UNIVERSITY OF SOUTHAMPTON

Market-based Task Allocation in Distributed Satellite Systems

by

Johannes Gerhardus van der Horst

A thesis submitted in partial fulfilment for the degree of Doctor of Philosophy

in the
Faculty of Physical and Applied Sciences
Electronics and Computer Science

March 2012

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF PHYSICAL AND APPLIED SCIENCES
ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

Market-based Task Allocation in Distributed Satellite Systems

by Johannes Gerhardus van der Horst

This thesis addresses the problem of task allocation in a distributed satellite system. These spacecraft specialise in different functions, and must collaborate to complete the mission objectives. The energy available for task execution and communication is, however, extremely limited, which poses a challenging design problem. I propose the use of a market-based, multi-agent approach to achieve the necessary macro-level behaviour. The development and verification of this allocation mechanism constitutes the first major objective of this thesis.

Although numerous examples of task allocation in related systems exist, I found a worrying disconnect between our general, theoretical knowledge of task allocation, and the specific application thereof. General analyses of abstracted task allocation exist, and specific implementations have been constructed in a heuristic way, but very little work navigates between these two extremes. My second major objective therefore contributes to mapping the problem space.

The proposed task allocation mechanism is based on human labour markets in order to obtain similar robustness and flexibility. It uses fully distributed auctions to efficiently allocate tasks in volatile networks, without any global knowledge of the system state. The energy required for communication is constant, irrespective of the size of the network, resulting in a highly scalable allocation mechanism.

To find the area in parameter space where market-based control is the more suitable solution, when compared to a centralised approach, I characterised the allocation mechanism in terms of network size, node failure rate, and robustness. The relationship between communication cost and topology is explored by looking at the overheads associated with different static topologies, and the impact of communication distance. The ability of the allocation mechanism to cope with realistic Keplerian dynamics is also confirmed. Finally, I investigate the difference in performance between the allocation mechanism, as an example of a cooperative market, and a competitive scenario where adaptive agents compete to maximise their revenue. Results show that competitive markets are subject to positive feedback loops which can result in inferior performance for sparsely connected and heavily loaded networks.

This exploration of the system parameters is treated as a traversal of the problem space, resulting in an emergent taxonomy of both problem and solution elements.

Declaration of Authorship

I, Johannes Gerhardus van der Horst, declare that the thesis entitled

Market-based Task Allocation in Distributed Satellite Systems

and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as van der Horst *et al.* (2009); van der Horst and Noble (2010, 2011, 2012)

Signed:

Date:

Contents

List of Figures

Nomenclature

Acronyms

AI	Artificial intelligence
CI	Centralised with immunity
EO	Earth observation
GPS	Global positioning system
HPC	High-performance computing
LEO	Low-earth orbit
MANET	Mobile ad-hoc network
MAS	Multi-agent system
MBC	Market-based control
WSN	Wireless sensor network

Symbols

a	Length of the semi-major axis
α	Task transfer cost scaling factor
B	Bid value
B_{rx}	Bid value received by auctioneer
B_{cc}	Worker fitness
c_{int}	Internal task execution cost
c_{os}	Outsourcing cost
$\hat{c}_{\mathrm{os}}$	Outsourcing cost estimate
c_{status}	Cost of a single status update for centralised control
c_{tx}	Transmission cost
c_{tf}	Task transfer cost
c_{cc}	System cost, centralised task allocation
c_{mbc}	System cost, market-based task allocation
c_{alloc}	Allocation cost
$c_{\mathrm{alloc_{total}}}$	Total allocation cost
c_{auc}	System-wide cost of running one auction
d_{alloc}	Allocation distance
d_{bid}	Bid distance
d_{tf}	Task transfer distance

d_{ttl}	Time-to-live distance (hops)
Δ_{μ}	Profit margin adjustment
Δ_{os}	Outsourcing cost estimate adjustment
E_{inc}	Incoming energy
e_{rem}	Remaining energy
e_{max}	Maximum energy
γ_{os}	First-order infinite impulse response filter coefficient
k	Commission
λ	Failure rate
n	Number of nodes in the system
n'	Number of nodes in the auction community
n_k	Number of task components
n_{os}	Outsource adjustment time-out
n_{status}	Number of status packets
n_t	Number of tasks
P	Orbital period
P_{tx}	Transmission power
$P_{\text{tx}_{\text{max}}}$	Maximum transmission power
P_{rx}	Received power
P_{th}	Minimum P_{rx} for successful receipt
μ	Profit margin for competitive agents
R_{th}	Communication range
r	Distance between nodes
τ_{status}	Status update (centralised allocation)
z	Task size
θ	True anomaly
u	Cartesian position of a satellite
u_{ref}	Position of point on reference orbit
Ω	Right ascension of the ascending node (RAAN)
ω	Argument of perigee

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Jason Noble, for his guidance and help and my colleagues in the SENSe and IAC groups for their diverse insights from different fields. Finally, I wish to thank my family and my partner for their constant support and encouragement.

1 Introduction

This thesis is about coordinating a group of satellites that need to collaborate to complete mission objectives. It may come as a surprise then, that my discussion does not start with satellite design or orbital mechanics, but rather with the work of Adam Smith on markets, because the challenge of managing such a group of satellites is fundamentally a labour allocation problem.

In Adam Smith's *Wealth of Nations* he identifies the division of labour as the driving force behind increased productivity in economies. Specialisation also results in greater interdependence: individuals in the economy have to rely on others in their vicinity to supply the skills they don't have (Slater and Tonkiss, 2001, Chapter 1). This leads us to the problem of the *allocation* of labour: given the choice of a number of potentially suitable specialists, which one is best suited to the task that needs allocating? Even if they are identical in terms of skills, factors such as location, availability, reputation and coordination effort can influence the decision. This is, in a nutshell, what this thesis investigates. Given a dynamic community of specialised individuals and a stream of incoming work, what system can we devise that would allocate tasks in an efficient and effective manner?

Although an economy is perhaps the most obvious example, it is by no means the only system where the allocation of tasks is a challenge. Social insects, such as ants, termites and bees, display specialisation into different castes with different roles. Tasks are clearly not allocated in any centralised way, and yet an effective division of labour is achieved through the parallel activities of largely autonomous individuals and the use of rather simple communication mechanisms, e.g., for the recruitment of foragers to a new food source. The colony as a whole can thus be viewed as a superorganism: an intelligent agent in its own right with lifespan and abilities far exceeding that of its component individuals (Oster and Wilson, 1979).

While the above examples display distributed coordination, many human approaches to task allocation, such as governments and military organisations, rely on centralised or hierarchical command structures. In these systems knowledge is transmitted to and analysed by a central authority — or multiple points in the case of a hierarchical system — before the optimal course of action is decided upon. Individuals are of limited autonomy and have tasks assigned to them by a superior who is assumed to be in a more information-rich position.

The above systems vary in their structure and implementation, but they all have task allocation as a core function, and as such have served as models for designers facing similar allocation problems in the construction of technological systems. The centralised approach has traditionally dominated in this sphere; it may be that the idea of a single mind or coordinator fits most comfortably into the patterns of Western thought (Resnick, 1997). In recent years interest in decentralised systems has increased significantly with the hope that they can provide us with ways of coping with our own increasingly complex and interdependent technological systems. Interest in market-based coordination (Smith, 1980; Huberman, 1988), swarm intelligence (Bonabeau *et al.*, 1999) and ant colony optimisation (Dorigo *et al.*, 1996) has been growing steadily since the 1980s. These techniques are not only useful for solving existing problems: they can also provide us with control methodologies for systems that have yet to be realised, where scalability, robustness and self-organisation are highly desirable characteristics.

1.1 The problem

This study is primarily motivated by the challenge of managing one such technological system; namely distributed satellites. Spacecraft have traditionally consisted of monolithic structures. A positive feedback loop has driven these designs to grow ever more complicated: to maximise value and reliability of already expensive projects, progressively more complex designs were developed. This in turn increased development time, which further increased cost. To counter this trend, the use of a number of modular, free-flying spacecraft has been suggested (Brown *et al.*, 2006; Barnhart *et al.*, 2007). Costs are kept down by keeping the spacecraft simple; this in turn means they can only complete mission objectives by collaborating. Some of the spacecraft provide infrastructure, such as data processing or communication with the ground station, while others fulfil payload functions, e.g., earth observation.

This approach offers numerous benefits, most of which translate to a greater return on the money invested. The system as a whole is made more robust by using redundant units, eliminating single points of failure and thus providing graceful degradation when component satellites fail. Resource sharing increases efficiency by optimally utilising redundancy in the system. Risk can be lowered by using multiple launches to deploy the system. Very large formations can be formed in orbit, providing much larger structures than are feasible using conventional approaches, which allow novel approaches to sensing. Significant progress is being made in miniaturisation (Vladimirova *et al.*, 2006; Barnhart *et al.*, 2006) and formation flight (Mueller and Thomas, 2005; Ferguson and How, 2003) but a number of challenges still remain, including the question of how to manage such an organisation of spacecraft.

Mission objectives can be decomposed into tasks that are executed by different satellites. These tasks need to be allocated efficiently, while taking satellite capabilities, their limited power and constrained communication into account. Allocation is further complicated by changes in the group topology due to orbital mechanics, as well as the possibility of failure of individual nodes. The scale of the system, ranging from tens to hundreds of satellites, may necessitate spacecraft autonomy, but the system must still reliably complete the desired mission objectives.

The complexity of this problem suggests the use of techniques based on biological or social complex systems. Market-based mechanisms offer a promising solution to this allocation problem, in particular, the model of a labour market seems very well-suited. In both distributed satellite systems and real-world markets, individuals have to deal with limited information, changing communication topologies, and spatially distributed agents. In addition, the robustness gained from having no central controller and the adaptive nature of the market are extremely attractive properties for a satellite application. With this approach, component satellites bid for jobs that they are capable of completing. By using their energy levels to calculate the bid value, they communicate their relative fitnesses to the allocating satellite. Topological information is added to the bid price by the spacecraft that relay bids, thereby helping to localise allocation and minimize the amount of energy spent on communication.

This problem of how to manage such a system is what originally attracted me to this area. However, I soon came to realise that the field of multi-agent systems is not yet at the stage where a reliable, over-arching methodology has been defined and validated, resulting in an alarming proliferation of systems that are of relatively little use to others facing similar problems (I was by no means the first one to come to this conclusion; see e.g., Chapter 10, Wooldridge, 2002). This led me to view my design process as part of the exploration of a greater problem space. By understanding the implications of design decisions and constraints, we can relate different systems in this space, thereby contributing to the development of a responsible methodology for the field.

1.2 Traversing the problem space

The challenge of task allocation in a distributed satellite system sounds like a traditional engineering problem: a number of physical and technical constraints need to be taken into account by making a series of design decisions to satisfy a set of operational requirements. One approach, popular in practice in the field of multi-agent systems, would be to treat this as effectively a stand-alone problem. We begin with such theory as is available, and proceed guided by intuition and experiment until we have a system that works well enough. The pressure to produce a working system often precludes efforts to relate the particular case to a more general theory of multi-agent systems.

But what happens if a similar, but slightly different problem needs to be solved? Intuition suggests that a significant portion of one solution should be applicable to closely related problems. But which portions? Will the design decisions that had a satisfactory outcome in one context necessarily do the same in another?

In surveying the literature relevant to the problem of task allocation in distributed satellite systems, a large number of closely related applications came to light. Spanning the fields of mobile robotics, distributed computing, wireless sensor networks and logistics, these applications all share some characteristics with the problem at hand. Given this multitude of specific cases, one would expect to find a layer of work relating these solutions and abstracting them into a generalised set of design guidelines and tools to better understand problems and engineer applications. This space is, however, remarkably empty.

Of course the field of multi-agent systems is not devoid of theory, but the most successful theoretical contributions are extremely abstract and concerned with underlying principles, e.g., rational choice theory, game theory and logic. However, mappings from abstract algorithms and mathematical performance proofs to actual implementations that take into account the messiness of the real world are few and far between.

Thus the field has a bimodal distribution of work, some of it very specific and some of it very general. This is graphically represented in Figure 1.1. The management of data in the distributed databases system *Mariposa* (Stonebraker *et al.*, 1994) is an example of a specific problem, while a microeconomic explanation about the optimality of markets would be a corresponding entry on the general side of the spectrum. We could describe work that progresses from top to bottom as science, while moving in the opposite direction, from the general to the specific, is engineering.

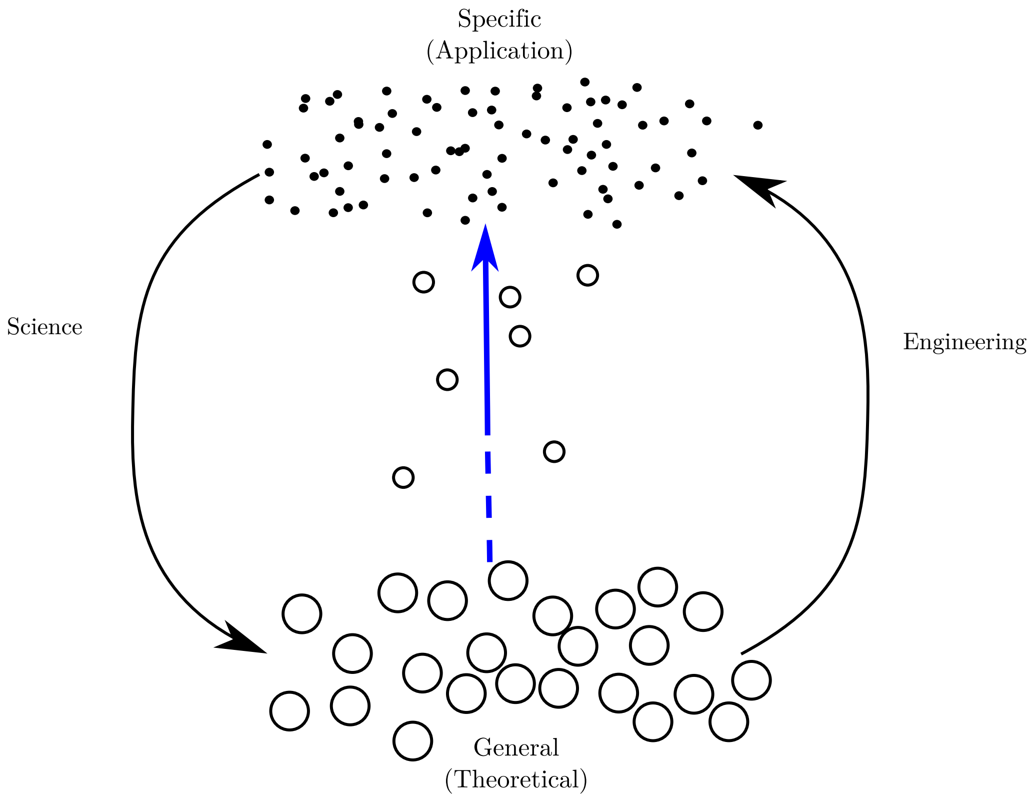


Figure 1.1: Work related to task allocation in multi-agent systems can broadly divided into two categories: application specific engineering problems (small circles), and abstract, general theories and proofs (large circles). The space between these poles is quite empty, apart from some taxonomies and a few attempts at agent design methodologies: we still do not really know how to design these systems. The work in this thesis traverses a section of this space, as indicated by the blue arrow. This allows the design process to be utilised as a way of exploring this space, as shown in more detail in Figure 1.2.

The central area is not completely empty; work that falls into this section includes some taxonomies (Dudek *et al.*, 1996; Gerkey and Matarić, 2004; Lau and Zhang, 2003) that seek to generalise from specific problems, as well as attempts that start from the theoretical end, e.g. Wooldridge (2002) and Ehrentreich (2007). These efforts can be visualised as a slow vertical creep from both the lower and upper clusters, in the hope that they will eventually meet in the middle. The sparsity of the central zone implies that we still don't really know how to construct a good multi-agent system for a specific problem. The vast majority of currently constructed systems appear to be based on what the designers know, which intuitive approaches are naturally suggested by the problem, what is currently fashionable, and what is readily fundable. This approach does not contribute as much as it could to developing a principled design methodology. However, a design methodology for such a large and diverse problem space can not be developed overnight; instead it must be preceded by diligent mapping of relationships between problems and solutions, and other closely related problems and solutions. This map forms a necessary first step towards the design methodologies that we need.

