parameters involved in order to reduce this type of behavior. Further tuning experiments will be carried out to test the relevance of each term.

### 5.3.2 Null weights experiments

The following experiments are designed by giving a null weight to a specific term that composes the total bid and setting the others to 0.25. The term with the null weight value is set by the type of experiment to be simulated; this will help to determine the relevance of each term by analyzing the effects on auction behavior.

- **Null weight: Load balance**

  This experiment is set up so that the null term is the pending requests (load term). The analysis of its effect on auction behavior is described below.

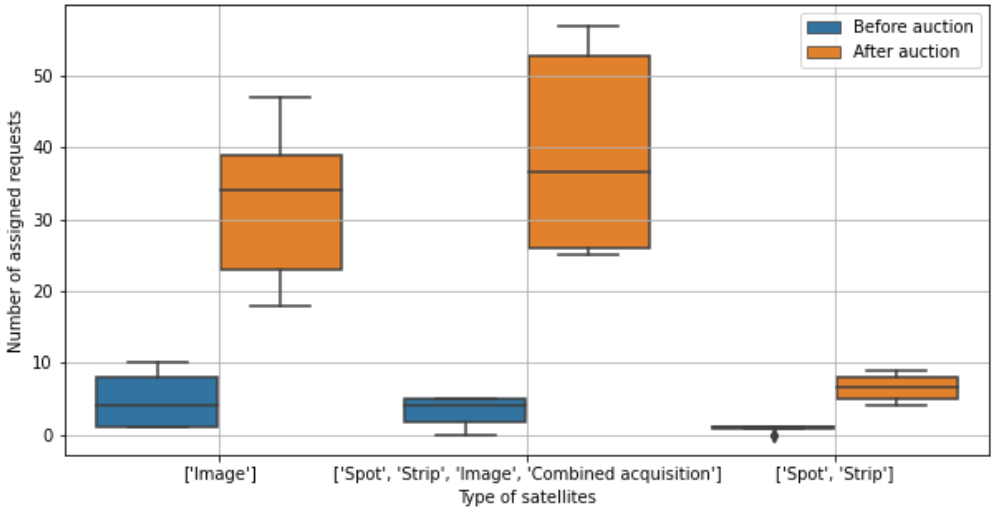


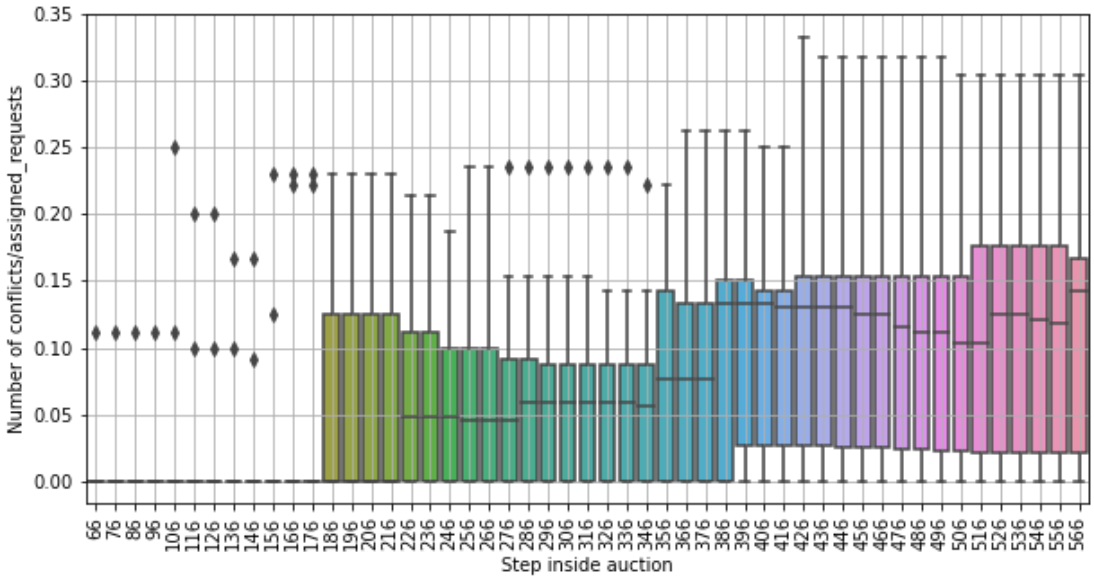

**Figure 5.12:** Box plot of load balance before/after auction per Satellite type - Null weight: Load balance term

| Satellite type | Average before | Variance before | Average after | Variance after |
|---|---|---|---|---|
| Optical | 4.888889 | 12.098765 | 31.666667 | 97.555556 |
| Optical+SAR | 3.250000 | 3.437500 | 39.250000 | 175.937500 |
| SAR | 0.750000 | 0.187500 | 6.500000 | 2.750000 |

**Table 5.8:** Load balance comparison before/after auction per Satellite type - Null weight: Load balance term

In principle, the load term is one of the main limiters of the variance of assigned requests between proxies since it bids based on the amount already assigned to a specific one. Therefore, in an experiment in which it has a null value, high

variance conditions are expected, as can be seen in Figure 5.12, even much higher (about $\sim$ 93%) than in previous experiments and being more noticeable in Optical + SAR satellites because they are capable of allocating numerous requests from a wide variety of services. This can be supported by the numerical data shown in Table 5.8, which shows a high increase in the post-auction variance of at least 92% with respect to that reported in previous experiments.

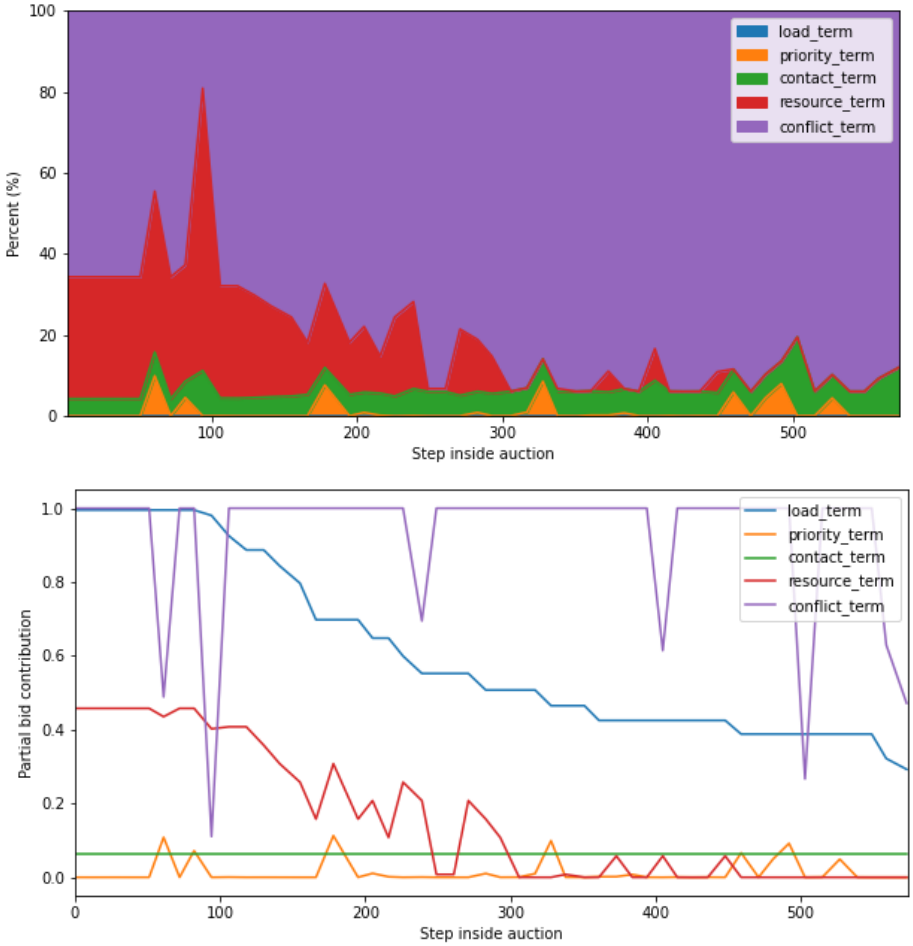



**(a)** Satellites types: Optical

**Figure 5.13:** Box plot of number of conflicts/maximum potential conflicts per satellite type - Null weight: Load balance term

A characteristic to evaluate in this experiment is how the conflict resolution evolves during the auction, this is due to the fact that the term of pending requests (load term) is null, the other terms gain more weight in the final bid. Therefore, as can be seen in Figure 5.13, a smaller ratio of conflicts is present compared to previous experiments.

To validate the fact that the other terms become relevant in the final bid. Figure 5.14 shows the evolution of the bid terms for Optical satellites, in which it can be observed that the satellite availability term (conflict term) comprises the highest relevance as the auction progresses, explaining the aforementioned fact of a smaller conflict ratio present in this experiment. On the contrary, the priority term only presents sporadic occurrences of contribution in the final bid, behavior also present in previous experiments, a deeper analysis of this term is performed in the following experiment.