1.2.1 The design process

In designing a task allocation process for a distributed satellite system a portion of the space in Figure 1.1 needs to be traversed, as represented by the marked arrow. Figure 1.2 displays an abstraction of the process in more detail. An initial model is constructed by drawing on available theory (e.g., social models of task allocation, theory of auctions, etc.).

This model is general and can be applied to a number of related problems in the field. The design process involves refining it by focusing in on a specific problem — we take into account successively more detailed constraints and commit to favoured design decisions at each stage. If we traverse Figure 1.2 from bottom to top, the breadth of our coverage narrows and we see that whole families of systems are progressively eliminated from our consideration. If we were to follow any one of these branches, we would reach a different point in problem space. One of the outcomes of this approach is thus an emergent taxonomy of closely related problems and suitable solutions.

The targeted problem of task allocation in distributed satellite systems is very much a point on the specific side of this spectrum. I believe a significant contribution can be made to the sparsely populated space in the middle of Figure 1.1 by the design process outlined above. Even though this thesis works up to a specific application, the contribution of significance to the wider community is through exploring a problem space.

1.3 Objectives and overview

In the discussion thus far, I have identified the two major objectives of this thesis, namely:

1. to develop a market-based task allocation solution for managing a distributed satellite system

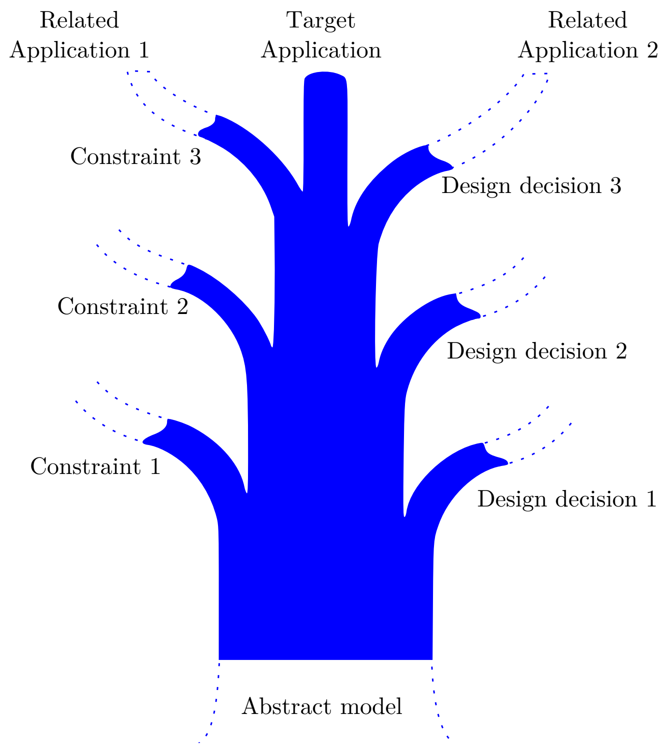


Figure 1.2: The design process can provide an emergent taxonomy of related problems. As we refine a general model to be more specifically applicable, we make design decisions and take constraints into account. These branches can lead to different, but related points in the problem space.

2. to position my specific solution in a larger problem space describing the allocation of complex tasks in dynamic environments with costly communication.

The first objective is primarily a design and engineering problem, but I believe the design process can be used to help bridge the gap between our abstract models of task allocation and the numerous applications where it is required — this will be discussed in more detail in Chapter 2. We can now proceed to a detailed discussion of how these objectives are met.

In Chapter 2, I explore the background literature relevant to this study. An overview of the current state of distributed satellite systems is followed by a discussion of task allocation in multi-agent systems. An important point noted in this chapter is that work on task allocation in systems consisting of multiple interacting agents seems to fall in either the abstract or specific camps in Figure 1.1, with relatively little work falling between them. Completing my second objective above is synonymous with providing analyses at intermediate levels of abstraction.

The specific design problem, namely task allocation in a multi-satellite system, is addressed in Chapter 3. I develop a reference mission which allows us to clearly identify the constraints on the system and the requirements for the task allocation mechanism. An abstracted model of a human labour market is used to develop an allocation model. This model is then mapped back to the satellite problem, to define the market-based allocation model that is used extensively throughout the rest of the thesis.

The behaviour of the market-based model is explored in more detail in Chapter 4. By working through the allocation process and testing it on simplified problems, we verify that the system is indeed well-suited to the problem at hand. The simplicity of the scenarios tested facilitates understanding of how and why this approach is successful.

The subsequent three chapters are all concerned with what happens when we change major parameters in the system. This realises the methodological approach explained in Section 1.2.1. Essentially, what I am working towards is a characterisation of which parts of a solution could stay constant and which parts would have to change as we move along various axes in problem space.

Specifically, Chapter 5 describes the process used to find the space where decentralised market-based control should be preferred above using a central allocator, and vice versa. It is widely assumed that centralised allocation is better suited to small systems, while distributed approaches make more sense in large systems. Distributed satellite systems, however, fall between these two extremes, with neither approach obviously superior. In this chapter I explore this grey area by using an analytical description of the task allocation cost in combination with simulation. The focus is not on proving the market-based task allocation approach absolutely superior, but rather to acknowledge that different approaches are required for different regions in the problem space.

The role of network topology is investigated in more detail in Chapter 6, as it has a significant impact on task allocation success and the resulting allocation overhead. I measure the costs associated with different spatial topologies. I then explore the trade-offs between connectivity, system performance and energy costs. The focus then shifts to the satellite domain to develop a mobility model of a network subject to orbital mechanics. This is used to verify the performance of the allocation mechanism in a realistic, dynamic environment.

The final modelling chapter, Chapter 7, addresses the fact that all the models up to this point have been cooperative. This is in contrast to real-world markets, where agents are self-interested. I therefore compare the performance of the proposed task allocation mechanism against allocation in a competitive market. This requires the development of an autonomous trading agent for single-sided auctions in distributed markets with limited information. Experiments show that the positive feedback integral to a competitive market results in a worse allocation, especially for sparsely-connected communication topologies.

Chapter 8 reviews the results of the modelling chapters and considers the implications for distributed satellite systems in particular, and for multi-agent systems in general.

2 Background

This chapter reviews the background literature applicable to task allocation in distributed satellite systems. It provides an overview of the related work and motivates the methodology employed, but most importantly it serves to affirm the relevance of the primary objectives of this thesis. Firstly, the allocation of labour in distributed satellite systems is a real problem that needs addressing: traditional approaches to control as generally used in spacecraft engineering fall short for various reasons. Secondly, it illuminates a serious shortcoming in the literature relevant to task allocation in multi-agent systems, namely that very little work bridges the gap between the general theories relevant to task allocation and the specific application examples: we still don't really know how to construct these systems. I propose using the design process of a task allocation mechanism to help connect these extremities.

The discussion starts with literature directly relating to satellite technology to identify the constraints of the problem. It then moves on to a discussion of multi-agent systems, positioning this thesis as using a multi-agent paradigm to address a problem. A market-based approach to task allocation appears very well suited to the specifics of the situation. The chapter concludes with a discussion of the state of task allocation as a field, identifying the shortcomings we are currently facing.

2.1 Distributed satellite systems

Since the launch of Sputnik in 1957 spacecraft sizes have been pushed up by steadily increasing requirements. This has been accompanied by a corresponding increase in development time, complexity, risk and cost (Barnhart *et al.*, 2007). To escape this trend, a drastic shift in mindset is required: instead of treating a spacecraft as a monolithic structure, we can split it into a group of smaller, simpler and less expensive satellites that collaborate to achieve mission objectives. The advantages offered by such a configuration include improvements in performance, cost and survivability when compared to missions that use single satellites (Brown and Eremenko, 2006).

2.1.1 Types of distributed satellites

Shaw (1999) defines the term *distributed satellite system* to be a system of many satellites designed to perform a specific function in a coordinated way. This includes several different types of mission configurations. “Distributed satellite systems”(Bridges and Vladimirova, 2008) can be used as an umbrella term for all of them, including:

Formation flight: Keeping spacecraft flying together with high accuracy, for example as required for synthetic aperture radar applications (Mueller and Thomas, 2005; Thanapalan and Veres, 2005; Ferguson and How, 2003).

Satellite clusters: Spacecraft that fly together in loose proximity to collaborate on a mission (Lee *et al.*, 2005). The exact position of the spacecraft is not critical, but the individual spacecraft all contribute to the mission.

Fractionated spacecraft: The subsystems of a spacecraft are divided into separate craft. These units have limited, but specialised capabilities, but by cooperating can they perform as a single “virtual spacecraft” (Brown and Eremenko, 2006).

The defining characteristic of all these systems is the use of multiple, interacting satellites to satisfy a global demand. As soon as the satellites become interdependent, the problem of task allocation in the multi-satellite environment needs to be addressed.

2.1.2 Missions

TechSat-21 was a U.S. Air Force program that aimed to investigate the benefits of a distributed satellite system. By distributing functionality over system components, it was hoped to reduce costs and increase reliability. The primary mission objective was space-based radar, moving target indication and geo-location (Burns *et al.*, 2000; Chien *et al.*, 2002). The program was cancelled when funding was stopped. As part of the research, the “ObjectAgent” infrastructure for distributed flight software was developed. Control is divided in a hierarchy of agents, with increasing intelligence for higher level agents. The agents exert centralised control over their respective domains; for example task allocation, formation flight or sensing decisions. Redundant instances of agents, which are maintained on other spacecraft, are activated in case of component failure (Mueller and Brito, 2003; Schetter *et al.*, 2000; Zetocha *et al.*, 2000).

DARPA's System F6 program — Future, Fast, Flexible, Free-Flying, Fractionated Spacecraft united by Information eXchange — serves to demonstrate the technological and paradigmatic aspects of the responsive nature of fractionated satellites. In a fractionated satellite the functional components of a monolithic satellite, such as payload, power systems, data processing, communication, etc., are separated onto free-flying spacecraft. These component spacecraft have to share resources transparently to appear functionally equivalent to the original design. The use of heterogeneous, interacting spacecraft modules makes management of the system significantly more complex than the relatively simple TechSat-21 mission. This should provide mission flexibility and responsiveness, as well as further improvements in robustness (Brown *et al.*, 2006).

The available information indicates that the System F6 program is pursuing a networking model similar to that of the internet, with high power and high bandwidth connections between satellites, which will greatly simplify the control strategy (DARPA, 2010; Lobosco *et al.*, 2008). Although System F6 is extremely promising as a potentially disruptive technology that can change the face of the space industry in years to come, its strong focus on the benefits of fractionation itself appears to overshadow consideration of the advantages that collaborating spacecraft can provide. The relatively generous capabilities of the first generation of spacecraft might lead to a management strategy that limits its relevance to smaller spacecraft.

Bekey (2005) proposes the use of 1 000 to 100 000 free-flying pico-satellites to form a radiometry antenna in geostationary orbit. A central receiver satellite serves as a communication link to the ground. The control issues created by the satellites are not explicitly addressed, but the design implies minimal intelligence and decision-making on the part of the pico-satellites.

The von Karman Institute for Fluid Dynamics has initiated the QB50 project, an international network of 50 pico-satellites for multi-point measurements of the thermosphere (QB50, 2010). The three to six month duration makes the use of traditional spacecraft prohibitively expensive, instead they plan to use the low-cost CubeSat platform (Heidt *et al.*, 2000). A single CubeSat has limited capabilities due to its small size (10 cm x 10 cm x 10 cm), however, by equipping a group of satellites with identical sensors, they can still be used for scientific purposes. The reliability of individual satellites is not a critical concern, due to the large number deployed.

2.1.3 Systems engineering

A major challenge is raised by these systems: how should we go about designing them? Some attempts have been made in this direction, but in my mind the matter yet is to be satisfactorily addressed. Part of the problem, I believe, is that we still have not decided how the component satellites should interact.

Jilla (2002) investigates a systems engineering approach that can be applied to distributed spacecraft systems. A large parameter space needs to be searched to find a good solution; this makes enumeration and analysis impractical, and multi-objective heuristics are used instead. This provides design teams with a tool that can identify the most promising areas of the design trade space. Although not explicitly addressing task allocation, the team composition question covered here is important to my work.

An alternative view, suggested by Shaw (1999), is to treat the group of satellites as an information network: all satellites are essentially involved in collecting, processing and publishing of information. By analysing the system in terms of the information flow, cost metrics can be developed. These metrics provide a valuable tool for engineering these types of space systems. This view shares some similarities with the task allocation approach suggested in chapter 3, although my focus is more on energy consumption: all spacecraft are networked individuals that process information.

2.1.4 Task allocation

A limited amount of literature exists that deals explicitly with task allocation in a distributed satellite system. One example is the stigmergy-based task management system proposed by Tripp and Palmer (2010). The ground station publishes a number of tasks to all satellites in the system, they then select which ones to execute based on their own workload, the tasks that had been earmarked by other satellites for execution, and the tasks that had been executed in duplication during the previous round. No direct communication exists between the satellites, but they all need to interface to the ground station. This approach results in a performance trade-off between decreasing the duplicate execution of tasks and minimising the response time. The weighting of these factors can however be changed rapidly while the system is online, to respond to changing mission objectives. The ad-hoc task allocation also makes the system robust to satellites joining or leaving the group. The authors argue against the use of negotiation-based allocation mechanisms on the grounds that the communication cost of negotiation is too high. I will show in chapters 3 and 5 that this is not necessarily true. Despite the potential of this approach it falls short in a number of ways, some of which will be addressed in this thesis.

The question of increasing the number of spacecraft in the system is, in my opinion, only partially addressed. Although the authors suggest the use of a hierarchy of spacecraft to cope with scaling, the behaviour of the allocation approach in systems with more than 18 agents is not demonstrated. In addition, the spacecraft considered were fairly homogeneous: variation in individual behaviour was allowed, but the capabilities of the spacecraft were the same. Tasks have different priorities, but are not constrained in where they are executed. These aspects will be addressed more completely in this thesis.

The work of Si-wei *et al.* (2010) is also relevant, although limited information is available: they present an extension of the contract net protocol (Smith, 1980) for managing the behaviour of observation satellites. The bids are calculated to take the available resources (energy, memory, abilities) and task characteristics into account. To decrease communication cost, auction announcements are only propagated to a subset of the satellites in the system. Allocation success is demonstrated through simulation in a system consisting of seven satellites. The use of a market-like mechanism appears well-suited to task allocation in these systems, as discussed in section 2.4. However, the contribution made by this paper is limited: the implementation overlooks a number of potential problems such as local communication and true scalability.

2.1.5 Formation flight and control

In contrast to the relative absence of work on task allocation, the formation flight and control problems have received significant attention. Wu *et al.* (2008) use a multi-objective evolutionary algorithm to develop a routing scheme that minimises signal delay and transmission power for communication between satellites. It requires global knowledge of the system and is computationally expensive, limiting the responsiveness of the actual formation. Fully distributed formation flight has been described by Pinciroli *et al.* (2008), where an artificial potential field is used to align satellites to a lattice. It achieves autonomy, robustness and scalability by using a bottom-up approach. Self-organisation has also been proposed in the in-orbit assembly of large structures (Ayre *et al.*, 2005; Izzo *et al.*, 2005). None of the above work explicitly addresses task allocation.

Mueller and Thomas (2005) use a multi-team framework for distributed satellite cluster control. The cluster is divided into a hierarchical team structure as a compromise between fully centralised and decentralised approaches, to decrease communication cost. Formation flight is managed in a distributed fashion using team level information instead of an accurate global state. Formation changes are handled by transmitting the desired state to all spacecraft using the hierarchical structure. They reply with a cost vector that relates to the remaining fuel percentage: the best assignment is determined by the top-level node. However, team formation is not discussed (it is apparently considered static), nor is the control architecture applied to anything more than formation flight — the payload functions of satellites are not considered.

2.1.6 Miniaturisation

The technologies used in construction of satellites determine the capabilities of the resulting system. In recent years the use of products and processes developed for the consumer market in small satellites has increased dramatically. Miniaturisation and integration of components decrease costs and reduce the satellite mass (Vladimirova *et al.*, 2006). This reduction in cost makes these access to space much more affordable and accessible for small companies, educational institutions and developing countries (Sweeting, 1992). Spacecraft with a mass below 5kg have therefore become a realistic and much researched goal — the extreme case of a satellite designed to be etched on a silicon wafer is presented in Barnhart *et al.* (2007).

The capabilities of these satellites are necessarily more modest than those of their larger cousins, both in terms of the mission possibilities and satellite reliability. However, if these satellites are utilised in the distributed satellite paradigm, they can also fill commercial and scientific niches, complementing existing monolithic satellites (Barnhart *et al.*, 2007). Currently, miniature satellites serve as valuable educational tools: the low cost and relative simplicity of a smaller spacecraft allow universities to expose students to the entire spacecraft design cycle. For example, several universities have used CubeSats as part of their curricula (Heidt *et al.*, 2000). The low development and launch costs of these satellites have also made them attractive as testbeds for flight-testing new technologies and components without endangering high-value missions, as in the case of the QB50 mission described above, or as discussed by Vladimirova *et al.* (2006).

However, the whole system does not scale equally well: power remains a significant concern. Batteries are still heavy and relatively bulky, and with decreasing surface area the availability of solar energy is also decreased. A certain transmission power is still required to communicate with the ground station, especially to download payload data.

2.1.7 Discussion

The literature confirms that distributed satellite systems have an important role to play in the space arena. Although the first steps towards these systems have already been made, many challenges still remain. While mission design, orbital dynamics and technological aspects have received some attention, questions surrounding the management of these systems have not been satisfactorily addressed, thus confirming the relevance of my previously stated objective of designing a task allocation mechanism for these systems.

In my opinion, a distributed system that consists of heterogeneous miniature satellites holds great promise as an affordable paradigm for a significant segment of the space market. However, this type of system requires an autonomous, robust and adaptive management strategy that takes the constraints on the system into account. Individual spacecraft have limited energy, which limits their ability to communicate, both in transmission range and data volume. Because wireless transmission consumes a significant amount of energy, it can decrease the ability of the satellite to do useful work. However, it also needs to communicate as collaboration is key to completing mission objectives. A successful management strategy will allocate tasks in a manner that balances these factors.

2.2 Multi-agent systems

In designing a mechanism that can manage such a distributed satellite system, it may be tempting to focus on the characteristics of the individual satellites: every one is a complicated electro-mechanical system in itself. While the engineering of the spacecraft undoubtedly matters, I believe the design of the *system* of spacecraft is even more important. The communication and interaction between spacecraft to form an autonomous and robust whole is what makes the distributed approach attractive in the first place. To design a distributed satellite system, the design process should be focused on the system, not the spacecraft.

The paradigm of multi-agents systems (MAS) provides a promising approach to constructing such a system: we define the interactions between agents to achieve a global behaviour. Before rushing to the task allocation problem, I would like to first situate it in a space populated by related systems. I firstly show that task allocation is a very general problem encountered in a number of natural systems: it should not be seen as confined to the technical realm, nor should it be associated with a particular application. Despite these systems being apparently unrelated, the task allocation problems encountered are recognizably similar to that faced in a distributed satellite system. I then proceed by classifying the multi-agent research area into three distinct directions, each driven by a different motivation. Task allocation in distributed satellite systems falls firmly into the segment interested in problem solving, but the overlap with the subject matter in the other areas lend them some relevance.

2.2.1 A broad view

To encourage a system level view of the task allocation problem, I briefly discuss three systems that accomplish it. Despite differences in function and organisation, they all consist of a number of individuals, interacting with each other and their environment, much like a distributed satellite system. These examples come from biological and social systems; their variety illustrates the diversity in possible solutions, ranging from highly decentralised control to strongly hierarchical command structures.

In all of these systems task allocation is woven into the functioning of the system. We are particularly interested in the structure and communication mechanisms that allow these systems operate successfully. All of them have evolved in response to specific environmental stimuli and the associated coordination problems. I deliberately chose non-technical systems to discuss here, to place the focus on task allocation as a problem that is frequently encountered in a variety of scenarios.

2.2.1.1 Social insects

Social insects, such as ants, termites and bees, present examples of societies that are organised in a fully distributed manner, but manage to forage, reproduce and survive, despite the lack of a single coordinator. Individuals interact locally, based on local information, with complex group-level behaviour emerging at a higher level. The colony as a whole can be viewed as a superorgansim: an individual in its own right with survival and information processing capabilities far exceeding that of its constituent parts (Oster and Wilson, 1979; Wilson and Sober, 1989).

Individuals display morphological differentiation into castes, which each caste fulfilling a specific role in the community. We also find task allocation within castes, where specific members fulfil roles of varying complexity. Anderson (2001) divided these into individual tasks, where an individual operates alone; group tasks, where everyone is doing the same; team tasks, with distinct subtasks that must be completed concurrently; and partitioned tasks, when subtasks must be completed concurrently. Communication is predominantly local by using pheromone-based stigmergy, although more complex signalling with higher informational content is also used, e.g., the waggle dance of honey bees (De Marco and Menzel, 2005).

2.2.1.2 Markets

Markets allow buyers and sellers to exchange goods, labour and services. As human society increased in complexity, with higher population densities and better communication, markets have developed from physical places into a wide variety of more abstract exchange structures. Some traditional mechanisms such as Dutch flower auctions were retained, but we now also have markets that are decoupled from physical goods and a physical location: modern derivative markets, for example (Slater and Tonkiss, 2001). The properties of specific markets are determined by the goods they trade and the communication and transport connections between the traders. The market can be regarded as "calculating" the allocation: for some problems and markets, optimal allocation is possible. Markets encourage specialisation: by focusing on doing something well, an individual can increase his returns. Economic agents in the system therefore form a network of interdependencies, where everyone is reliant on the other agents in the system.

Individuals are self-interested as they trade to primarily serve their own needs, but this still results in an efficient global allocation of resources. If unsatisfied demand exists in the market, a supplier will rise to provide it. This lack of central control results in an extremely scalable system.

Money is a key enabler in this system. It facilitates exchange by being highly substitutable, provides a standard for expressing value and allows for storage of wealth. Additionally, through prices, it provides a level of informational abstraction between consumers and the chain of production (Cagan, 1958; Brunner and Meltzer, 1971). The abstraction allows us to, for example, buy a cup of tea without having to take into account the long chain of suppliers, from the tea plantation in India, through various intermediate agents that provide shipping and packaging, to local retailers. All we do is decide whether our valuation of a cup of tea is more than the asking price. By buying it we then increase our personal utilities.

2.2.1.3 Military organisations

Organisations exhibit a wide variety of different forms of organisation, I focus on military organisations because they are the polar opposite of the self-organising systems described above. Classical military command and control structures generally display a strongly hierarchical topology. This structure relies on institutional authority, where one party perceives his or her relationship with another to be institutionally established, where the appropriate interaction is based on obedience. Information is transmitted to a central point (a higher ranking officer), where it is combined with other information to make decisions which in turn flow down the hierarchy for execution. Filtered information is passed higher up the tree — the highest-level individual therefore has an overview of the system, without having to cope with the large amounts of data generated all over the organisation. To allow this abstraction, the lower-level individuals are treated as being functionally equivalent and therefore largely substitutable. The motivation behind this structure is to allow the deployment of a vast number of units, while still maintaining the associated supply and command chains.

Of course, this structure assumes good communication channels exist between the different levels of the hierarchy, even when units are spatially separated. In fact, units with higher autonomy, such as special operations or guerilla forces, usually operate in scenarios where communication is difficult or unreliable. To improve the reliability of the system, a line of succession is established to maintain a command structure. If a high-level individual is killed or otherwise prevented from making decisions, one of a predefined series of successors assumes authority (Coakley, 1992).

2.2.2 Directions in multi-agent research

The above systems can all be regarded as multi-agent systems in the broadest sense of the term: they consist of multiple agents that interact to form a greater whole. A narrower definition of multi-agent systems is however more frequently used, a definition that refers to the research field in computer science that arose from distributed artificial intelligence.

The concept of an *agent* is fundamental to this field. As the name suggests, the concept of agency is central, but a universally acceptable definition is still under debate. The definition suggested by Wooldridge (2002, p.31) in his introduction to the field provides a useful starting point:

> "[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."

The most literal interpretation of a multi-agent system is therefore a system of such agents, interacting with each other and their environment. I will be using this definition throughout the rest of the thesis when referring to a multi-agent system.

Although it grew from an artificial intelligence (AI) base, the wide scope of the field has also drawn researchers from the economics, distributed computing, biology, political science and sociology communities, amongst others. They brought with them a number of different perspectives on how multi-agent systems can be utilised — e.g. Axelrod (2006) describes how agent-based modelling builds bridges between different fields, with reference to his work on the evolution of cooperation. While the different backgrounds provide a rich set of ideas, they have also introduced some confusion: the objectives of multi-agent systems for one research application are not necessarily the same as for another.

To address this, I have divided work in the field into three segments, according to the motivation behind the research: intelligent agents, problem solving, and agent-based modelling. These divisions are not exclusive, instead a large amount of overlap exists between them. This thesis falls into the category of problem solving, but some the work in the other segments is nonetheless relevant.

2.2.2.1 Intelligent agents

At the time of writing, intelligent agents probably represents the most visible portion of multi-agent systems research. The interest here lies in the intrinsic properties of the agents themselves: how to manage with limited information, how to force agents to be truthful, and how to respond to different environmental signals. These agents are seen as representatives for their human owner, and need to make similar decisions for them. When viewed in this manner, it seems like a natural extension of traditional artificial intelligence research. A fundamental assumption is that agents are self-interested, as introduced in the seminal paper by Rosenschein and Genesereth (1985). This leads to a research perspective that seeks to maximise the personal utility of a specific agent in a multi-agent setting. The selfish nature of agents means game-theoretical approaches have been widely used to model and control the behaviour of agents. Game theory has therefore become the dominant tool in this area; however, the computational complexity of resolving large, multi-agent games can limit the tractability of game-theoretic approaches.

The variety of multi-agent scenarios makes direct comparison of the different agent strategies and implementations hard. The Trading Agents Competition (TAC) (Wellman *et al.*, 2003) presents a common ground for comparing agent performances. Having a published problem coordinates research effort and provides a common metric for measuring performance. The classical variant of TAC involves a travel shopping game, where traders combine flights, accommodation and entertainment into packaged trips for probabilistically generated clients. The agents' objective is to maximise the value of each trip. The respective goods categories are traded in separate auctions through the day.

A wide range of agents have been developed, varying greatly in complexity and specialisation. On the relatively simple end of the spectrum, we find agents which result in realistic system dynamics, but with understandable interactions — these types of agents therefore frequently also appear in agent-based modelling work (Cliff and Bruten, 1998; Gjerstad and Dickhaut, 1998; Bagnall and Toft, 2004). More complicated agents usually have strategies that are tailored to specific scenarios, e.g., Gerding *et al.* (2007); Niu *et al.* (2008).

A more complete introduction to agent architectures is given in Wooldridge (2002) and Shoham and Leyton-Brown (2008), while Kraus (2001) provides a useful overview of different negotiation approaches.

2.2.2.2 Problem solving

An alternative approach to multi-agent scenarios is more interested in the ways such a system could be used to address specific problems. In this approach the functions of the individuals in the system do not interest the researcher as much as their combined effect: it is the group-level behaviour that is the ultimate measure of success.

In situations where the individuals cannot be fully controlled or trusted, incentive-based approaches such as mechanism design (Hurwicz, 1973) can be employed. This has been applied to real world problems, for example to decrease carbon dioxide emissions (Ellerman and Buchner, 2007). The difficulty in doing it well is perhaps best illustrated by the auctions for wireless spectrum licences, where governments wanted to maximise revenue. The allocation of 3G licences in the United Kingdom was very successful (Binmore and Klemperer, 2002), while a similar auction for spectrum in Europe had a much smaller revenue (Klemperer, 2002). The same principles have been used in markets of electronic agents to encourage selfish agents to behave in a predictable manner, primarily by incentivising truthfulness. However, the design of such mechanisms remains an apparently delicate art, in many cases relying on assumptions such as individual rationality and complete information to succeed. In some cases mechanism design is definitely needed to help order an otherwise uncontrollable system, although in situations where the designer can control all functions a more direct approach to control is frequently sufficient.

A variant of the Trading Agent Competition Mechanism Design Tournament focuses on market design. It is named CAT: an abbreviation for catallactics, the science of economic exchange (Cai *et al.*, 2009), and also the inverse of TAC (the Trading Agents Competition). The tournament organisers provide a set of traders that interact through a number of markets entered by competitors. The traders decide in which markets they want to trade depending on the market characteristics such as fees, pricing and clearing policy. The observations of the 2007 tournament are discussed in Niu *et al.* (2008).

In many technological systems the system designer is in the unique position of being able to specify the behaviour of agents in the system. Although less open than the systems described above, such systems are frequently easier to manage. These agents can be seen as benevolent: although they may be self-interested, they are as honest as the designer wants them to be, reveal as much information as desired, and they follow the rules stipulated by the designer. Of course, some independence is required to deal with the environment and unforeseen events, such as failures, but the agents do not actively work *against* the system designer. These systems can be managed using similar tools to the open systems above (e.g., Rogers *et al.*, 2004), but have also inherited the legacy of distributed AI and distributed computing systems.

I believe the task allocation problem for distributed satellite systems can be successfully addressed using the latter approach: a multi-agent system, where nodes are autonomous and make selfish decisions, but work towards an improved system state. Distributed task allocation is discussed in more detail in section 2.4. Multi-agent systems as an approach to problem solving has also been applied to coordination of robots, coordination of wireless sensor networks, routing in telecommunication networks, process control (Voos and Litz, 2000), as well as operations research. Some the more relevant applications are presented in Section 2.5.

2.2.2.3 Agent-based modelling

The third component of research on multi-agent systems uses it as a tool to better understand complex interactions in real-world systems. Researchers from social systems (Moss, 2001; Gilbert, 2004), economics (Krugman, 1996; Tesfatsion, 2002) and biology (Emonet *et al.*, 2005) are using simulated models to test hypotheses and explore system-level behaviour.

One example of work in this area is the famous iterated Prisoner's Dilemma contest run by Axelrod and Hamilton (1981). The contest sought to understand how cooperation can arise in biological systems, between apparently self-interested agents. Researchers from a number of fields submitted agents with strategies that were pitted against each other.

In a broader sense, Gilbert (2004) argues for the use of agent-based models to simulate, investigate and understand complex social phenomena. Moss (2001) similarly believes that agent-based models allow us to capture the source of the properties observed at system level, unlike game-theoretical approaches that primarily model these properties. Bedau (1997) makes probably the strongest claim about multi-agent simulation: it is a necessary tool if we want to study emergent system characteristics, as traditional mathematical or empirical tools cannot be successfully employed. Agent-based models allow a constructive approach to systems — instead of describing the behaviour of the system in terms of abstract equations, low-level interactions between agents are defined which lead to emergent global effects.

The work of systems modellers is of relevance primarily because it confirms that man-made systems can exhibit desirable macro-level characteristics. In the case of distributed satellite systems, we are particularly interested in robustness, self-organisation and scalability. This also suggests that designing complex technological systems requires simulation as part of the design process, to test and understand the interdependencies between different components.

2.3 Decentralised task allocation

Task allocation is the process whereby a job is assigned to a particular agent for execution. The challenge lies in matching the right agent to the right job, to maximise global system performance.

The literature relating to decentralised task allocation spans several different disciplines: multi-robot coordination, distributed computing, wireless sensor networks, and operations research. The history and constraints of these fields have largely determined the approaches followed. Task allocation in satellite networks shares this problem space: in surveying the related work, I have focused on the similarities and differences between the systems.

A few attempts at classifying this space have been made, but usually these are field specific. Casavant and Kuhl (1988) composed a hierarchical structure of scheduling in distributed computing systems, while Dudek *et al.* (1996) classified according to the architecture of multi-robot systems (e.g., communication topology, team structure, and team organisation). Seuken and Zilberstein (2008) analysed five different formal frameworks to sequential decision making for distributed cooperative agents in decentralised problems, with an in depth discussion of the performance and complexity of different optimal and approximate techniques. I found the taxonomy of the underlying *problems* of multi-robot task allocation by Gerkey and Matarić (2004) to be most aligned with my objectives. They defined the following three axes:

Single-task vs multi-task agents: Agents can execute a single or multiple tasks simultaneously.

Single-agent vs multi-agent tasks: Some tasks can be completed by one agent, while others need multiple agents.

Instantaneous vs time-extended assignment: Instantaneous assignment permits only instantaneous allocation of tasks, with no information about the future. In time-extended allocation more information is available, such as the distribution of future tasks or the set of all possible tasks.

These dimensions are satisfactory for multi-robot systems, but in reviewing the literature I found the following additional parameters to be useful in describing the more general problem space of task allocation in multi-agent systems.