**(a)** Satellites types: Optical

**Figure 5.14:** Stacked area plot and line plots per partial bid term of specific satellite types - Null weight: Load balance term

- **Null weight: Request priority**

  The null term for this experiment is the request priority (priority term). The analysis of its effect on auction behavior is described as follows.

  In Figure 5.15, it can be observed that there are no significant changes in the variance after the auction compared to previously presented experiments. As in the case of uniform weights, there is more variation in optical satellites due to the number of requests for this type of service and, on the contrary, less variation in SAR satellites. It can be stated that the priority term does not cause a major impact on the distribution of the assigned requests among the proxies.

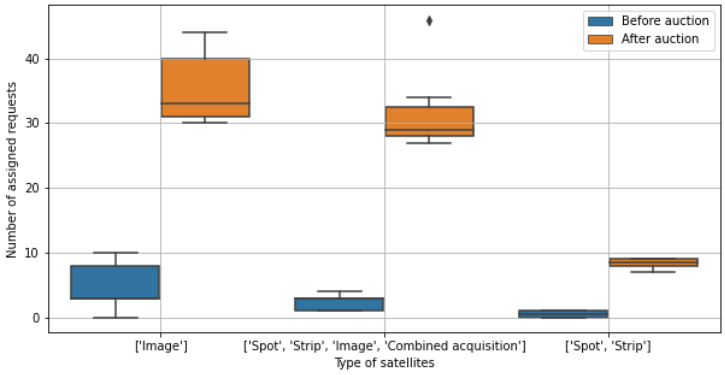
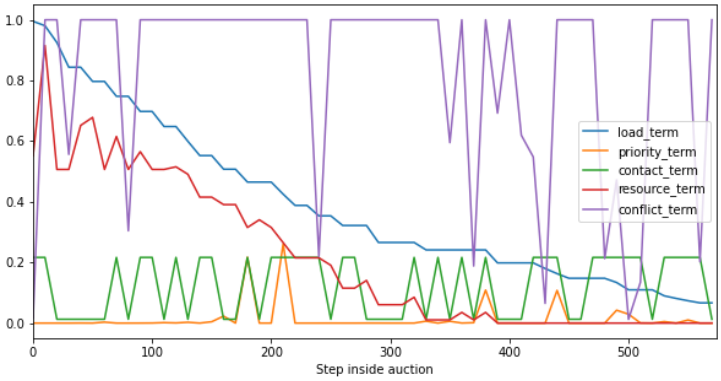
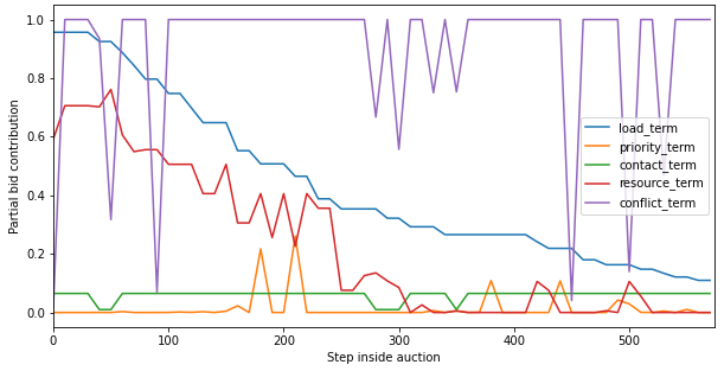



**Figure 5.15:** Box plot of load balance before/after auction per Satellite type - Null weight: Request priority term




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR

**Figure 5.16:** Line plots per partial bid term of Optical satellite types - Null weight: Request priority term

In Figure 5.16, the line plots show that the other terms are expected to maintain a behavior similar to previous experiments. This also validates what was previously mentioned for the experiments with uniform weights, that the priority profile of the requests remains the same for all types of satellites because it depends on the requirements of the requests themselves.

- **Null weight: Request satisfaction time**

  This experiment is set up so that the null term is the request satisfaction time (contact term). The analysis of its effect on auction behavior is described below.

  This experiment has produced results with low variance conditions, the highest variance reduction with respect to other experiments. It can be seen in Figure 5.17 that for all types of satellites it starts with a certain variance (roughly high before auction) and after the auction it is drastically reduced.

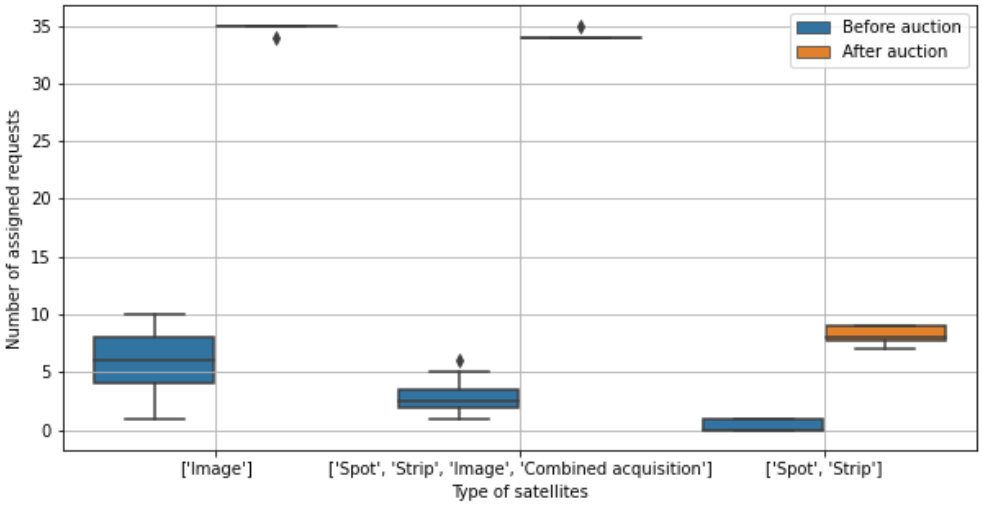
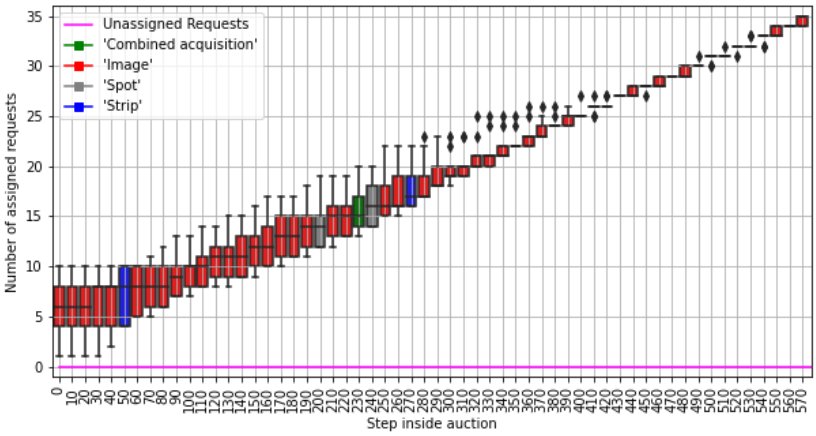



**Figure 5.17:** Box plot of load balance before/after auction per Satellite type - Null weight: Request satisfaction time term

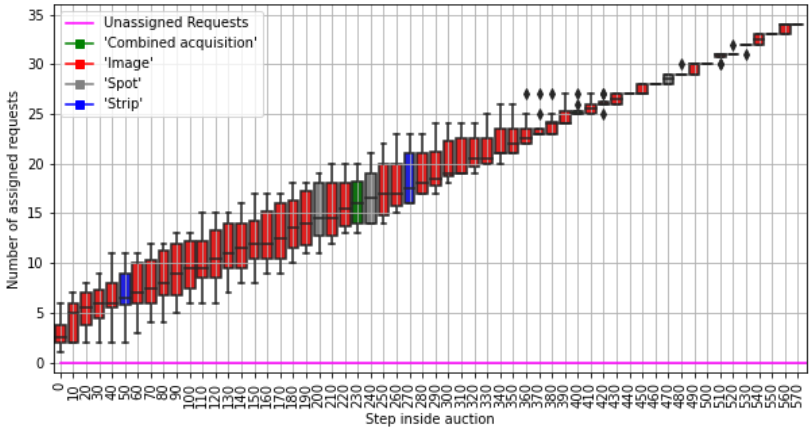
The behavior presented in the previous graph can be confirmed by the numerical values shown in Table 2, where the variance, unlike other experiments, does not exceed unity. However, a more detailed analysis should be performed since this term models a condition that allows giving preferences to those satellites that have more frequent contact with ground control unit, since it is common to observe in different experiments that the load balance term and conflicts between requests tend to have a higher contribution (about 80% of the maximum value) throughout the auction.

| Satellite type | Average before | Variance before | Average after | Variance after |
|---|---|---|---|---|
| Optical | 5.555556 | 7.580247 | 34.888889 | 0.098765 |
| Optical+SAR | 3.000000 | 2.500000 | 34.125000 | 0.109375 |
| SAR | 0.375000 | 0.234375 | 8.125000 | 0.609375 |