Communication cost: In some networks bandwidth is effectively unlimited, which makes communication costs negligible, while in others communicating more than is necessary decreases system utility. The cost of transmitting information therefore determines the amount and accuracy of available information.

Group size: The number of agents in the group and the granularity with which it is viewed determine the rules and behaviour that apply: there is a non-linear relationship between the size and the laws that dominate in a group, as argued elegantly by Anderson (1972). A small group (less than ten agents) can feasibly be controlled either individually, or using centralised control; neither of which will work for a large group (in the hundreds to thousands range). The space in-between is an area where the appropriate control mechanism can be contested.

Agent heterogeneity: Diversity in agent types allows role specialisation, while homogeneous agents typically have greater redundancy. Heterogeneity complicates the abstraction process because the specific properties of individuals must be taken into account when making decisions, which increases the information content of the system.

Distributed satellite systems span a portion of this space. The majority of missions will involve single-task agents, with tasks requiring multiple agents to complete (either consecutively or concurrently). In some cases task assignment can be regarded as time-extended in the sense of Gerkey and Matarić (2004). However, the flexibility allowed by the distributed satellite paradigm suggest tasks will often be handled using instantaneous assignment, because the details of future tasks are unknown and roles may change at any time due to component failure or changing mission objectives. For most applications, communication costs will be high due to the power it requires. Initially group sizes will be moderate, in the range of tens to hundreds, although massive systems have also been proposed (e.g. Bekey, 2005). The component spacecraft will be heterogeneous, but with some redundancy.

2.4 Market-based control

Market-based mechanisms provide an attractive set of tools for controlling task allocation in multi-agent systems. Note, however, that a market-based approach to task allocation is just one option from a number of possibilities: coordination using scheduling algorithms (Casavant and Kuhl, 1988), distributed constraint optimisation (Modi *et al.*, 2006) and coalition formation (Shehory and Kraus, 1998), amongst others, all have a role to play in different scenarios. My decision to focus on markets is motivated by their relevance to complex systems, as well as my interest in developing socially-inspired approaches to computing problems. In this section I provide an overview of market-based control, and motivate why it is well suited to address task allocation in distributed satellite systems.

Markets and prices allow modern economies to allocate resources between competing users (Begg, 2005, Chapter 1). Self-interested users trade labour and resources to maximise their own gains, which simultaneously result in an efficient global distribution of goods. In recent years researchers have started to apply these principles to the control of multi-agent systems (Clearwater, 1996). By using an artificial currency, the relative value of resources and tasks in the network can be established to find the best allocation. The market can be regarded as calculating the allocation: for some problems and markets, theory indicates that optimal allocation is possible. If we construct a market in this manner, from the bottom up, it fits happily into the multi-agent systems paradigm. It is closely related to agent-based computational economics and automated trading, and requires use of knowledge from both disciplines (Tesfatsion, 2002). Cliff and Bruten (1999) argue that these market-driven systems display collective social adaptive behaviour, allowing autonomous adaptation to a dynamic environment.

Kraus (2001) reviews game-theory and economics-based techniques used in automated decision making and in negotiation for multi-agent systems. The taxonomy by Gerkey and Matarić (2004) also deals with some aspects of market-based task assignment, comparing it with combinatorial optimisation. Dias *et al.* (2006) evaluated the status of market-based control, specifically for multi-robot systems. They define market-based coordination in multi-agent systems as having the following characteristics:

- The group has an objective that can be subdivided and spread between components. The units in the group have a limited set of resources that can be used to address the problem.
- A global objective function describes the system's behaviour and quality of solution.
- Every agent has an individual utility function, that quantifies the gain (or cost) for executing a certain task. This function can use only local information, but can take multiple factors into account, some representing the cost of task execution, others representing the quality of the result.
- A mapping exists from the individual utility functions to the global objective function. It defines how individual actions will influence the global objective.
- Resources and objectives can be redistributed between agents, using a cost-based negotiation mechanism, such as an auction. Bids are computed as a function of individuals' utilities, and allocation is made to the agent whose bid will maximise the global objective function.

Just as myriad types of human markets have developed, with different rules for different applications; so a great variety of market models are used in control. Although we are interested in artificial markets, contributions from mechanism design and auction theory provide a link to the theoretical basis that validates the approach.

A major challenge is the definition of individual rules that will lead to the desired global behaviour. In the case of distributed satellite systems, this means being sensitive to the capabilities and utilisation of individual nodes, and minimising energy spent on communication, while still achieving robust system-level task throughput. I therefore review related literature with the objective of identifying the definitions of utility functions and the types of markets employed, and how this relates to emergent behaviour.

The contract net protocol (Smith, 1980) is a seminal, high-level protocol for task distribution using a sealed-bid, first-price auction, which relies on contracts. Tasks are announced in the system, agents place bids with valuations based on their perceived utility, and the auctioneer node collects bids until the task can be awarded. Davis and Smith (1983) further explore the use of the protocol, with application to an area surveillance problem. Although no explicit economic notions are included in the bidding, the contract net protocol inspired a significant amount of work that includes stronger market implementations. Miller and Drexler (1988) further explored the market-based approach, with specific emphasis on the computational environment. Wellman (1996) introduced "market-oriented programming", where a computational economy is used to derive the activities and resource allocations of a set of agents. By using a framework that allows definition of a market structure and protocols for deriving price equilibria, it is applied to a multi-commodity flow problem. The generality of the implementation is attractive, due to the freedom it allows in exploring the problem space.

Auctions present a particularly compelling type of market for our purposes: they evolved to allocate goods and labour efficiently, and to function well with limited communication and inaccurate information. The low communication requirement is demonstrated in sealed-bid auctions, where agents do not know the value of bids placed by other agents, yet can deduce the equilibrium price of a commodity over repeated auctions. Gerkey and Matarić (2003) analysed the communication complexity of sealed-bid auctions, finding that they are well-suited to applications where communication is expensive, such as space applications. The extreme case was demonstrated by Blumrosen and Nisan (2002): the efficiency of auctions where only one bit of communication was allowed was only slightly lower than completely unconstrained auctions. In contrast to the work in this thesis, communication had no effect on the utility of agents, nor was topology considered.

The effect of costly bids on the dynamics of sequential English auctions is analysed in Daniel and Hirshleifer (1999). When bidding is free an incremental increase in bid values is expected, which results in many bidding rounds before bidders drop out. This is however at odds with the behaviour of bids in corporate acquisitions, where bidding often proceeds in large jumps, significantly decreasing the number of bidding rounds before a winner is found. They ascribe this behaviour to expensive bids, due to the regulatory and administrative costs associated with corporate deals. If bidding in the auction is treated as a learning process, with bids as a costly signalling mechanism, an equilibrium that maximises the *rate* of learning will economise on bidding.

Babaioff *et al.* (2009) investigated spatially distributed markets. For computerised markets the physical location is irrelevant, but the transferral costs of moving goods between different markets are significant. They present a welfare maximising mechanism that takes the cost of these transfers into account. The allocation calculation can be done in polynomial time. Although spatial distribution is of relevance to my work, this work assumes negotiation is free and therefore transfer costs and topology are known.

2.4.1 A note on terminology: market-based control

The label *market-based control* has been used to describe a variety of systems, ranging from the open-ended markets envisioned by Waldspurger *et al.* (1992) and Wellman (1993) to those that rather use markets as a metaphor for decision making. The former class of systems implement a very real market, where everything is defined by the negotiation and exchanges between agents. Although undoubtedly potentially very powerful from self-managing and robustness perspectives, the complexity of these systems makes them very hard to control reliably and predictably. Some, such as Huberman (1988) and Brooks and Flynn (1989), would argue that the lack of direct control is at least part of the flexibility of these systems. We give up some verifiability and optimality to increase robustness and adaptability. However, this paradigm shift still sits uneasily with the current engineering establishment.

A more moderate view of market-based control is used in the second group, where the market serves as inspiration, but the primary concern is still the *control* of the system. I believe we can retain some of the attractive characteristics offered by markets by borrowing the information flow and decision making mechanisms. By limiting the strategic capabilities of agents and not allowing prices to float freely, the volatility of the system can be limited. The market then serves as a negotiation mechanism: bids are communication messages and auctions a way of determining allocation. Most importantly, prices and money provide a way of representing information upon which decisions can be based. One obvious limitation of this approach is that the system can be susceptible to exploitation by malicious agents, because the market environment is designed to be safe. Systems where the designer does not have complete control over the agents therefore require additional mechanisms to enforce compliance with the expected behaviour.

The proposed task allocation approach should be seen as market-*based*, therefore falling into the second class of approaches. Although undeniably inspired by the power of the open market, restricting some of the degrees of freedom allows for more predictability as required for real-life problems. The primary aim is to allocate tasks successfully, not to emulate a market accurately.

2.5 Applications

The market-based task allocation work described above relies heavily on the theoretical grounding offered by economics, using analytical methods and game-theoretic approaches to understand system behaviour. The literature does, however, also provide a number of application examples. I will review a few of the more significant ones here, in an attempt to identify approaches that can be of use in the distributed satellite system problem. Note how relatively insular these are: many acknowledge theoretical work as inspiration, but successful implementation involves a combination of trail and error, experience and luck.

The characteristics of these applications are compared with a distributed satellite system in Table 2.1.

2.5.1 Distributed computing

A large body of work exists that deals with the management of distributed computing systems: high-performance computing (HPC), grid and cloud computing. In these systems a large number of machines must be coordinated to perform on-demand program execution. The resource availability on different systems varies due to the allocation of tasks, or dissimilar underlying hardware. The definition of optimal allocation varies according to the application: in grid computing fair resource utilisation is important, while HPC data centres consider turn-around time and reliability as critical. Although the systems can consist of a large number of geographically dispersed servers, high speed communication channels exist between them, which allows more accurate information for decision making.

The management of such systems has traditionally been based on a queuing theory approach, which assumes a central scheduler (Streit, 2001; Hovestadt *et al.*, 2003). These approaches have been largely successful, due to the availability of communication and ample processing power. However, as argued by Bullock and Cliff (2004), the increasing complexity of information and communication technology systems results in the emergence of unpredictable

Table 2.1: Comparison of the characteristics of problems related to task allocation in distributed satellites. All of these systems share some of the characteristics of distributed satellite systems and are therefore relevant. However, no clear mapping exists between the types of solutions seen in these fields and the characteristics of the problem space.

	System size	Communication cost	Communication bandwidth	Processing power	Mobile nodes	Node failure	Topology
Distributed satellites	Medium – large	High	Low	Medium	Yes	Yes	Variable
Grid computing	Large	Low	High	High	No	Yes	N/A
WSNs	Large	High	Low	Low	No	Yes	Sparse, distributed
Multi-robot systems	Small – large	Medium	Medium	Medium	Yes	Yes	Variable, distributed
Telecommunication networks	Large	Low	Medium – High	Low	No	Yes	Sparse, distributed

system-level behaviour. They claim that complex adaptive systems present a suitable paradigm for managing these applications.

Chakravarti *et al.* (2006) proposed the use of a biological metaphor for coordination: strongly mobile agents colonize under-utilised machines on the network. The distributed computing system *Spawn* uses a market to allocate computing tasks in a heterogeneous network (Waldspurger *et al.*, 1992). The system self-organises to distribute loads fairly by mapping idle resources into currency. Another example is *Mariposa*, a distributed database system that also uses an economic paradigm to address query execution and storage management (Stonebraker *et al.*, 1994). Huberman and Hogg (1995) argue that a computer network can be viewed as a "community of concurrent processes", or a computational ecology (Huberman, 1988). They further show that local rules can lead to globally stable allocation behaviour (Hogg and Huberman, 2002).

Robinson (2002) presents the development of a market-based control system to manage the workload in a simulated utility data centre. ZIP-trading agents are used to allocated jobs in a continuous double auction; this achieves efficient computational load-balance performance under a variety of scenarios. Evolutionary algorithms are used to tune both the parameters of the individual agents, and the marketplaces through which they interact. A particularly relevant feature of this work is the use of distributed markets — interactions between agents occur on topologically local markets. The topological constraints are treated as a given, with no explicit mention of communication costs.

Autonomic computing appears to the currently fashionable incarnation of adaptive, distributed computing. Kephart and Chess (2003) describe the vision of autonomic computing as consisting of heterogeneous hardware from different vendors, that can configure, pro-actively optimise, heal and protect itself in a dynamic and unpredictable environment. This would be in contrast to the status quo, where configuration and management are time-consuming procedures requiring detailed knowledge of the other components in the system. The large number of interdependent but manually-tuned parameters results in fragile systems that are vulnerable to cascading failures and external attacks. Different approaches to achieving these goals are being explored, including agent-based approaches (Jacyno *et al.*, 2008; Kota *et al.*, 2009) and heuristics inspired by nature (Shackleton *et al.*, 2004).

Although much of the work in distributed computing systems incorporates the characteristics of the underlying infrastructure — such as servers, routers and a communication network — the problem of distributing jobs and data is very similar to the task allocation problem in distributed satellite systems.

The most prominent difference is the relatively inexpensive communication in most ICT systems: although bandwidth needs to be considered, the physical topology of the network has little impact on its operation. The self-management objective of the autonomic computing community is very relevant to our system, as are the requirements of robustness and graceful degradation.

2.5.2 Wireless sensor networks

Literature relating to wireless sensor networks (WSNs) constitutes another interesting domain. These networks consist of spatially distributed modules that cooperatively measure the environment. Each module can sense information, perform limited processing and relay data to other modules. The accumulated data from all the nodes in the network provide a system-level view of the environment. These networks usually consist of hundreds of sensing nodes, with a central sink to extract data. Nodes often have a limited power supply, therefore unnecessary communication shortens the node lifetime and decreases network performance. Nodes function in different ways, depending on their position in the network. Nodes near the perimeter have a sensing role, while those closer to the source need to spend a greater proportion of their energy relaying the measurements of others.

These networks are challenging to manage efficiently due to the scale of the system, the high cost of communication, the frequently unknown topology, the multiple roles nodes can assume, and node failures — all problems that are faced by distributed satellite systems too. The main differences lie in the information flow in the two systems: for satellites tasks are allocated to specific individuals, while WSNs frequently rely on group measurements.

In many cases retrieving data is more important than knowing which node measured it (Jamal and Kamal, 2004). This has led to the adoption of a data-centric routing protocol by Intanagonwiwat *et al.* (2003), named directed diffusion. A sensing task or subtask is publicised through the network, specifying the data of interest. This sets up gradients along which measured data flows back to the sink. Reinforcement is used to select the shortest paths, while data aggregation (duplicate measurements are not propagated) and in-node processing minimise communication. A negotiation-based routing scheme is presented in Heinzelman *et al.* (1999): nodes use meta-data to eliminate the transmission of redundant data. Rogers *et al.* (2005) and Rogers *et al.* (2006) used mechanism design to balance the conflicting needs of data sensing and data routing in the network. Nodes need to transmit their own data to the sink, but also spend some of their energy relaying measurements by other nodes. The described payment rule rewards the relaying of data from distant nodes, while still encouraging locally sensed data. This gives a local decision rule that delivers good results. Hassanein and Luo (2006) improved communication reliability by taking the remaining energy in nodes into account to prevent depletion, in effect spreading the routing cost over the fittest nodes. The routing of data in WSNs is analogous to task allocation in the distributed satellite network, as the cost of routing needs to be distributed in a way that maximises network utility. Furthermore, several of the energy optimisation approaches in WSNs can be useful in satellite applications too (Vladimirova *et al.*, 2008).

2.5.3 Multi-robot control

From a computational point of view, a multi-spacecraft system can be treated as group of interacting robots that need to be autonomously coordinated. Researchers of distributed robot coordination have extensively drawn on natural metaphors, with varying levels of sophistication. Garnier *et al.* (2005) demonstrated cockroach-like aggregation, while collaborative stick-pulling is discussed in Lerman (2004). Minimisation of energy expenditure in a foraging task was investigated by Campo and Dorigo (2007): a multi-foraging strategy was identified where efficiency is defined as a function of energy; robots then base their behaviour on the expected global energy gain from foraging. Stigmergic communication has also been used in task allocation (Bonabeau *et al.*, 1999). White and Helferty (2005) applied this to robotic soccer in the RoboCup competition (Kitano *et al.*, 1998), where robots choose their roles based on environmental stimuli.

The information that can be extracted from the environment is, however, limited. Explicit communication is necessary where knowledge of the internal state of other agents, or the global status of the system, is required. This can range from relatively simple signalling, such as used by Vaughan *et al.* (2000) for resolution of spatial interference, to exchange of a significant volume of data, for example in collaborative exploration where partial maps are shared (Konolige *et al.*, 2006). The use of markets, where robots base their bids on their fitness for a task, lies between these extremes. An artificial currency allows robots to make decisions based on their internal state and global information, which is encapsulated in a single price figure, while relatively little communication expenditure is required.

Gerkey and Matarić (2002) used a variant of the contract net protocol (Smith, 1980) to coordinate embodied agents in “loosely coupled” and box-pushing tasks. Roles are allocated using an auction, but the communication protocol assumes sufficient bandwidth to allow flooding of packets. Tasks are randomly introduced to the system over time and assigned to the fittest robot available. Zlot *et al.* (2002) presented the use of a market to manage efficient exploration of an unknown area: robots compare their own cost of visiting a waypoint against trading the task with a potentially better situated vehicle. This is an instance of time-extended assignment, as robots can trade tasks that need to be completed in the future. Target detection and allocation using miniature aerial vehicles is presented by Sujit and Beard (2007). A distributed auction is used to make decisions that take the kinematic and sensing constraints of the vehicles into account. In some of the above systems, attempts at minimizing communication are made. However, in none of them is communication treated as an expensive resource, as is the case in distributed satellite systems.

2.5.4 Operations research

Lessons can also be learned from operations research. Human organisations typically consist of a heterogeneous collection of specialised agents that collaborate on larger problems. Communication can be inexpensive in small organisations, but bureaucracy in large organisations or transport costs in physically distributed systems can become prohibitively expensive. Chang and Harrington (2000) compared centralised and decentralised organisation in retail chains. They found that centralisation performs best in markets that are relatively homogeneous, while greater variation favours a decentralised approach because local adaptation is necessary. Local variations in markets can in turn be ascribed to their isolation, either absolutely or due to high communication cost.

System reliability is the result of interaction between policy and investments in infrastructure: Hsieh (2003) investigated the relationship between hardware redundancy and optimal task allocation in a distributed computing network by using a genetic algorithm and local search hybrid. Hardware cost and communication time are unified in a mathematical cost model, which is then minimised while taking system reliability into account. Although the exploration of task allocation, reliability and the system component space has great relevance to management of a distributed satellite system, the level of abstraction in the model fails to take many real-world aspects into account, such as allocation overhead, topology changes, communication delays and imperfect information.

2.5.5 Call routing

Gibney *et al.* (1999) presented the use of market-based control for routing in telecommunication networks. Two sealed-bid auctions are used: slices of bandwidth on links between nodes on the network are sold to path agents, while a path market sells the slices of bandwidth to call agents to connect calls. A first-price and Vickrey (second-price) auction are compared to a conventional static routing algorithm. The first-price auction performed as well as static routing, while the Vickrey auction did not show an improvement in efficiency over the first-price auction. Vickrey auctions were originally introduced to prevent counter speculation between agents, resulting in more efficient auctions Vickrey (1961). However, as this model shows, in many closed artificial systems Vickrey auctions do not improve efficiency and are effectively unnecessary, because the designer defines the agent strategies.

2.6 The state of task allocation in multi-agent systems

In the preceding sections I presented the literature that is of relevance to the task allocation problem. This literature spans a number of fields, because task allocation is a problem common to many different applications. In this section I would like to take a step back and analyse the shape of the task allocation landscape and discuss its deficiencies. My most important observation is that a gap currently exists between the work on the theoretical end of the spectrum, and that dealing with applications: we, in effect, still don't know how to build these systems. I conclude by arguing that this shortcoming can be partially addressed by using responsible design practices, thus motivating the second of my major research objectives.

2.6.1 Where are we?

The vast majority of multi-agent task allocation cases cited above involve specific problems, for example multi-robot exploration or management of autonomic computing systems. The specific nature of these examples encourages solutions that deal with the characteristics of these problems. It can therefore be hard to separate the task allocation component of the solution from the additional infrastructure that allows it to function correctly.

At the other end of the scale is the portion of literature that deals with high-level ideas that are more generally applicable. Most of these are derived from economic theory or operations research, with much stronger mathematical or theoretical bases. The generality is achieved through abstraction, so in the

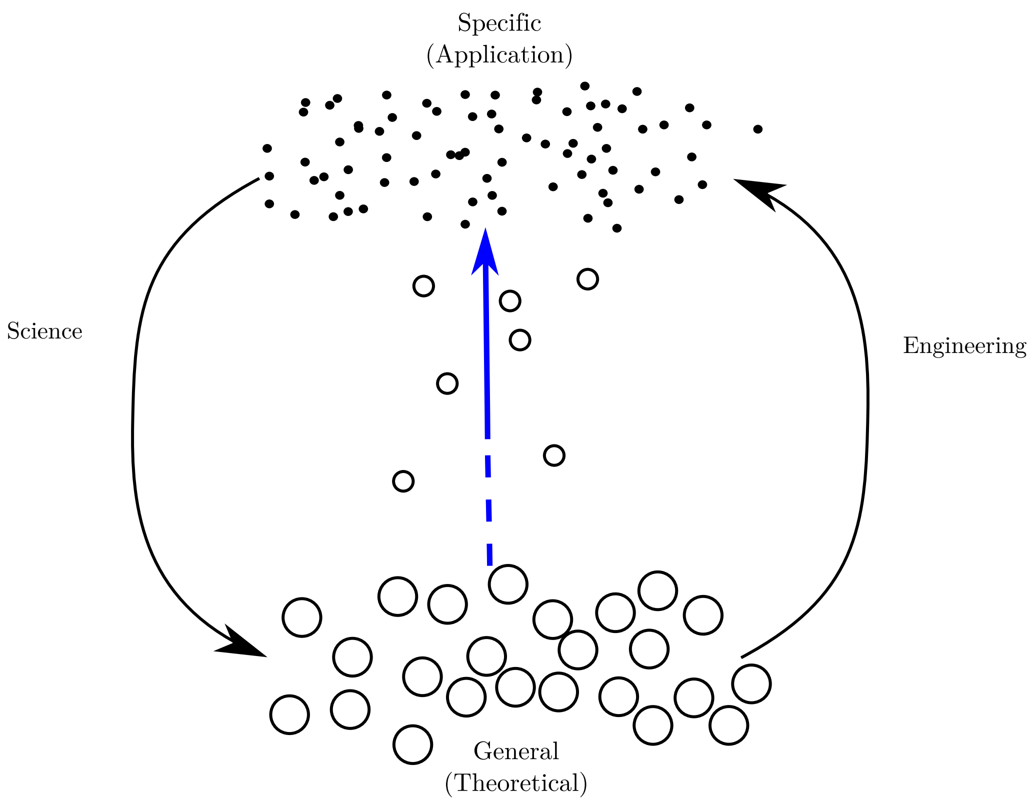


Figure 2.1: Work related to task allocation in multi-agent systems can broadly divided into two categories: application specific engineering problems (small dots), and abstract, general theories and proofs (large circles). The space between these poles is quite empty, apart from some taxonomies and a few attempts at agent design methodologies: we still don't really know how to design these systems. The work in this thesis traverses a section of this space, as indicated by the blue arrow. This allows the design process to be utilised as a way of exploring this space, as shown in more detail in Figure 2.2.

process the application level detail is lost. This could be seen as a fair trade-off: after-all, we need the abstraction to better comprehend the system.

When I initially encountered this apparent wealth of related subject matter, I was optimistic. Surely, with so many applications and a general theoretic basis, the design process should be straightforward? However, I soon realised that a vast gulf exists between the two extremities. For example, we have elegant abstract models that can tell us much about the finer points of market design, and we have extremely complex physical systems doing our bidding, but we have no bridge connecting the two. My initial excitement quickly faded when I realised that *we still have no principled methodology for moving from a specific problem to a trusted solution.*

This situation is graphically represented in Figure 2.1, reproduced from Chapter 1. The large number of specific applications is represented using the small dots at the top of the diagram, while the more general theorems can be found at the lower end. Distilling the specific examples into something more general can be seen as science, while movement in the opposite direction can be described as engineering. The distribution of work is bimodal: we have a large cluster at the top and another one at the bottom. The centre is, however, noticeably empty, implying that a severe disconnect exists between the two poles. I am not the first to make this observation: Wooldridge (2002, chapter 10) similarly concludes that the existing methodologies for designing agent-based systems are "rather tentative". The techniques he discusses are largely derived from current software development methodologies. As such they offer descriptions of the *process*, but do not offer any guidance in selecting the "right" solutions.

We find ourselves in a situation where applications are "legitimately" developed without a glance at the theoretical side of things, as many designers feel the gap is too great to be crossed. Another symptom of the problem is that even those who would like to build on theoretical principles are faced with a vast array of general concepts, with no map of which ones are applicable to their specific problems. This is once again due to the fragmented nature of our knowledge: we need paths linking the general side to specific applications. Only by focusing on filling the empty space, can these issues be resolved.

The central area is luckily not completely empty. Over time the work on either side will slowly grow towards the opposite end, hopefully meeting somewhere in the middle. Some taxonomies already provide an analysis of related problems, such as the work by Dudek *et al.* (1996), Matarić *et al.* (2003) and Gerkey and Matarić (2004), dealing with their specific domains. The comparison of different decentralised decision-making processes by Seuken and Zilberstein (2008) also falls in this space. A few engineering examples exist that use a theoretical grounding to design multi-agent systems, as demonstrated by Rogers *et al.* (2004), while Ygge and Akkermans (1999) identified the need for empirical comparison of market-based control and other systems. These examples are unfortunately few and far between.

2.6.2 How did we get here?

This state of affairs can be ascribed to a combination of a number factors; I will describe the ones I find most relevant here.

The argument can be made that we are involved in a young discipline.

Computer science is but a few decades old, and multi-agent systems have only become a realistic prospect in the last twenty years. Operations research, as a research field, has only existed since the Second World War. In comparison, disciplines such as physics and chemistry can lay claim to a coherent narrative developed over hundreds of years. The central ideas and methods of mature academic fields were established by a combination of vigorous debate, careful experimentation and categorisation. In the process core concepts were tested again and again, and initially promising ideas were disproved and discarded. While the perimeters of these fields are still contested, the base is sound. Given this lack of time, it is therefore no wonder that our multi-agent task allocation knowledge looks pale in comparison. Over time, we will hopefully continue to fill in the empty parts of the space.

Or will we? I suspect that the modern incarnation of the academic system also plays an important role in polarising this distribution. The basic problem lies in the inter-disciplinary nature of the task allocation domain. Those working on the application end are generally specialists in their respective fields, where task allocation is but one problem they encounter. These researchers have little incentive to relate their work to the bigger picture in a disciplined manner, as this requires a substantial investment of time and effort. In the meantime it is still expected of them to be experts in their respective fields, which results in relatively few people exploring the commonalities between disciplines.

On the general end of the spectrum, task allocation is usually encountered as a more abstract concept. To enable themselves to generalise, these researchers avoid the specifics. This too is understandable: the returns from having a working implementation is limited as we cannot use one example to prove a theory correct. Then there is the time investment to consider: physical systems are not trivial to implement. In both cases the academic system rewards specialisation in existing fields; the attractions in becoming a task allocation specialist, with knowledge of both theory and application, are slim. If the wide variety of goals of people doing multi-agent research is added to this, then the scenario becomes even more bleak. (Games such as TAC, CAT and the RoboCup competitions do offer some relief here by providing a common problem and focusing research effort, which results in a thorough exploration of a particular area in problem space.)

Finally, we, the researchers, cannot be absolved of all blame. We are after all the ones who make up this system. Perhaps we find it easier to add new points to the space in Figure 2.1 than we do to order the space, connecting the already existing dots? The claim that something is “novel” is after all much more exciting than saying “the space has been tidied up”. It is also often easier to address a new problem than it is to spend days ordering the existing literature.

2.6.3 What can be done?

To bridge the gap between the general and the specific, a two stage process is required. Initially, similar problems and applications need to be identified and related to each other. Once we have a better idea of problem space, we can move on to formalise a design methodology. This methodology will allow system designers to select an approach from the numerous options available. This is the grand effort towards principled design of multi-agent systems, but where does my contribution fit in? This thesis is concerned with a small part of the problem space: I can therefore only address problems directly related to distributed satellite systems. However, this should be done in a way that contributes to the greater effort. By identifying the similarities and differences between systems that are similar, the local problem space can be mapped. At the same time, the methodological issues of relevance to system designers are also explored, again with application to distributed satellite systems: how do we navigate the design decisions as we move from the general to the specific? My contribution should therefore be seen as one brick in the bridge across the gap in Figure 2.1 — with sustained effort from the multi-agent research community, we can develop the understanding and tools required for principled design of these systems.

I believe that the systemic factors discussed above will continue to inhibit research in the space between the general and specific ends of the task allocation spectrum. Some brave individuals might make contributions, but until a critical mass is reached in the central area, the vast majority of work will still be focused at either end. Things may improve over time, but this shouldn’t stop us from addressing the situation right now. This led me to pose the question: *how can we utilise our existing research work flows to improve the situation?*

I therefore propose the use of my design and verification methodology to help bring some order to this space. When designing a solution to a specific problem, we start with a general model of the final result. As this model is progressively refined by making design decisions or incorporating physical constraints, it becomes a more and more accurate representation of the ultimate solution. A key realisation is that the original, generic model can potentially be mapped to a number of solutions, depending on the decisions and refinements we make. In this way we can see the design process as branching off at each decision point, as shown in Figure 2.2 (reproduced from Chapter 1).

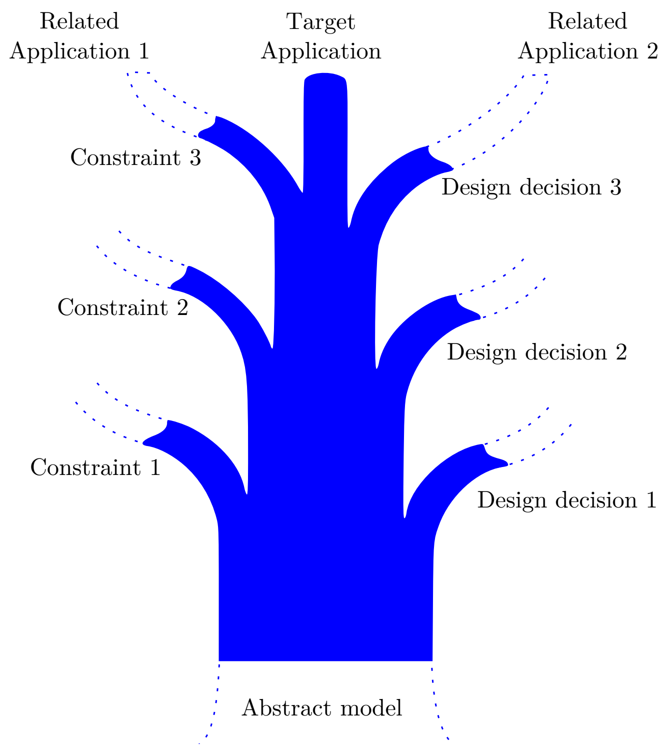


Figure 2.2: The design process can provide an emergent taxonomy of related problems. As we refine a general model to be more specifically applicable, we make design decisions and take constraints into account. These branches can lead to different, but related points in the problem space. This represents a more detailed view of the arrow indicating traversal in Figure 2.1.

The problem this thesis is addressing is definitely a point at the specific end of the spectrum in Figure 2.1, but in the process of moving from the general solution to the specific one, we will encounter wireless sensor networks, distributed computing and other types of allocation problems. The design decisions relate these applications to our allocation problem: by traversing the problem space in this way we are, in effect, constructing a taxonomy based on the decision points in the design process. This forms the second of my primary objectives: helping to order the problem space surrounding my task allocation problem, by situating my approach relative to other work and related systems.

In short, this boils down to responsible engineering. Throughout the design process, decisions are verified through testing. By identifying the family of systems we are excluding with a specific design choice we are also relating that system to ours. The communicated results should not only describe where we ended up and how we got there, but perhaps most importantly, what related problems we past on along the way.

2.7 Discussion

This chapter reviewed the background literature relevant to task allocation in a distributed satellite system. A review of current space technology confirms the need for a mechanism that addresses the management of multi-satellite systems in a way that is robust, scalable, and conscious of the limited energy available to the satellites. In my opinion, approaching it as a pure satellite engineering problem is incorrect: it is the interactions between components that determine the dynamics of the system. I therefore propose the use of a multi-agent approach to task allocation. Specifically, market-based control offers a promising solution to managing such a system: using an auction mechanism to allocate tasks should allow for efficient and adaptive allocation with minimal communication.