**Table 5.9:** Load balance comparison before/after auction per Satellite type - Null weight: Request satisfaction time term



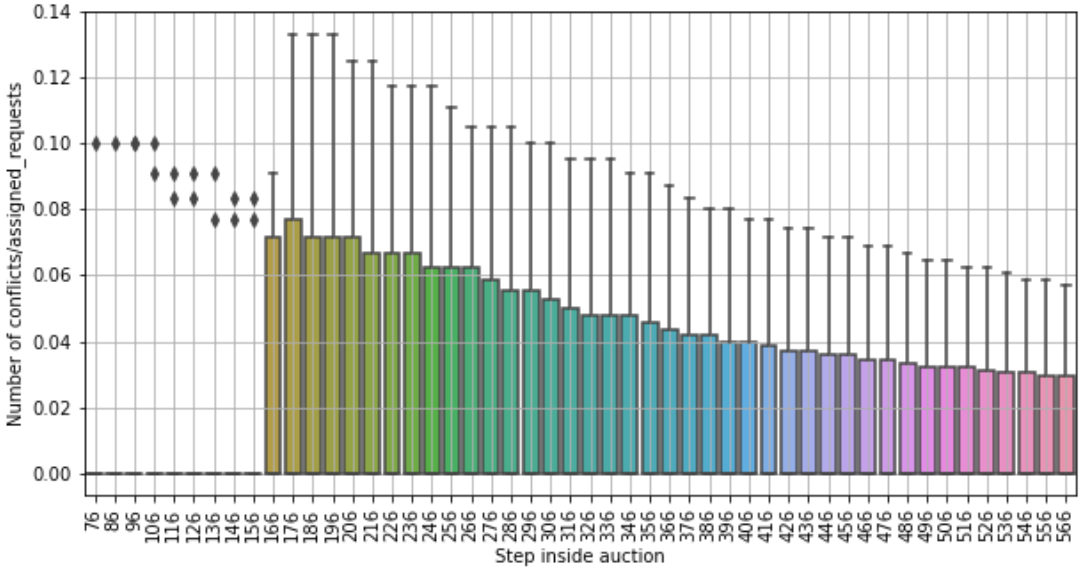

**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR

**Figure 5.18:** Box plot of load balance evolution per satellite type - Null weight: Request satisfaction time term

Figure 5.18 helps to confirm what was previously established regarding variance reduction, in this case for two types of satellites that tend to take a large percentage of the incoming requests for the services they have available on board. This large reduction in variance is explained by the fact that the contact term plays a restrictive role in the auction, because it forces to give preference to those satellites that have more frequent contacts with ground stations, therefore, when it has no weight the auction can more optimally balance the load among the proxies.




**(a)** Satellites types: Optical

**Figure 5.19:** Box plot of number of conflicts/maximum potential conflicts per satellite type - Null weight: Request satisfaction time term

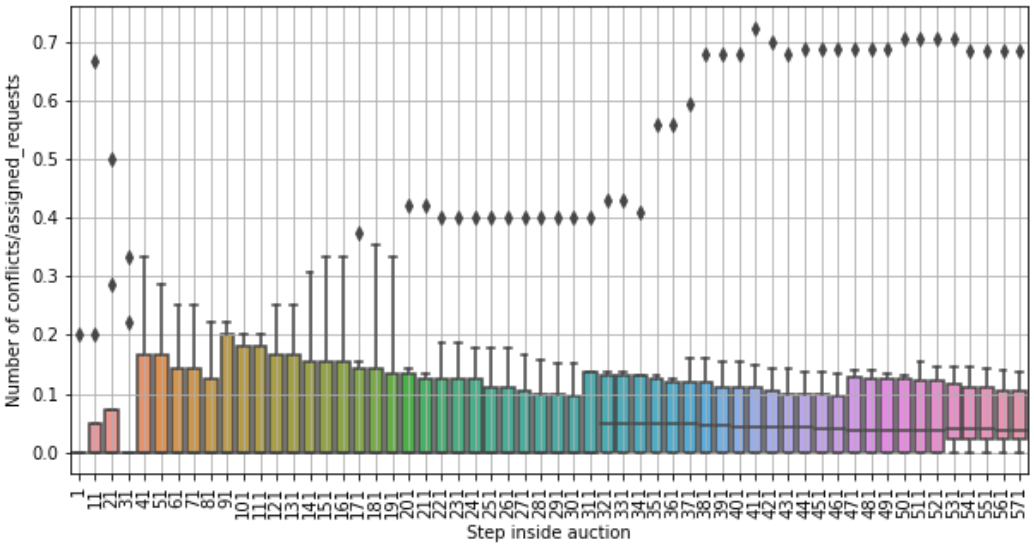
Figure 5.19 helps to corroborate that conflict reduction is still present under the conditions of this experiment. Therefore, as previously established, not necessarily the fact that there is a large reduction in variance makes these experimental conditions ideal, since this term plays a fundamental role in the auction, making it necessary to find a balance between the weight of this term in conjunction with the others. The latter in order to maintain the variance reduction feature while not subjecting the auction to highly restrictive conditions.

- **Null weight: Satellite resources**

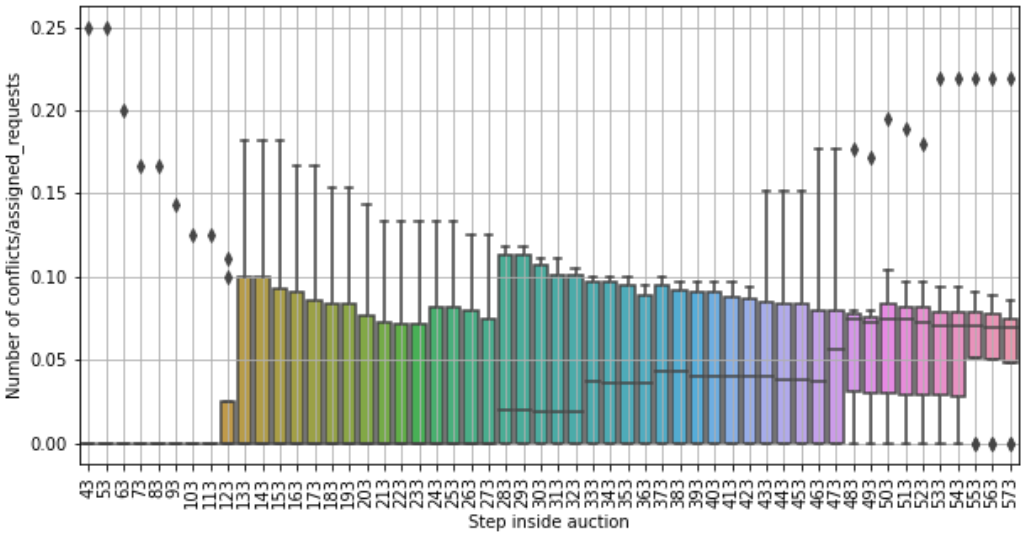
  The null term for this experiment is the satellite resources (resources term). The main effects on auction behavior is described below.

  For the conditions of this experiment, it is important to check that the conflict

management feature is preserved, since multiple conflicting assigned requests would cause a depletion of the satellite's resources. Figure 5.20 shows the reduction of the ratio of conflicts over potential assignable requests.




**(a)** Satellites types: Optical

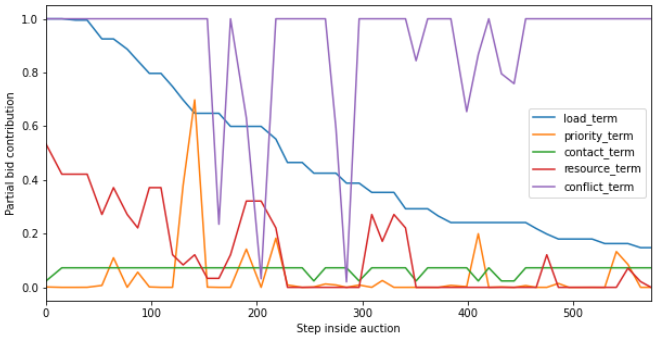



**(b)** Satellites types: Optical+SAR

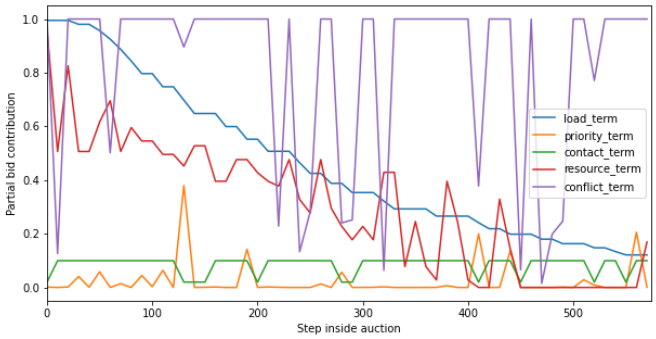
**Figure 5.20:** Box plot of number of conflicts/maximum potential conflicts per satellite type - Null weight: Satellite resources term

This resource parameter, besides playing a fundamental role in modelling the physical conditions of the satellite, needs to be better tuned with respect to

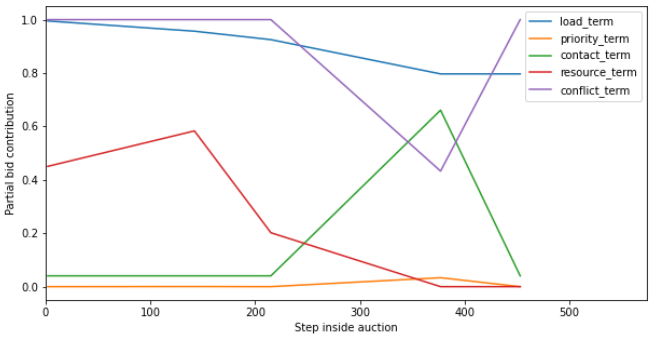
the other terms. The latter to reduce situations like the one depicted in Figure 5.21, in which, despite its low resource level, the auction continues to allocate a large number of requests to it instead of allocating a portion of these to Optical+SAR satellites that, for the first 300 auction steps, have an optimal amount of resources.