However, the literature also highlighted that we don't really know how to reliably map from our abstract models to the messiness of the real world, and still have a working system. At the one extreme we have numerous specific applications and at the other general theories regarding task allocation, but we lack a bridge between these two parts. A basic step towards improving this situation involves relating different task allocation problems and the types of solutions that can be used to address them. To this end I propose using a responsible engineering approach, where the decisions made during the design process provide links to closely related problems. In this way the design and verification process can be used to help build a taxonomy of the space surrounding task allocation in distributed satellite systems.

The work in this thesis should therefore be seen in the context of a larger research effort into ways in which we can borrow from social and biological systems to find ways of coping with our own increasingly complex technological systems.

3

Design of the task allocation model

This chapter describes the process followed to design the task allocation mechanism for the distributed satellite system. To better understand the challenges of task allocation in this application, I first describe a mission scenario that will serve as a model of the type of system that we'd like to control. Presenting a specific model in this way forces us to specify exactly what we mean by "tasks", and helps to identify the constraints inherent to such a system. Furthermore, the model provides a context for precise definitions of the objectives of successful task allocation, namely maximized allocation, robustness and scalability. With these aspects clearly defined, we then move to the other end of the design spectrum: a series of abstract models of task allocation in human organisations is presented. A simple model of task outsourcing is used to develop a market-based allocation process, which is then related back to the distributed satellite scenario, thereby defining the simulation model that is used in the rest of this thesis.

3.1 Reference mission

The literature review showed that although several distributed spacecraft missions have been proposed, none have been flown yet. I therefore construct a generic mission scenario to serve as a reference in the design procedure.

A group of small, low-cost satellites, numbering in the tens to hundreds, is positioned in close proximity to each other in low earth orbit. The overall mission objective is a combination of earth observation and measurements of the space environment. The satellites are not homogeneous, but specialise in various roles which correspond to different configurations of spacecraft.

The first two kinds of satellites form the payload component of the multi-satellite system. In other words, they do the work the mission was designed for. The earth observation part of the payload employs one class of pico-satellite, where every individual is equipped with a camera which can be used for low-resolution photos of a wide area. Alternatively the image data can be combined using super-resolution methods for higher resolution coverage of a narrow area (Farsiu *et al.*, 2003). The environmental measurements are taken with a number of small satellites, each equipped with sensors to map the magnetosphere. The simultaneous measurements allow for high resolution observations of spatial and temporal behaviour of the Earth's magnetic field — similar to Clarke *et al.* (1996) and Friis-Christensen *et al.* (2006).

The third class of satellite provides the mission infrastructure that is used by the payload spacecraft. Dedicated communication satellites are responsible for communicating with the ground station: commands are uploaded to the satellites, while telemetry and payload data are downloaded again. To achieve this, the communication craft are equipped with high-gain antennas and amplifiers, as well as non-volatile memory for storing data before downloading to earth. Redundant instances of all the spacecraft are deployed simultaneously. If, for example, three communication spacecraft are available they can all be used to transmit data back to the ground station. If one of these spacecraft stopped functioning the available bandwidth will be reduced, but communication can still proceed.

The system receives a set of high-level commands while in contact with the ground station, e.g. "photograph Rio de Janeiro" or "return magnetosphere measurements". Note that these commands do not have to specify the specific satellite that will take the picture, or the satellites involved in measuring the magnetosphere, instead just the *function* is specified. We need to abstract to the functional level if we want to free the ground station from managing the individuals in the group. It is of course possible to address the specific modules in the system by aiming the request specifically at them, e.g. "measure status of `0xfac3b3e5`", but this type of micromanagement should not constitute a regular part of system operations.

As the group orbits around the earth, their formation varies periodically due to individual spacecraft orbiting around the centre of mass of the earth, as shown in Figure 3.1. In addition, non-periodic perturbations influence different satellites in different ways: atmospheric drag effects, magnetic interaction and solar pressure all depend on the characteristics of the individual satellites (Larson and Wertz, 1999, Chapter 8). This continual change in topology has a significant impact on the routing of communication within the group: maintaining a map of the network topology requires constant communication and processing.

Individual satellites are powered by a combination of photovoltaic cells and

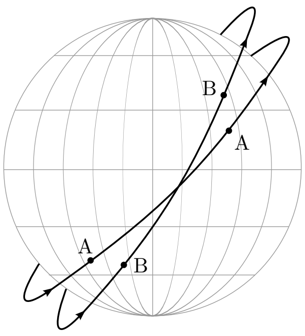


Figure 3.1: As two satellites, here labelled A and B, orbit around the earth their relative positions change continuously due to both spacecraft orbiting around the centre of mass of the earth. In a network of co-orbiting satellites, the topology will change similarly.

batteries. Solar energy is collected while the spacecraft is illuminated and stored in rechargeable batteries. When more power is needed than can be supplied by the solar cells, for example when the spacecraft is in eclipse or while transmitting data to the ground station, the accumulated energy in the batteries is used. This raises an important constraint: allocation of tasks will substantially decrease the energy available to a module, which regenerates relatively slowly. This scarcity of power also implies that all communication will have a significant cost: the act of communicating decreases the energy available for task execution. Furthermore, batteries can only store a finite amount of energy: if they are fully charged, potentially useful solar energy cannot be captured.

3.1.1 Constraints

The distributed satellite system is subject to a number of constraints that determine system operation and potential task allocation mechanisms. These constraints range from mission design level issues, such as money, available launch mass and mission objectives; to spacecraft design issues, for example mechanical structure, attitude control and propulsion; to operational questions such as ground support. When designing a multi-spacecraft system, the interactions between satellites also need to be taken into account, as these play a key role in determining the system-level behaviour. The most prominent constraints to be considered here are the inter-satellite data rate, processing power, and the energy available to individual satellites.

The *inter-satellite data rate* determines the amount of data that can be exchanged between spacecraft. Bandwidth is determined by the modulation scheme, antenna design, system noise temperature, and transmission power.

The first three parameters are usually fixed early-on in mission design, leaving transmission power as the main factor determining the effective communication range. A in-depth discussion on link design can be found in Chapter 14 of Larson and Wertz (1999); for the purposes of this discussion it is sufficient to note the importance of transmission power.

The second constraint under consideration is *processing* power. Component satellites require sufficient processing power and memory to maintain network information, process communication packets, and calculate allocation. The exact amount required is determined by the allocation mechanism. Although additional processing is relatively inexpensive, it will increase the power needs of the spacecraft. To keep this reasonable, the computational requirements of the task management approach should therefore be scalable.

This brings us to node *energy*. Satellites are self-contained entities, and as a result all energy used is either stored or generated on-board. If we consider smaller, simpler satellites, we find that the available power is extremely limited[1]. This energy is used on spacecraft housekeeping (determining position, orientation, etc.), payload functions (taking photos or measurements) and, in the case of a distributed satellite system, network-level management (task allocation, network information). Because all these components compete for the same energy, we see that communicating decreases the energy available for payload functions. In allocating tasks, we need to balance the exchange of data needed to achieve a good allocation against actually getting work done.

3.1.2 Abstraction

With the above scenario as a reference mission, we can now develop an abstracted model that captures the relevant characteristics of distributed satellite systems. A graphical depiction of the mission configuration is shown in Figure 3.2. A number of spacecraft communicate with their local neighbours to form a network of specialised agents. This network can be represented as a graph of agents, where vertices correspond to satellites and edges to communication links between them. The vertices are treated as agents, as they can behave as autonomous actors, or respond to commands received from other agents. The agents form a spatially distributed network where nodes only have direct access to their neighbours, while communication with more distant spacecraft takes place via chain of local connections. These agents need to collaborate to complete complex tasks. A network representation of the same

[1]For a 100 mm x 100 mm x 300 mm cubesat, the orbital average power is approximately 6 W.

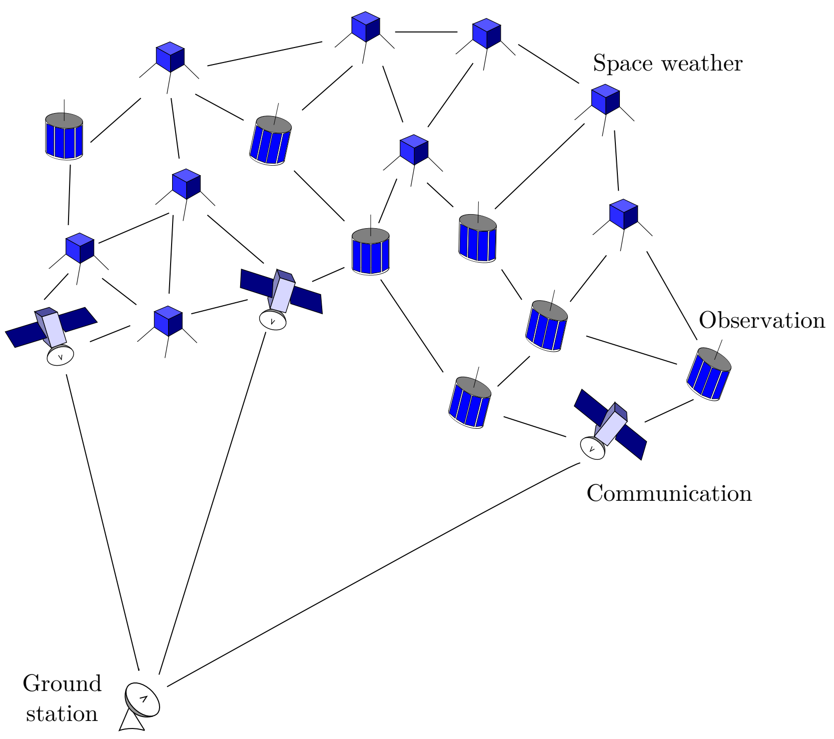


Figure 3.2: Distributed satellite system consisting of three different types of satellite: communication satellites to provide the interface with the ground station, remote observation satellites and the space weather sensing payload.

scenario is given in Figure 3.3. Different colours are used to represent the types of agents, each with a particular set of skills.

To allow better comparison to other multi-agent allocation problems, the characteristics of the task allocation problem can be summarised as follows:

Multi-component tasks: Tasks consist of multiple, interdependent components and are executed by a number of agents, each performing the task component that it is equipped for.

Distributed task origins: There is no central source of all tasks; instead, tasks can spawn at any node. Although tasks initially originate from the communication nodes, subsequent task components can be generated by any node, depending on the task structure.

Ad hoc task assignment: The sequence in which tasks will arrive is unknown at design time. The structure of tasks is however assumed to be known.

Task execution: Agents are characterised by a current energy level. Successful task execution will lower their energy levels, and agents will only attempt a task if they have sufficient energy available.

Heterogeneous agents: No two agents are exactly the same: agents differ in terms of the types of tasks they can execute, their resources (available energy) and their location in the network.

Autonomous agents: Large systems are hard to manage at an individual agent level; instead the agents need to autonomously manage themselves.

Benevolent agents: The agents are presumed to be self-interested and maximise their own gain, but they are also benevolent. They do not try to exploit other agents by cheating, nor do they attempt to exploit the system.

Limited energy: Agents have limited energy available to execute tasks and communicate. By measuring the energy available on-board, they can calculate their capacity for work. Energy can regenerate over time (e.g. by using solar panels).

Agent failure: Agents can stop functioning at any time, due to lack of energy (temporary) or physical failure (permanent). This has an impact on the available resources in the system, and also influences the communication topology.

Local communication: To conserve energy, most communication is with local neighbours. Multi-hop routing is required to transmit messages to more distant agents. Agents therefore rely on local information to make decisions.

Communication cost: Communication requires a significant amount of energy, but the same energy could instead be used to perform payload functions. A balance must be found between negotiation and doing work. If an agent's energy falls too low, it will be unable to communicate. We distinguish between the cost of negotiation, which is significant but small, and task transfer cost, which involves much more data and is therefore up to several orders of magnitude larger.

Broadcast communication: Messages are transmitted to all the neighbours of an agent, as a result of the wireless communication medium. The energy cost to an agent is therefore the same whether it transmits to one

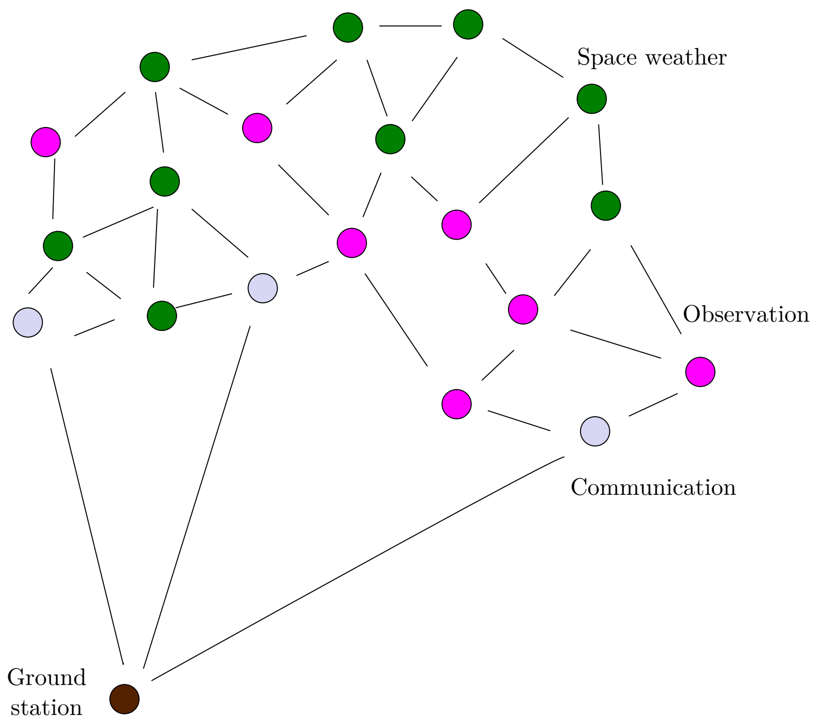


Figure 3.3: Equivalent network for the distributed satellite system shown in Figure 3.2. The different types of agents are represented using different colours: brown for the ground station, grey for communication nodes, while the remote observation spacecraft are coloured magenta and the space weather sensing payload units are green.

or to all of its neighbouring agents. An addressed communication scheme could be implemented on top of this, where recipient nodes filter incoming messages and only respond to the relevant subset.

Spatial distribution Agents are distributed in three-dimensional space, which determines the communication topology. A given transmission power corresponds a maximum range for communication. Thus, by specifying transmission power, we implicitly define a communication topology in the form of a random geometric graph. The space can be regarded as uniform in transmission characteristics, and clear from obstruction. Note that there is no risk of the Earth occluding some parts of the network from others and interrupting communication, due to the relatively small communication ranges (under 10 kilometres) when compared to the altitude of the satellites (more than 500 kilometres).

Symmetric communication As reliable communication will require transmission from both parties involved in a message exchange, I restrict myself to symmetric communication links. In other words, if node A can communicate with node B, then B is also able to communicate with A. In situations with asymmetric transmission powers, the communication link will only be treated as valid if both parties can receive the other's messages. All links in the communication graph are therefore bidirectional.

Volatile topology: The communication topology is volatile due to continuous changes in the relative position of agents, as well as agent failures.

Scale: The number of agents in the system ranges from tens to hundreds of agents.

The above characteristics position this problem as closely related to several other multi-agent applications. The task allocation problem is quite similar to resource allocation in distributed computing domains, but their communication cost is effectively free. Wireless sensor networks have a very different task structure, as they focus on measurements, but the communication cost problem is nearly the same. Local communication is seen in social networks and mobile ad hoc networks (MANETs). Mobile robotics can also share a subset of these factors, depending on the application. These relationships will be explored in more detail in subsequent chapters.

3.1.3 Task allocation objectives

At this point, with the system properties described, it is necessary to clearly define the desired characteristics of the as-yet-unspecified task allocation mechanism. When expressed in the most general sense, we would like to allocate tasks in a manner that maximises the amount of work done by the system. Expressed differently, the money that pays for the system should deliver the best possible returns, despite agent heterogeneity, network scale and agent failure. Most of the following points have already emerged from previous discussion, I will now summarise and formalise them.