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR




**(c)** Satellites types: SAR

**Figure 5.21:** Stacked area plot and line plots per partial bid term of different satellite types - Null weight: Satellite resources term

- **Null weight: Satellite availability**

  This experiment is set up so that the null term is the satellite availability (conflict term). The following section describes the analysis of its impact on auction behavior.

  From previous experiments, it has been observed that the conflict term plays an inherent role in variance control because it adds a further constraint to the auction by controlling the cases of conflicts between multiple assignable requests. Therefore, as shown in Figure 5.22, a high variance environment is expected especially for satellites that could allocate a large number of incoming requests (Optical and Optical+SAR).

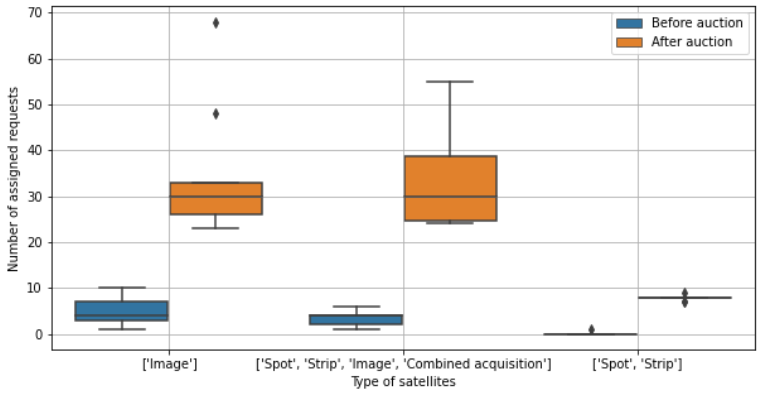


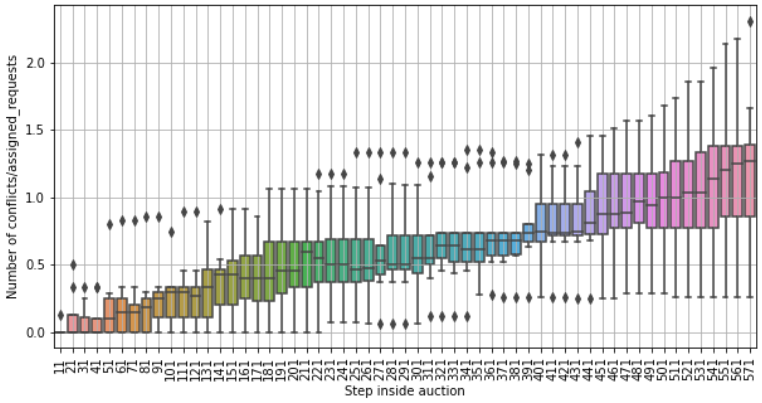

**Figure 5.22:** Box plot of load balance before/after auction per Satellite type - Null weight: Satellite availability term

The data in Table 5.10 confirms the previous statement by showing an increase in the variance of about 95%, and higher than in previous experiments where this term had a weight in the final bid.

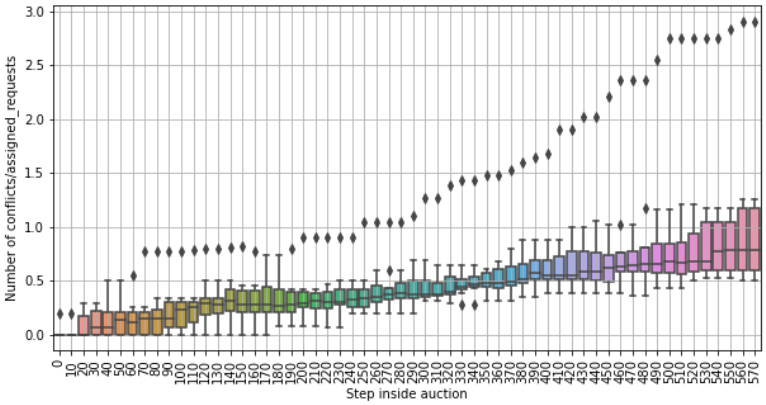
| Satellite type | Average before | Variance before | Average after | Variance after |
|---|---|---|---|---|
| Optical | 4.777778 | 7.950617 | 34.666667 | 189.777778 |
| Optical+SAR | 3.375000 | 2.234375 | 33.875000 | 119.859375 |
| SAR | 0.125000 | 0.109375 | 7.875000 | 0.359375 |

**Table 5.10:** Load balance comparison before/after auction per Satellite type - Null weight: Satellite availability term

The main behavior to be evaluated is how the ratio of conflicts over potential assignable requests evolves, since this term is the main conflict restrictor as the




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR

**Figure 5.23:** Box plot of number of conflicts/maximum potential conflicts per satellite type - Null weight: Satellite availability term

auction evolves. As can be seen in Figure 5.23, unlike other experiments, when the main conflict moderator is not present, there is an increasing trend as the auction progresses, even reaching the maximum conflict ratio of 2.73, for the case of Optical+SAR type satellites.

As a result, it can be established that this term plays a critical role in the management of potential conflicts in the auction, which is crucial for the proper operation of the satellite because, as previously stated, multiple conflicting assignments would damage and degrade the resources on board the satellite.

### 5.3.3 Predominant weights experiments

The following experiments are designed by giving a predominant weight of 0.4 to a specific term that composes the total bid and setting the others to 0.15. The type of experiment to be simulated determines the term with the highest weight value; this will help to assess the relevance of each term by analyzing the effects on auction behavior.

- **Predominant weight: Load balance**

  For this experiment, the term with the highest weight is the pending requests contribution (load term). The following section describes an analysis of the auction's effects.

  As expected and in contrast to the null weight experiment for this same term, Figure 5.24 shows low variance conditions and variance reduction before and after the auction. The latter is consistent with the predominance of this term since, as previously established, this is one of the main load drivers among the Proxies because it is directly linked to the variance.

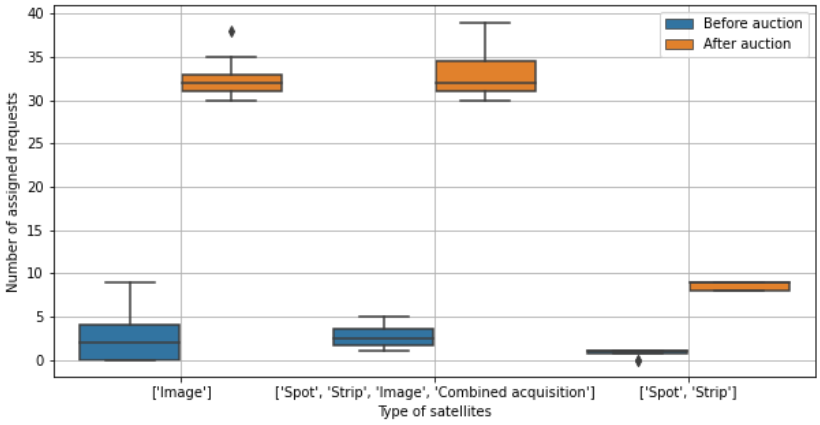



**Figure 5.24:** Box plot of load balance before/after auction per Satellite type - Predominant weight: Load balance term

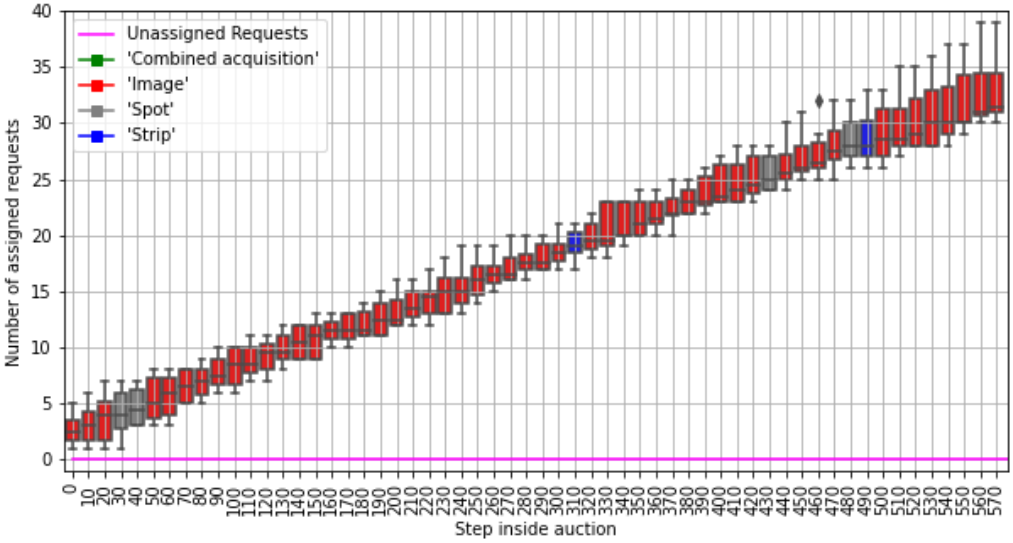
To validate the results shown in the previous chart, Table 5.11 shows the numerical values of the variance for this experiment. A significant reduction in variance can be seen for Optical satellites, while Optical+SAR satellites show a moderate increase in variance. And, as present in most experiments, SAR-type satellites maintain a low variance due to the low number of incoming requests requiring this service.

These results suggest that only giving predominance to the pending request term (load term) does not directly imply a reduction in variance for all cases. This term,

| Satellite type | Average before | Variance before | Average after | Variance after |
|---|---|---|---|---|
| Optical | 2.888889 | 10.543210 | 32.777778 | 5.283951 |
| Optical+SAR | 2.750000 | 2.187500 | 33.125000 | 8.359375 |
| SAR | 0.750000 | 0.187500 | 8.625000 | 0.234375 |

**Table 5.11:** Load balance comparison before/after auction per Satellite type - Predominant weight: Load balance term

as described previously in the null experiments, must be fine-tuned in conjunction with the request satisfaction time term (contact term) to achieve optimal variance reduction values, despite the fact that the values shown previously are considered to be in an adequate range.




**(a)** Satellites types: Optical+SAR

**Figure 5.25:** Box plot of load balance evolution per satellite type - Predominant weight: Load balance term

Figure 5.25 complements the previous analysis by showing the evolution of the number of requests assigned for an Optical+SAR satellite, which is expected to have a higher variance than other satellites due to the wide range of services it can assume. With this experiment, it is stated that for an appropriate reduction in variance, a joint tuning of the following terms is needed: pending requests, satisfaction time (crucial), and satellite availability (to a lesser extent).

- **Predominant weight: Request priority**

  For this experiment, the term with the highest weight is the request's priority. The analysis for this experiment is focused on the evolution of this term as the auction progresses.

  Figure 5.26 shows the evolution of this term for an Optical satellite. As in previous experiments, it can be observed that despite having a predominant weight with respect to the other terms, it continues not to have a major impact on the final bid. As previously established, it can be seen that this term depends on the characteristics of each request and not on the characteristics of the satellite itself.

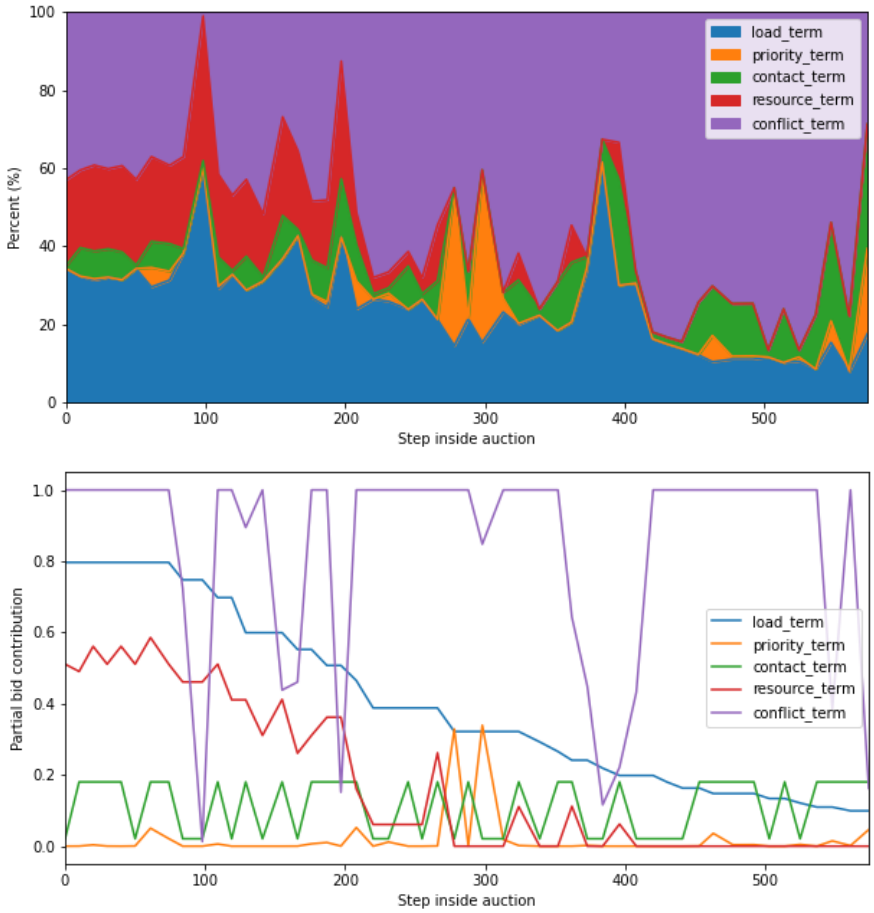



**Figure 5.26:** Stacked area plot and line plots per partial bid term of Optical satellite types - Predominant weight: Request priority term

  The evolution of this term is also presented for the case of a SAR type satellite, as shown in Figure 5.28.

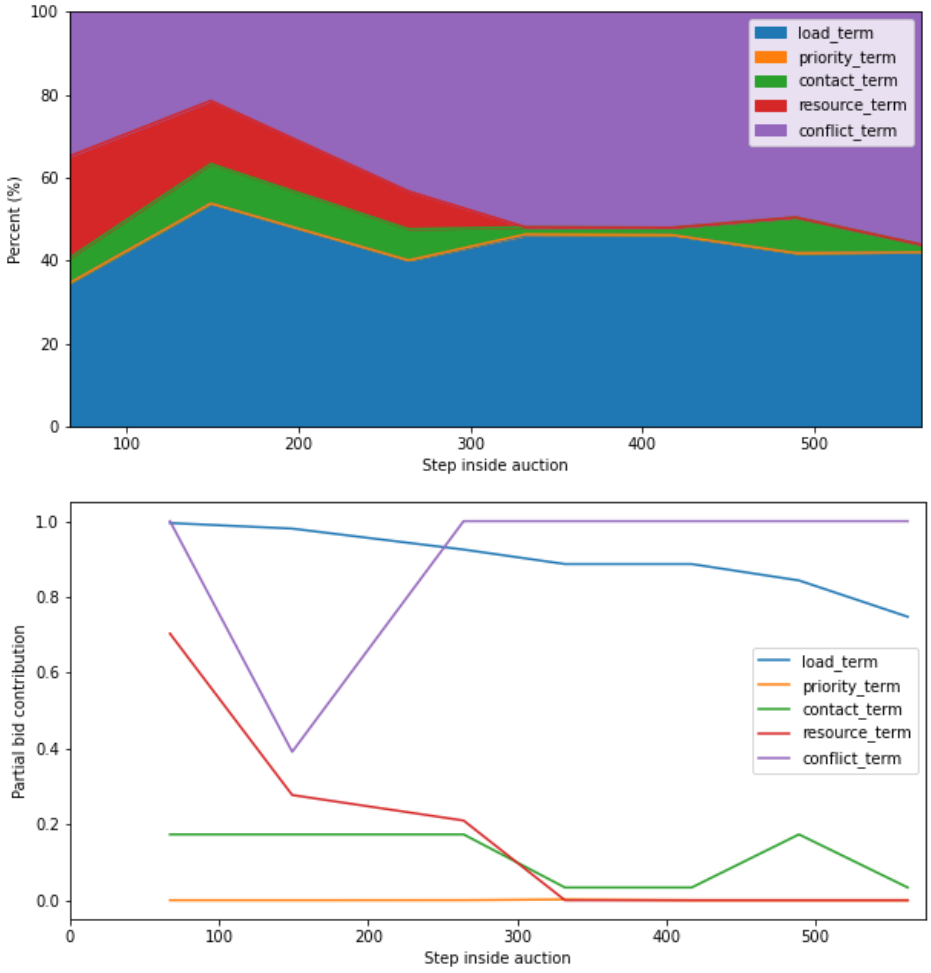



**Figure 5.27:** Satellites types: SAR

**Figure 5.28:** Stacked area plot and line plots per partial bid term of SAR satellite types - Predominant weight: Request priority term

It is observed that this term has almost no effect on the final bid. This type of behavior may occur due to the fact that the expiration time for SAR requests is twice as long as for other types of requests.