3.1.3.1 Maximising allocation

The most obvious requirement is that the number of tasks successfully allocated is maximised. If a task consists of multiple components that are executed separately, it can only be considered completed when all the subcomponents have been executed. In a stable, observable system with free communication the allocation problem is relatively simple: it can be seen as an instance of the bin packing problem (see Yao, 1980, for examples of algorithms). However, the characteristics of the scenario described in Section 3.1.2 increase the complexity of the problem.

Agents need energy for both communication and task execution. If all energy is spent on tasks, the agent will be unable to communicate, thus disrupting the topology of the network. A fundamental problem is that agents have no global view of the network; yet their actions can have a global impact. The preferred allocation involves spreading tasks across the system in a manner that takes the cost of remote communication into account.

This dependency on energy led me to use two different metrics for allocation. Firstly, the number of tasks allocated is an obvious measure. However, if tasks have different sizes or priorities, this approach does not capture the whole picture. In addition, it is possible for two different systems to allocate the same number of tasks — in this case a second metric of energy efficiency is required. For two approaches that allocate an identical number of tasks, the one that requires a smaller energy overhead is regarded as better, because it has the *potential* to complete more tasks. In mission design terms, higher efficiency translates to lower cost, with smaller power systems and lower launch mass. Energy overhead is thus used to provide a more nuanced view of task allocation performance.

3.1.3.2 Robustness

As the system becomes larger and more diverse, the probability of unforeseen events in the network increases: agent failures, topology changes, changing work loads and new mission objectives, for example. We define robustness as the ability of the system to successfully allocate tasks, despite these disturbances. In measuring robustness, I therefore look at the allocation performance, in terms of tasks and energy, in the presence changes in the network structure.

3.1.3.3 Scalability

Network sizes will be in the range of tens to hundreds of nodes for the distributed satellite application, but larger systems are not impossible. To scale well in this range, the task allocation mechanism should function autonomously, while the total communication cost remains reasonable. Direct management of agents by humans becomes prohibitively complex and expensive in larger systems. The ideal would be a system that acts as a superorganism: despite the multitude of autonomous parts, the system as a whole is managed as a single individual. The detail of which module performs which component of the task should be abstracted away from the operators and decided transparently by the group of satellites.

The proportion of the energy budget devoted to communication is one of the most visible indicators of the scalability of the system. A greater number of agents can lead to more messages sent during negotiation; to avoid spending all energy on communication, the communication overhead needs to grow in sub-linear fashion with the number of nodes in the network.

3.2 Tasks

Before we can decide how to manage job allocation, we need to define a language we can use to discuss tasks. The term *task* is generally used to specify a unit of work that must be completed by the system, with different task types corresponding to different operations. Tasks can be decomposed into subtasks: atomic units that are executed by agents with different skills. I will assume that this decomposition is known *a priori*, but that the order in which tasks will arrive is not known. The work on task allocation in multi-robot systems by Zlot (2006) informed much of my thinking around the representation of tasks, although I will restrict myself to a simpler set of tasks. Instead of considering boolean relations between task components, I will use deterministic task definitions, where one task component always results in a specific set of further task elements.

3.2.1 Command flow example: building a house

The command structure of tasks can best be illustrated using a simple allocation example: suppose a man wants to build a house. As the owner, he does not know how to construct it, so he decides to contract someone to do the building. The building process can be split into three different components:

- laying the foundations,
- building the walls, and
- constructing the roof.

These task components must be executed in sequence, but the command structure can assume several different forms: one option using a single builder

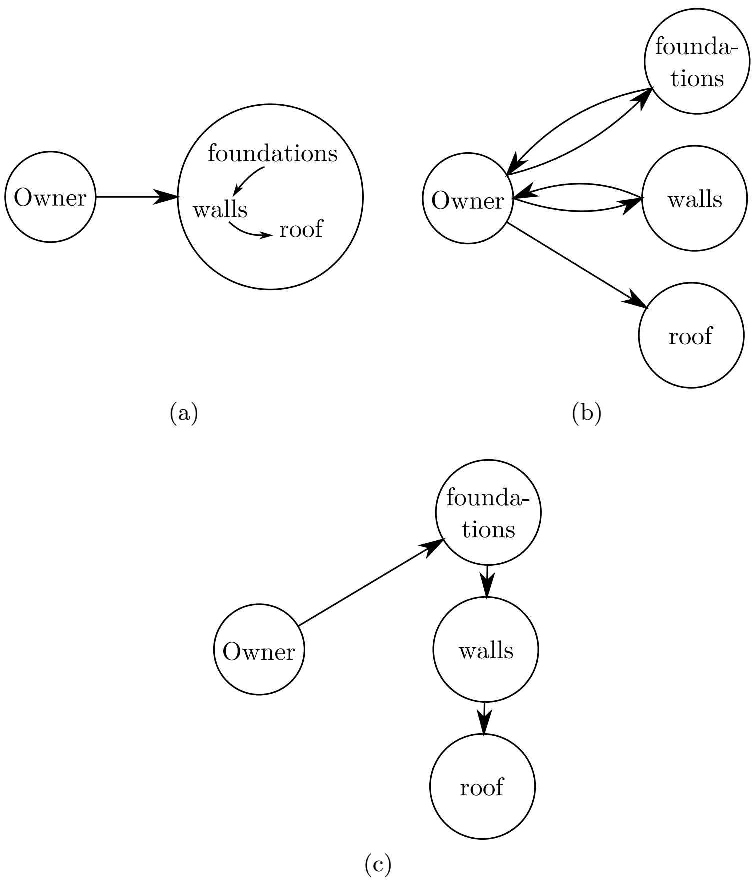


Figure 3.4: Graphical depiction of different command flows for the house-building example. In (a), the owner assigns responsibility for the entire construction to a monolithic entity with all the necessary skills. The case where the manager manages every component is shown in (b), while the recursive subcontracting command flow is shown in (c).

with multiple skills, and two alternatives which rely on specialists. When viewed in the satellite domain, the first case is very similar to using a monolithic satellite, while the last two cases are architecturally similar to the distributed satellite problem, where specialised units need to be coordinated. A graphical representation of the different command flows is given in Figure 3.4.

3.2.1.1 A single builder with multiple skills

In the first configuration, the owner approaches a number of builders to obtain quotes. He selects one based in a combination of cost and promises of quality, who then proceeds to build the entire house. Employing someone who is very good at all the aspects of building is very expensive though. This cost is largely due to the number of skills and amount of equipment the builder must maintain to be proficient in all aspects of construction. Most of it is underutilised — only a single task is performed at one time, yet all the builder's resources are allocated to one building project.

This approach is analogous to using a single, complex satellite: to be good at everything a highly advanced system is required, but the entire system is dedicated to a single objective. If the satellite is isolated and completely self-reliant, this can be justified, but if others are around who could share resources, a more efficient and robust allocation can be made.

3.2.1.2 Direct management

An alternative approach sees the owner contracting labour to different specialists. When the first contractor completes the foundation, the owner enlists a builder who specialises in walls, after that one who is good at constructing roofs. This works out less expensive than the single builder approach, because everyone in the chain of construction is only occupied for the portion of time where they are actually used — if another job becomes available after the foundations have been laid, the first contractor can start work there while the walls are still being erected. Different task components are therefore pipelined which allows a higher throughput.

The owner still acts as building manager throughout: he communicates all the information about the status of the foundations (e.g., dimensions, load capacity, etc.) to the wall builder, and again all the wall information to the one who constructs the roof. Although this allows a very high level of control, this requires significant time and effort on the part of the manager. He needs to select all the contractors and communicate with them in sufficient detail to serve as interface between them. The greatest benefit of this approach is that it offers fine-grained control over the construction process, however, this incurs significant overhead on the part of the owner.

If this approach is applied to managing a distributed satellite system, we will find a central manager spacecraft that can control a number of simpler workers. The manager can use global information to realise optimal (or near-optimal) system performance; however, the manager presents a single point of failure, and the information processing requirements will be significant in larger systems.

3.2.1.3 Recursive outsourcing

In the final control flow model the owner contracts the specialist in foundations to build the entire house. After laying the foundations, this specialist calls in a wall specialist he trusts and contracts him to complete the house. When his component is complete, the wall builder again finds a roofing specialist to complete the remainder of the structure. At every stage the responsibility for the task is transferred to the next person in the construction chain.

This approach utilises the skills of the individual builders to the full, both in a technical and an informational capacity. On the technical side, all builders can be specialists, which allows investment in their skills and equipment. In addition, because one type of task component is usually followed by another specific type, they learn about the next contractor in line: where they are, what information they need, and how much they will cost. This “local information” is potentially very useful: over time everyone builds up a model of their immediate neighbourhood, which is used in quoting for new jobs.

The result is a robust system, mainly due to the following three factors:

1. Individuals are modular, which makes them substitutable. The functionality required is procured on an ad hoc basis, with no commitment to or reliance on specific individuals.

2. There is no central control, therefore no single point of failure.

3. If demand for a new type of task were to arise, units can change their associations (the links along which tasks are outsourced), which is significantly easier than developing new competencies.

This model is frequently seen in complex manufacturing contracts. For example, a small engineering company is paid to build the test equipment that is used to verify an instrumentation dial, constructed by another company, for the cockpit of a passenger airliner. If the airliner constructor had to oversee component procurement at the lowest level, the amount of administration would completely swamp the whole construction process.

This distributed, self-organising command flow is very attractive when applied to the multi-satellite task allocation problem: it allows for specialised units, an adaptable execution flow, scalability and robustness. I will combine this approach with a labour market mechanism in 3.3 , but we first need to define a notation that allows us to discuss tasks.

3.2.2 Notation

Although relatively simple tasks, such as the construction example in section 3.2.1, can be adequately described using natural language, we are quickly pushed beyond the bounds of what can comfortably be expressed. Researchers have therefore used directed graphs (Kota *et al.*, 2009) or trees with logical operators (Zlot, 2006) to describe the transitions between task components. I have however found production rules, as used in context-free grammars, to provide a clear and concise description of tasks at a level of detail suited to this study. Note that this is primarily a descriptive tool: it contains the same information captured by directed graphs or task-trees. The main attractions of using a grammar are the brevity with which task structures can be described, as well as the ease with which complex tasks can be generated. The reader is not assumed to have any previous knowledge on formal grammars; the main objective of this section is to convey the manner in which task elements will be described in the rest of the thesis. If a more in-depth discussion to context free grammars is required Kakde (2007, Chapter 5) provides a good introduction.

At the highest level, we use a *task* to denote a sequence of operations that delivers a result. These operations are known as *task elements* or task components (the terms task elements and task components are used interchangeably in this thesis). The task elements can be seen as commands, for example "take picture" or "build walls". Every command has an action associated with it (the taking of picture, or laying bricks to form a wall), and sometimes one or more further commands that result from the action (e.g. "download photo data" or "construct roof"). These offspring task elements are usually associated with the transfer of control and of data that resulted from the previous action.

We can define the relationship between task elements formally using a context free grammar described by the four-tuple $G = (V, T, P, S)$, where:

1. V is a finite set of non-terminals that correspond to task elements (e.g., the command to execute a task element such as "build walls" in the house building example).

2. T is a finite set of terminals representing the actual execution of the task (building the walls).

3. P is the set of production rules that specify the relationship between different task elements (e.g., the walls have been completed, now the roof must be constructed).

4. S is a set[2] of start non-terminals, defining the tasks that are visible from a position external to the system. The command "build a house" is an example which then starts with the task element "lay the foundations".

In our task allocation case, non-terminals correspond to task elements, i.e., the *command* to do something; while the terminals represent the actual execution of the task. Task elements are indicated using capital letters (A, B, C, etc.); execution terminals use lower case (a, b, c, etc.). The production rules describe the tree of hierarchical dependencies.

In referring to a task, we therefore refer to such a grammar and the resulting execution tree. The possible tasks that can be executed by a system correspond to the set of starting tokens, S. Allocation of a task T_i results in the execution of one or more non-terminal task components (W):

$$T_i \rightarrow W$$

As our main focus here is on how one task element generates further task components, we restrict production rules to have the following structure:

$$X \rightarrow x\mathcal{S}$$

The execution terminal is the leftmost character of the generated string (x), optionally followed by one or more further task elements (non-terminals) in the string $\mathcal{S}$. This format is well suited to the outsourcing model used in this thesis: an agent executes a task element then passes the result on to one or more other agents for further processing. If an agent has the ability to execute a task x, we will refer to that ability as the *skill* x.

As an example, we can now describe a task that consists of the elements a, b and c which are be executed sequentially, such as the house building example above. The production rules for this task are given by:

$$\begin{aligned} A &\rightarrow aB \\ B &\rightarrow bC \\ C &\rightarrow c \end{aligned}$$

with $S = \{A\}$. The execution flow is graphically depicted in Figure 3.5.

A different task description might require a to lead to both b and c:

$$\begin{aligned} A &\rightarrow aBC \\ B &\rightarrow b \\ C &\rightarrow c \end{aligned}$$

[2]Here I deviate from convention in having multiple starting points; these correspond to different tasks.

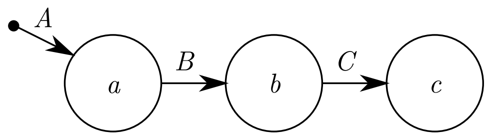


Figure 3.5: Sequential task execution, where task elements follow each other in a linear manner.

again with S = $\{A\}$. The resultant branching and parallel execution is shown in Figure 3.6.

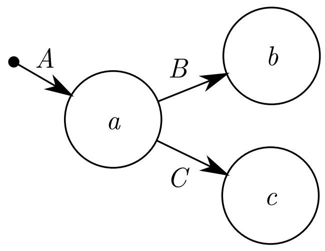


Figure 3.6: Branching task execution, which results in two tasks elements executed simultaneously.

Note that if we focus purely on the produced terminal sequences (*abc*) these two grammars appear equivalent. However, if we take into account that the execution of *b* and *c* in the second example effectively occur in parallel, the resulting string can be either *abc* or *acb*. It is therefore important to keep in mind that our interest here lies in the *transitions* that make up the task tree, because that is what needs to be allocated.

Using production rules can easily describe an execution tree, but it can also be used to describe more complex graph-like task dependencies where tasks converge to a point. If data from several nodes needs to be merged by a single unit, this can be accomplished by initially allocating a task that defines which node will serve as the convergence point. When the individuals that generate data allocate the subsequent task components, only the convergence node is seen as suitable, and receives all components. The skill set of the convergence node is therefore temporarily modified to allow convergence.

3.2.3 Task size

The last point in the description of tasks that needs to be defined is that of *task size*. Two tasks of similar structure can have different sizes: building a small cottage requires less work than a large villa. The size is an indication of the amount of resources (e.g., time, processing power, bricks, cement, fuel) required for a particular instance of a task type. When considering a task consisting of multiple components, I will assume the size relationship between the task elements remains constant. If T_2 is double the size of T_1, and both are instances of the task in Figure 3.5, the size of the task elements (a_1, b_1, c_1) of T_1 will be double that of the corresponding elements (a_2, b_2, c_2) of T_2. The work capacity of an agent is the sum of the task sizes it can execute.

When expressed in a satellite context, the task size reflects the amount of energy required to execute a task element. For example, using the execution flow in Figure 3.6, a task size of 1 will require 1 unit of energy to execute the a component, and another unit for each of the b and c components. The total cost of executing the task is therefore 3 units. The task size is set for every instance of a type of task: a different instance of the same type of task might have a size of 2, which will therefore result in a total energy expenditure of 6 units of energy. This convention focuses on the task execution cost for an agent, and is therefore independent from the complexity of the task tree.

3.3 Task allocation in a labour market

In previous sections I described an abstracted model of the system we are interested in and discussed tasks, skills and the task elements we use to define the process. With these prerequisites covered, we can now develop the task allocation mechanism. I will base my discussion on the dynamics of a human labour market. Why a labour market? The use of a social metaphor is not purely a pedagogical tool — I believe that when both systems are viewed from a task allocation perspective, they are remarkably similar.

Human markets have certain characteristics due to the environment in which they operate. Markets encourage specialisation to increase efficiency. Individual self-interest results in inherently decentralised systems: a large number of individuals can therefore take part. These individuals have limited information, but still manage to make decisions that are good enough and their interactions result in a highly dynamic system. Similarly, for task allocation in distributed satellite systems, we would like to handle large numbers of heterogeneous spacecraft, employ distributed control to enhance robustness, and work effectively despite real-time changes in task load and mission objectives.

The market system that evolved in human societies resulted in characteristics that are highly desirable for the satellite task allocation problem. I therefore propose to recreate a similar mechanism in our technological system, to benefit from the same dynamics. The command flow will be similar to the recursive outsourcing allocation described in section 3.2.1.3.