Therefore, an analysis should be performed using another auction modality, since for a single-item auction, the fact of announcing and assigning a single request affects the contribution of this term because its relevance lies in the comparison of multiple potentially assignable requests and the evaluation of them as a whole, depending on the characteristics of the requests themselves.

- **Predominant weight: Request satisfaction time**

  For this experiment, the term with the highest weight is the request's satisfaction time. The analysis for this experiment is focused on comparing the changes in variance before and after the auction with respect to the null experiment, as well as analyzing the evolution of the variance during the auction.

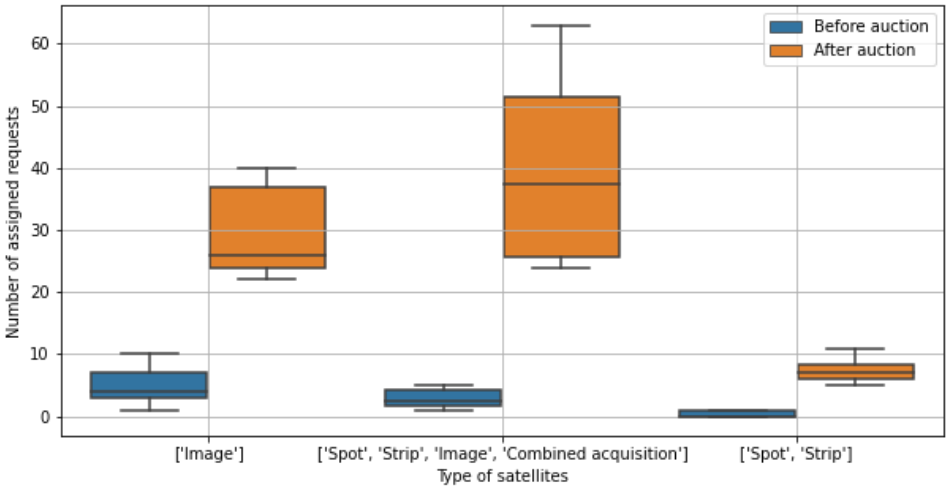
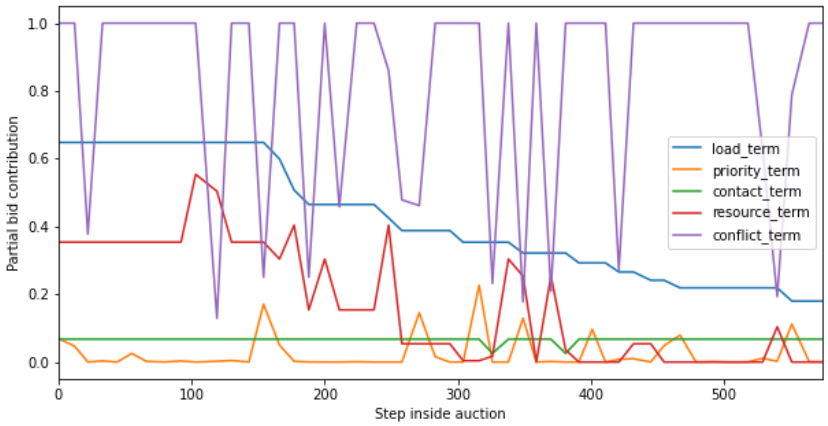



**Figure 5.29:** Box plot of load balance before/after auction per Satellite type - Predominant weight: Request satisfaction time term

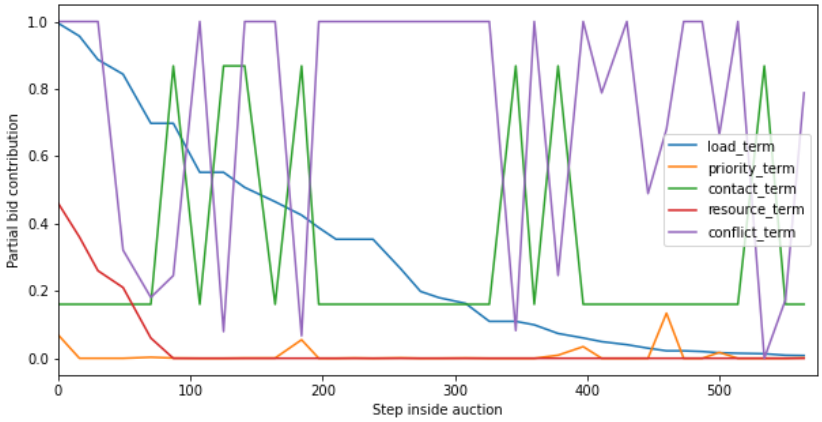
| Satellite type | Average before | Variance before | Average after | Variance after |
|---|---|---|---|---|
| Optical | 5.000000 | 9.555556 | 29.777778 | 44.172840 |
| Optical+SAR | 2.875000 | 2.359375 | 39.875000 | 214.859375 |
| SAR | 0.375000 | 0.234375 | 7.375000 | 3.234375 |

**Table 5.12:** Load balance comparison before/after auction per Satellite type - Predominant weight: Request satisfaction time term

From Figure 5.29 and the values in Table 5.12, it can be seen that, in the case of Optical+SAR satellites, the post variance increased significantly. This type of situation can occur because, by giving dominance to the time-to-satisfaction term, more restrictions are imposed on the distribution of tasks throughout the auction. As a result, the satellite that has an earliest contact with the ground stations after the time to target will be preferred regardless of whether the payload of the proxies is balanced, which is precisely the scenario that can be observed before.




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR

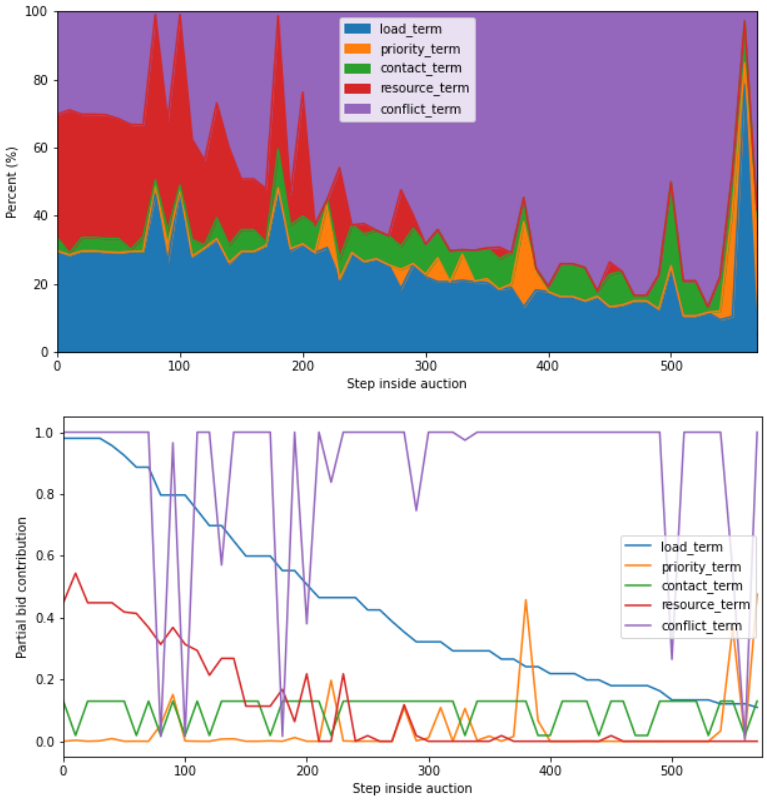
**Figure 5.30:** Stacked area plot and line plots per partial bid term of different satellite types - Predominant weight: Request satisfaction term

In addition, in Figure 5.30, it can be observed the comparison of the evolution of this term for Optical and Optical+SAR satellites. It should be noted that in the case of the latter, due to the introduction of a greater restriction due to the dominance of this term, the auction continues to add requests at a faster rate than in the case of Optical satellites, despite the fact that the latter deplete their resources before the 100 steps during the auction. This illustrates the situation mentioned before since it shows that the contact term for Optical+SAR has higher and more frequent peak bids than in the case of Optical satellites, and, therefore, these will be preferred for assignment to incoming requests, neglecting the relevance of the other terms.

- **Predominant weight: Satellite resources**

  For this experiment, the term with the highest weight is the satellite resources. The analysis for this experiment is focused on the evolution of this term as the auction progresses.

  In contrast to the null experiment for this same term, it can be seen in Figure 5.31 that despite giving dominance over the resource term, there was no major impact on the aforementioned behavior of continuing to add requests after the satellite has depleted its resources. But, it should be also take into account the autonomous decision that orbital_OLIVER would take to avoid these conditions. A possible evaluation could be further investigated on whether the resource term goes to zero slower than other cases.




**(a)** Satellites types: Optical+SAR

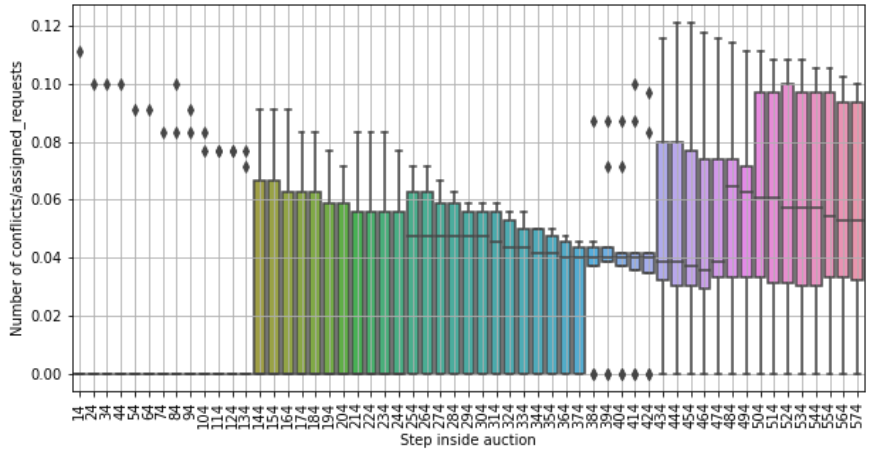
**Figure 5.31:** Stacked area plot and line plots per partial bid term of different satellite types - Predominant weight: Satellite resources term

This helps to validate that a fine-tuning of this term in conjunction with those that are directly linked to the number of requests to be allocated (i.e., the pending requests term) should be performed. It is also important to note that

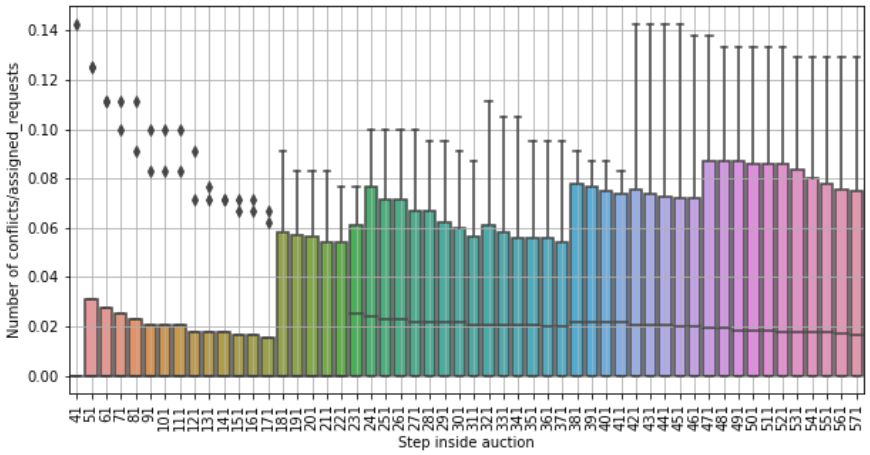
by giving dominance to a specific term, it does not imply an immediate effect on the behavior of the auction; fine tuning in conjunction with the other terms is required for the final bid to reflect what is expected.

- **Predominant weight: Satellite availability**

  For this experiment, the term with the highest weight is the satellite availability term (conflict term). The analysis for this experiment is focused on comparing the changes in reduction ratio of conflict over potentially asignable requests with respect to the null experiment, because it is the expected behaviour by giving dominance to this term.




**(a)** Satellites types: Optical




**(b)** Satellites types: Optical+SAR

**Figure 5.32:** Box plot of number of conflicts/maximum potential conflicts per satellite type - Predominant weight: Satellite availability term

As can be seen in Figure 5.32, for different types of satellites, the expected behavior of a reduction in the conflict ratio, as it shows a decreasing trend throughout the auction steps. Compared to other experiments, this one has the smallest conflict ratio (roughly 0.14) consistent with the type of experiment being conducted. However, it has been mentioned above that one of the main advantages of the model is an adequate conflict management; therefore, giving a high weight to a term that is directly linked to this characteristic is not useful since it is already within the conditions that are quite satisfactory.

# Chapter 6

# Conclusion and Future Work

This chapter summarizes the thesis's research results and proposes possible future developments.

This thesis developed a prototype solution for the task allocation problem in a constellation of autonomous satellites. Furthermore, the entire base software architecture was presented with the potential to run several simulations with different scenarios, various initial conditions, and satellite models of different types.

Simulation results have been used to provide a preliminary analysis of the feasibility, potential advantages, and general behavior of the proposed solution. A scalability analysis enabled a preliminary assessment of the model's complexity with different parameter configurations; the major findings are listed below:

1. The complexity of the model responded in a suitable way to the scaling in the number of proxies and incoming requests. Approximately linear complexity was evidenced in scaling experiments with up to 150 proxies and 3000+ incoming requests participating in an auction. This demonstrated the model's ability to be escalated without major time complexity concerns, especially in scenarios with a high number of interacting elements in the constellation.

2. The announcement and assignment size did not have a major impact on the time complexity, and for the different experiments performed, the linear dependence was preserved. The best results in time complexity (with an auction duration of 3.5 seconds) were obtained with an announcement size of 50 and, proportionally, an assignment size of 50.

3. The scalability experiments also reported that a large radius of the announcement size with respect to the assignment size leads to a longer duration in the auction. In terms of time complexity, this result is not desirable. However, future research can focus on experiments with larger ratios to examine the

quality of the bids and determine whether there is a trade-off between these two points.

Sensitivity analysis allows to examine the effects of different bidding strategy experiments on the auction behaviour. Different intrinsic aspects of the auction such as variance, the evolution of different bidding terms, conflict management and the evolution of the final bid were investigated. The main findings are listed below:

1. The pending requests term played a fundamental role in controlling the variance among the different proxies during the auction.

2. The request satisfaction time term played a role of constraining the satellite's willingness to assume new requests. Thus, in the case of a homogeneous bidding strategy, it is the primary factor causing a higher variance. Therefore, tuning in conjunction with the load balance term is required if the objective is to minimize the variance over the proxies participating.

3. The model exhibited a remarkable ability to manage the number of conflicts with respect to the number of potential assignable requests. The satellite availability term played a key role in the decreasing trend of the number of maximum conflicts during the early steps of the auction seen in the different experiments. Furthermore, a slowly and steadily increasing trend was observed, enabling to validate that the algorithm in the long term of the auction manages to distribute the requests among the proxies despite the fact that the probability of being in conflict increases with each new assigned request.

Nevertheless, future examinations could be performed using different auction modalities by varying the announcement and assignment size, since in this way it would be possible to further analyze the importance of the request priority term, which it could become more relevant when multiple requests are about to be auctioned. On the other hand, based on the results obtained, joint weight tuning experiments can be conducted to examine the consequences on the auctions and also on faulty scenarios to analyse the auction performance under such conditions.

Future work may also be directed towards the development of a complete simulation environment capable of simulating the operations of a constellation over time, with subsequent assignment processes and request executions. This work was focused on a single auction operation, enabling the extension to a full problem simulation with sequences of auctions.

# Appendix A

# Problem definition

## A.1 List of names

| | |
|---|---|
| $a$ | Auctioneer agent |
| $b_{j,i}$ | Bid of Proxy $p_j \in P$ for request $r_i \in R_a^k$ |
| $C$ | Contact Plan |
| $c$ | contact window |
| $\mathcal{CS}$ | Constellation System |
| $G$ | set of ground stations in $\mathcal{CS}$ |
| $g$ | ground station in $G$ |
| $\mathcal{GU}$ | Ground Tasking Unit |
| $ID_{cli}$ | Client ID |
| $ID_{req}$ | Request ID |
| $K$ | set of services offered by $\mathcal{CS}$ |
| $K_j$ | set of services offered by Proxy $p_j$ |
| $\mathcal{K}_j$ | function of $s_j$ that maps each $k \in K_j$ to the subset of $\Pi_j$ required to perform it |
| $k$ | service in $K$ |
| $L_j$ | set of resource level arrays in Proxy $p_j$ |
| $l_j^\rho$ | resource level array for resource $\rho$ in Proxy $p_j$ |
| $\hat{l}_j^\rho$ | upper bound of the resource level for resource $\rho$ in Proxy $p_j$ |
| $l_{j,t}^\rho$ | resource level for resource $\rho$ in Proxy $p_j$ at time $t$ |
| $m$ | announcement size |
| $N$ | length of the resource prediction horizon (number of time steps) |
| $N_b$ | number of bidding terms used to compute the bid |
| $P$ | set of Proxy agents |
| $p$ | Proxy in $P$ |
| $R$ | set of unassigned client requests in $a$ |

| | |
|---|---|
| $R_j$ | set of pending client requests assigned to Proxy $p_j$ |
| $\tilde{R}_{j,i}$ | set of pending client requests assigned to Proxy $p_j$ in conflict with $r_i$ |
| $\hat{R}^k$ | set of assigned client requests during the k-th auction round |
| $R_a^k$ | set of requests that are put up to auction in the k-th round |
| $r$ | client request |
| $ROI$ | Region Of Interest for a request |
| $S$ | set of satellites in $\mathcal{CS}$ |
| $s$ | satellite in $S$ |
| $t$ | continuous time or time step |
| $t^{exp}$ | expiring date of a request |
| $t^{sub}$ | submission date of a request |
| $t_0$ | date of the current auction |
| $t_C$ | contact plan horizon |
| $t_e$ | end time of contact window |
| $t_{ROI}^j$ | time to target: time required by $s_j$ to reach the region of interest $ROI$ |
| $t_s$ | start time of contact window |
| $W_j$ | bidding strategy of Proxy $p_j$ |
| $w_{j,n}$ | weight applied to the bidding term $\gamma_n$ by Proxy $p_j$ when computing the bid |
| $\alpha$ | priority level of request, ranges in [0, 1] |
| $\gamma_n$ | n-th bidding term that contributes to the bid |
| $\Delta_j$ | function of $s_j$ that maps each $k \in K_j$ to the time required to perform the service |
| $\Pi_j$ | set of payloads available onboard satellite $s_j$ |
| $\rho$ | tracked satellite resource |
| $\sigma_{s,t}$ | status of satellite $s$ received at time $t$ |

# Bibliography


[1] AIKO s.r.l. *orbital_OLIVER*. URL: https://www.aikospace.com/whitepaper/AIKOOLIVERWhitePaper.pdf (cit. on pp. 2, 10).

[2] Alessandro Benetton, Christian Cardenio, Lorenzo Feruglio, Riccardo Maderna, Gabriele Giordana, Temenuzhka Avramova, and Mattia Varile. «MiRAGE: AI-based onboard autonomy enabling the future of satellite operations». In: Oct. 2021 (cit. on pp. 2, 10–12).

[3] Thomas Uhlig, Florian Sellmaier, and Michael Schmidhuber, eds. *Spacecraft Operations*. Springer Vienna, 2015. DOI: 10.1007/978-3-7091-1803-0. URL: https://doi.org/10.1007/978-3-7091-1803-0 (cit. on pp. 4–7).

[4] P. Fortescue, G. Swinerd, and J. Stark. *Spacecraft Systems Engineering*. Wiley, 2011. ISBN: 9781119978367 (cit. on p. 5).

[5] Sean Augenstein, Alejandra Estanislao, Emmanuel Guere, and Sean Blaes. «Optimal scheduling of a constellation of earth-imaging satellites, for maximal data throughput and efficient human management». In: *Proceedings of the International Conference on Automated Planning and Scheduling*. Vol. 26. 2016, pp. 345–352 (cit. on p. 7).

[6] James Boerkoel, James Mason, Daniel Wang, Steve Chien, and Adrien Maillard. «An Efficient Approach for Scheduling Imaging Tasks Across a Fleet of Satellites». In: *International Workshop on Planning Scheduling for Space (IWPSS). https://ai. jpl. nasa. gov/public/papers/Boerkoel_IWPSS2021_paper_23. pdf*. 2021 (cit. on p. 7).

[7] Vishwa Shah, Vivek Vittaldev, Leon Stepan, and Cyrus Foster. «Scheduling the world's largest earth-observing fleet of medium-resolution imaging satellites». In: *International Workshop on Planning and Scheduling for Space*. Organization for the 2019 International Workshop on Planning and Scheduling ... 2019, pp. 156–161 (cit. on p. 7).

[8] Feng Yao, Jufang Li, Baocun Bai, and Renjie He. «Earth observation satellites scheduling based on decomposition optimization algorithm». In: *International Journal of Image, Graphics and Signal Processing* 2.1 (2010), p. 10 (cit. on p. 8).

[9] Fabian Lang, Andreas Fink, and Tobias Brandt. «Design of automated negotiation mechanisms for decentralized heterogeneous machine scheduling». In: *European Journal of Operational Research* 248.1 (2016), pp. 192–203 (cit. on p. 8).

[10] Ping-an Gao, Zi-xing Cai, and Ling-li Yu. «Evolutionary computation approach to decentralized multi-robot task allocation». In: *2009 Fifth International Conference on Natural Computation*. Vol. 5. IEEE. 2009, pp. 415–419 (cit. on p. 8).

[11] Adam Herrmann, Joao Vaz Carneiro, and Hanspeter Schaub. «REINFORCEMENT LEARNING FOR THE MULTI-SATELLITE EARTH-OBSERVING SCHEDULING PROBLEM». In: () (cit. on p. 8).

[12] Chris Daehnick, Isabelle Klinghoffer, Ben Maritz, and Bill Wiseman. «Large LEO satellite constellations: Will it be different this time». In: *McKinsey and Company* 4 (2020) (cit. on p. 8).

[13] Mohamed Khalil Ben-Larbi et al. «Towards the automated operations of large distributed satellite systems. Part 1: Review and paradigm shifts». In: *Advances in Space Research* 67.11 (2021), pp. 3598–3619 (cit. on pp. 8, 9).

[14] Rebecca Castano et al. «Operations for autonomous spacecraft». In: *2022 IEEE Aerospace Conference (AERO)*. IEEE. 2022, pp. 1–20 (cit. on p. 9).

[15] Zixuan Zheng, Jian Guo, and Eberhard Gill. «Onboard mission allocation for multi-satellite system in limited communication environment». In: *Aerospace Science and Technology* 79 (2018), pp. 174–186 (cit. on pp. 10, 12, 13).

[16] Kelin Jose and Dilip Kumar Pratihar. «Task allocation and collision-free path planning of centralized multi-robots system for industrial plant inspection using heuristic methods». In: *Robotics and Autonomous Systems* 80 (2016), pp. 34–42 (cit. on p. 12).

[17] Feng Yao, Jiting Li, Yuning Chen, Xiaogeng Chu, and Bang Zhao. «Task allocation strategies for cooperative task planning of multi-autonomous satellite constellation». In: *Advances in Space Research* 63.2 (2019), pp. 1073–1084 (cit. on p. 13).

[18] Han-Lim Choi, Luc Brunet, and Jonathan P. How. «Consensus-Based Decentralized Auctions for Robust Task Allocation». In: *IEEE Transactions on Robotics* 25.4 (2009), pp. 912–926. DOI: 10.1109/TRO.2009.2022423 (cit. on p. 13).

[19] William E Walsh and Michael P Wellman. «A market protocol for decentralized task allocation». In: *Proceedings International Conference on Multi Agent Systems (Cat. No. 98EX160)*. IEEE. 1998, pp. 325–332 (cit. on p. 13).

[20] Michele Atkinson. «Results analysis of using free market auctions to distribute control of UAVs». In: *AIAA 3rd" Unmanned Unlimited" Technical Conference, Workshop and Exhibit*. 2004, p. 6558 (cit. on p. 13).

[21] M. Wooldridge. *An Introduction to MultiAgent Systems*. Wiley, 2009. ISBN: 9780470519462. URL: https://books.google.it/books?id=X3ZQ7yeDn2IC (cit. on pp. 13–15, 17, 23).

[22] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. ITPro collection. CogNet, 1999. ISBN: 9780262731317 (cit. on pp. 14, 16–18, 23, 24).

[23] Michael N Huhns and Larry M Stephens. «Multiagent systems and societies of agents». In: *Multiagent systems: a modern approach to distributed artificial intelligence* 1 (1999), pp. 79–114 (cit. on pp. 18–21, 23).

[24] Tuomas Sandholm. «An implementation of the contract net protocol based on marginal cost calculations». In: (cit. on p. 24).

[25] Bernardo A Huberman and Scott H Clearwater. «A Multi-Agent System for Controlling Building Environments.» In: *ICMAS*. 1995, pp. 171–176 (cit. on p. 24).

[26] Brian P Gerkey and Maja J Mataric. «Sold!: Auction methods for multirobot coordination». In: *IEEE transactions on robotics and automation* 18.5 (2002), pp. 758–768 (cit. on p. 25).

[27] Craig Tovey, Michail G Lagoudakis, Sonal Jain, and Sven Koenig. «The generation of bidding rules for auction-based robot coordination». In: *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III*. Springer, 2005, pp. 3–14 (cit. on p. 32).

[28] Michail G Lagoudakis, Evangelos Markakis, David Kempe, Pinar Keskinocak, Anton J Kleywegt, Sven Koenig, Craig A Tovey, Adam Meyerson, and Sonal Jain. «Auction-Based Multi-Robot Routing.» In: *Robotics: Science and Systems*. Vol. 5. Rome, Italy. 2005, pp. 343–350 (cit. on p. 33).

[29] Liu Lin and Zhiqiang Zheng. «Combinatorial bids based multi-robot task allocation method». In: *Proceedings of the 2005 IEEE international conference on robotics and automation*. IEEE. 2005, pp. 1145–1150 (cit. on p. 34).