The discussion starts with a basic auction, which is then refined to more accurately reflect the abstracted system in section 3.1.2 by including the topological and associated communication constraints.

3.3.1 Basic auction

The first, and most abstract, example uses a basic auction to determine allocation. Picture a fully-connected network of agents, one of which is an auctioneer with tasks to assign, while the others are workers that will be assigned tasks. The agents are initially homogeneous: there is no topology to distinguish between nodes, they have the same skills and work capacity, and therefore all are equally capable of completing tasks.

A single-bid reverse auction is used to allocate tasks. To initiate the allocation of a new task, the auctioneer *announces* it to the network. For the moment, let's assume the task consists of a single subtask:

$$A \quad \rightarrow \quad a$$

The auction announcement is most efficiently communicated using a broadcast message to all the workers, thereby utilising the one-to-many broadcast channel between agents. The workers return bids that communicate their suitability for the allocation. The auctioneer then selects the lowest bid to identify the best agent for the job and transfers the task to it. When the agent receives the task, its work capacity decreases, but it is also rewarded financially. We will use capacity as a general term to describe the availability of an agent in terms of time and resources.

This approach relies on bids reflecting some system cost: if the auctioneer assigns to the lowest bidder the global cost is minimised. But how should worker agents calculate their bids? As our objective is to maximise the total system utilisation as discussed above, the desired allocation would distribute tasks evenly across the available workers. One way of achieving this is by having the bids reflect the available capacity of the agent and the size of the task:

$$\text{bid} \quad = \quad \text{size} \times \text{capacity}^{-1}$$

This causes a task to be allocated to the agent with the greatest capacity: idle agents (with capacity at 100%) will place lower bids than agents with lower capacity (due to previously assigned tasks). Initially, all agents are equal, so allocation can go to anyone. The agent that receives the task has its capacity decreased, so in the next allocation round, someone else will receive the task. As a result the allocation resembles round-robin assignment, where all agents are evenly utilised. Note that in economics literature, the terms “quote” or “ask” would be preferred to “bid”, as this is a reverse auction. However, to maintain consistency with the terminology used in the definition of the Contract Net protocol (Smith, 1980), I will use “bid” to indicate the price quoted by worker agents.

If we were to start with a range of initial capacities, this approach will start by assigning tasks to the highest capacity workers first. As their capacity is decreased to the level of other workers, these workers will also be assigned tasks: having the bid structure communicate work capacity encourages equalisation of resources across the network. Large tasks require more energy to complete, the quoted price will therefore be scaled accordingly.

Note that the workers are not trying to outbid each other or mislead through incorrect information. They are selfish in the sense that they try to maximise a private utility function, but also benevolent in the sense described by Rosenschein and Genesereth (1985). Despite their benevolence the agents are all still trying to maximise the amount of work they receive, but only by staying within the defined rules of interaction. The differences between this cooperative market and one where agents are truly selfish, with no regard for system-level performance, will be explored in more detail in Chapter 7.

If more than one type of task needs allocation, we will need a community of workers that have the necessary skills, but the allocation mechanism itself is still valid with the added condition that workers only bid on tasks that they are capable of completing. This basic auction is illustrated in Figure 3.7 for a task of type a. Note how the bid value abstracts the network information: the network topology, skills of workers and their capacity are all communicated in the bid price.

We now expand the scenario to a multi-component task, which is sequentially assigned to different agents. To determine the next allocation, an agent can assume the role of auctioneer, eliminating the need to transfer the task back to its origin. By allowing any node to be an auctioneer, this approach is equivalent to the recursive outsourcing example in section 3.2.1.3.

The outsourcing costs of the future task components need to be taken into account when bidding. The resulting bid calculation now assumes the form:

$$\text{bid} \quad = \quad (\text{size} \times \text{capacity}^{-1} + \text{outsourcing cost})$$

To determine this outsourcing cost the workers could perform a dummy auction,

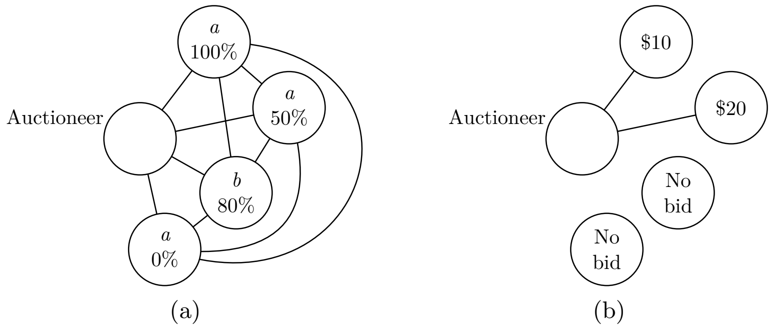


Figure 3.7: The view of the network that results from the basic auction. In (a) the network topology, skills (*a* or *b*) and capacity (in percentage) of the workers are given. (b) represents the auctioneers perspective of the network, where all the information has been abstracted to bid values.

where potential contractors state what they will bid; however, it is expensive from a communication point of view and not guaranteed to be accurate when allocation eventually happens. Alternatively, and more simply, the worker can use his history of outsourcing attempts to estimate the expected outsourcing cost. The reduced communication and faster turnaround time makes this the preferred approach in my model.

3.3.2 Spatial distribution

We now increase the complexity of the problem by introducing a network topology that limits agents to only communicate with a subset of the other agents in the system. The greatest difference between this scenario and the previous one is that an auction will not reach all workers involved. Instead, messages need to be relayed through other agents.

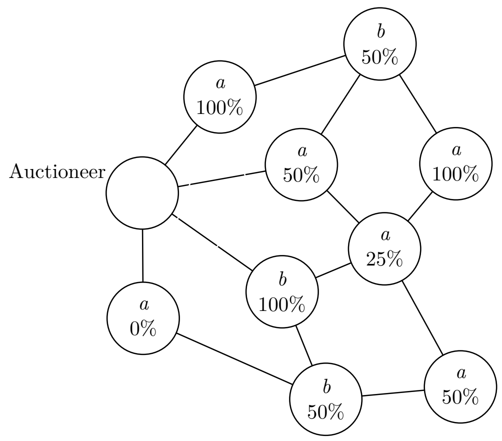
Imagine an auction, as described above, that fails because none of the workers can accept the task, either due to insufficient capacity or due to lacking the requisite skills. The task could be discarded as impossible to assign, but instead the scope of the auction is enlarged: everyone notifies a few of their friends about the pending task. Some of these friends are available, so the intermediate workers return to the auctioneer with a solution: they can facilitate the allocation of the task by acting as middlemen. As a result the auctioneer will now receive valid bids, despite none of his direct neighbours having capacity available.

Two routes can be followed from here. In many human examples, a meeting between the auctioneer and the prospective bidder would be arranged to complete the allocation. This can be seen as creating a new connection between these agents. The second route involves cases where connections cannot be changed quickly, or creating the link is expensive (for example, when new equipment must be acquired and people trained). In these cases the intermediate agent serves as a conduit between the auctioneer and the successful bidder.

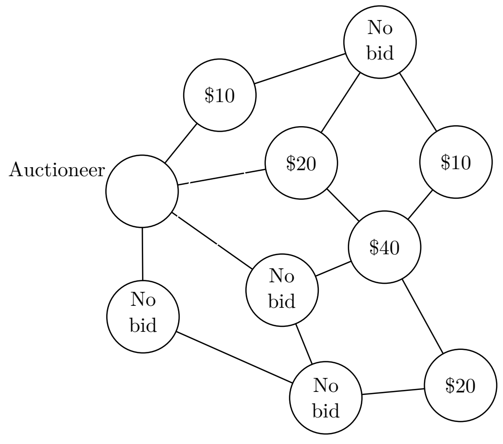
We can take it even further if we assume that the auctioneer is not interested in who is executing the task, merely in assigning the task. The auctioneer then treats the facilitating neighbour as if he was the final task destination, transferring both task and payment to him. The facilitator subsequently transfers the task to the actual bidder. The usage of a middleman is not limited to only one intermediate hop; we can expand it to form allocation chains across the network. This approach is very attractive, as it allows us to abstract away the topology of the network, thereby simplifying processing and allocation. Note that this allocation process is *task-centric*: the objective of the auctioneer is not to assign work to someone specific, instead he simply wants the work to be completed. The allocation process in a spatially distributed auction is shown in Figure 3.8, (a) to (c).

The relaying of auction messages has improved the allocation of tasks, but has also resulted in increased communication. As stated before, communication decreases the capacity of agents to do what we really care about, namely complete tasks. The first aspect of communication cost is on an agent level: workers now spend time and resources facilitating deals that do not benefit them, not even if the task is successfully allocated due to their effort. The second problem is that the relaying of messages makes allocation to distant nodes possible. While it is desirable to have this functionality, we would also like the auctioneer to be capable of distinguishing between nearby and faraway workers, because local allocation has a smaller system-level energy cost.

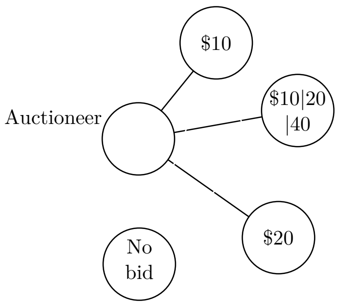
Human markets have, however, encountered these problems before. To account for the negotiation expenses, agents charge a commission fee for facilitating successful allocations. The commission addresses both the above concerns: the intermediate agent is rewarded for resources committed to facilitating allocation, and the auctioneer has an incentive to prefer nearby nodes because distant nodes are more expensive. The effect of commission on the bid values of Figure 3.8 is shown in Figure 3.8d. Note the difference in received bid values: without commission the \$10 bids appear equivalent, even though the more distant node will result in greater communication cost. With 10% commission, however, the auctioneer will prefer the closer node.



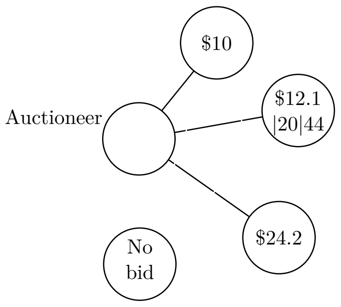
(a) Network topology and node states.



(b) Initial bid values.



(c) Received bid values, no commission.



(d) Received bid values, 10% commission.

Figure 3.8: Auction across a network without commission as seen by auctioneer. The true state of the network is shown in (a) with the corresponding bids calculated by workers for a task of type *a* in (b). The auctioneer has no topology information, instead it only receives a number of bids from its neighbours which includes their own bids, and bids relayed from more distant nodes. This is shown in (c) for the case without commission, while (d) again depicts a system where all nodes add 10% commission to the bids they relay. Note the differences in relayed bid values in the last two cases.

Not all instances of nodes acting as intermediaries result in successful allocation however; lower cost assignment to a different node is frequently possible. The human response is to charge a generous commission: if task allocation is successful, the facilitator receives enough money to survive a number of failed attempts. Different commission calculation strategies will be discussed in more detail in the following section, when I describe the simulation model.

The use of intermediate agents can be seen as a streamlined version of a repeated auction to do the allocation. The first auctioneer would assign a task to one of his neighbours, but as the neighbour does not possess the necessary skills to perform the task, he initiates his own auction to find a suitable candidate. This process is repeated until the task reaches a viable destination. Of course, such a model relies on the intermediate auctioneers to know enough about the network to believe they can successfully outsource the task. If we treat these intermediate auctioneers as facilitators, they fulfil the same function, but with a much lower communication expenditure.

3.4 Simulation model

The labour market described above forms the abstract model at the base of the design tree in Figure 2.2. It is a general description of an allocation process which could potentially be mapped to a number of related scenarios. Whereas many mechanism design problems focus on forcing agents to be truthful, the problem here is subtly different. As agents are benevolent, they are also assumed to be truthful. However, as system managers we do not have reliable access to their utility valuations, i.e., how much energy they have for completing tasks. In addition, we cannot assume that their estimates of outsourcing costs will be correct, because these are based on previous observations, not the current state of the network. Instead of addressing various agent strategies, our problem centres on the access to information in a dynamic environment with expensive communication. While I am interested in a distributed satellite system application, I also need to be aware of the families of systems that are pruned off as the design increases in specificity. As a first step, a realistic implementation of the allocation mechanism must be defined. This forms the simulation model used in the rest of the thesis, and also serves as an implementation reference for physical systems.

The labour market above is used as a metaphor that inspires the solution to the multi-satellite task allocation problem. I describe the complete model here to provide a single, comprehensive overview of the task allocation system. In the following chapters I will explore particular components of the system while assessing the allocation mechanism. At this point I would like to reiterate that the validity of the mapping between these two scenarios is the result of the similarity between them. Communication, topology and task constraints are similar in both cases, because the task allocation problem should be seen as much broader than the specifics of the respective domains.

A discrete event simulation is used to model the system: satellites are represented as independent agents, each with a set of skills that correspond to the types of tasks it can execute. Agents measure their capacity (i.e., the amount of work they can perform) by their available energy. They are spatially distributed to form a network with local communication links between nodes. Tasks can originate anywhere in the network, and can only be allocated to nodes with sufficient energy and the correct skills.

We have already identified the energy spent on communication as a major constraint on the work completed by the system. To accurately model this, the simulation has to start on the packet level: every transmission must be tracked to be able to measure communication cost. All packets require a finite time to propagate between nodes, and all packets use energy for transmission. The packets are built into a routing framework, upon which the negotiation and allocation logic is constructed.

3.4.1 Task allocation

The auction protocol for a single task component is described using a message sequence diagram in Figure 3.9; the various steps and associated actions will be explained sequentially.

When an agent becomes aware of a task element that needs to be allocated, it sends an auction announcement message to all nodes in the network through flooding. If an agent receives more than one auction announcement packet for the same auction due to cycles in the network topology, only the first packet is propagated. This constructs a spanning tree across the network which will be used for routing, as discussed in Section 3.4.4.

Relevant nodes, i.e., those with the appropriate skills and enough energy, calculate bids that reflect their fitness: the bid value (B) is based on the ratio of maximum (e_{max}) to remaining energy (e_{rem}), plus the expected outsourcing cost (c_{os}), as discussed in 3.3.1. A scaling factor is applied to take the size of the task component into account (z). Note that the task size also applies to the size of the outsourced task components; the outsourcing cost is therefore also scaled by z as discussed in Section 3.2.3.

$$B = z\left(\frac{e_{\mathrm{max}}}{e_{\mathrm{rem}}} + c_{\mathrm{os}}\right) \qquad (3.4.1)$$

This cost function communicates the available resources of the agents: under-utilised agents will place inexpensive bids, while those that receive frequent allocations increase their bids as their available energy decreases. By including the expected outsourcing cost, information about the network in the vicinity of the bidding agent is also communicated. Agents are benevolent: our objective is to control the system by transmitting the minimum information; we are not modelling bidding strategies for a competitive real-world market. Bids are routed back to the auctioneer along the path of the original auction announcement.

The auctioneer assigns the task component to the agent with the lowest bid and transmits an allocation-offer message. If the winning agent wants to accept the offer, it returns an acknowledgement message, upon which the auctioneer transfers the task and payment. If, however, the agent has changed state since bidding by accepting another task component from a different source, it transmits a negative acknowledgement to the auctioneer, who will then repeat the auction up to three times. If no suitable agents exist, e.g., due to several node failures or insufficient node energy, allocation will fail.

The outsourcing cost is calculated by averaging over the prices paid in the last five outsourcing events, i.e., where the node that is currently bidding previously acted as an auctioneer for the corresponding task type. Of course, many other learning rules could also be used instead of this moving average filter — the main desirable characteristic is that it rejects noise, while responding quickly enough to changes in the system state. The moving average filter was easy to implement and exhibited satisfactory performance. To bootstrap the learning of the outsourcing cost, the history was populated with initial values of 1. This is equivalent to assuming that the task can be outsourced to an adjacent node with full capacity, i.e., the lowest cost outsourcing case. This value is also low enough to ensure that the node does not exclude itself from receiving an initial task by quoting too high a price. When a task is outsourced, this value is updated to reflect the learned cost.

This strategy would work smoothly when a system contains a single auctioneer. However, multiple simultaneous auctioneers exist in our system: tasks can originate with any agent, multiple tasks will be in the system at any one time, and tasks will of course be passed around the system as each component is executed. With multiple auctioneers we potentially find conflicting allocation: the best available node will be allocated several different tasks concurrently.

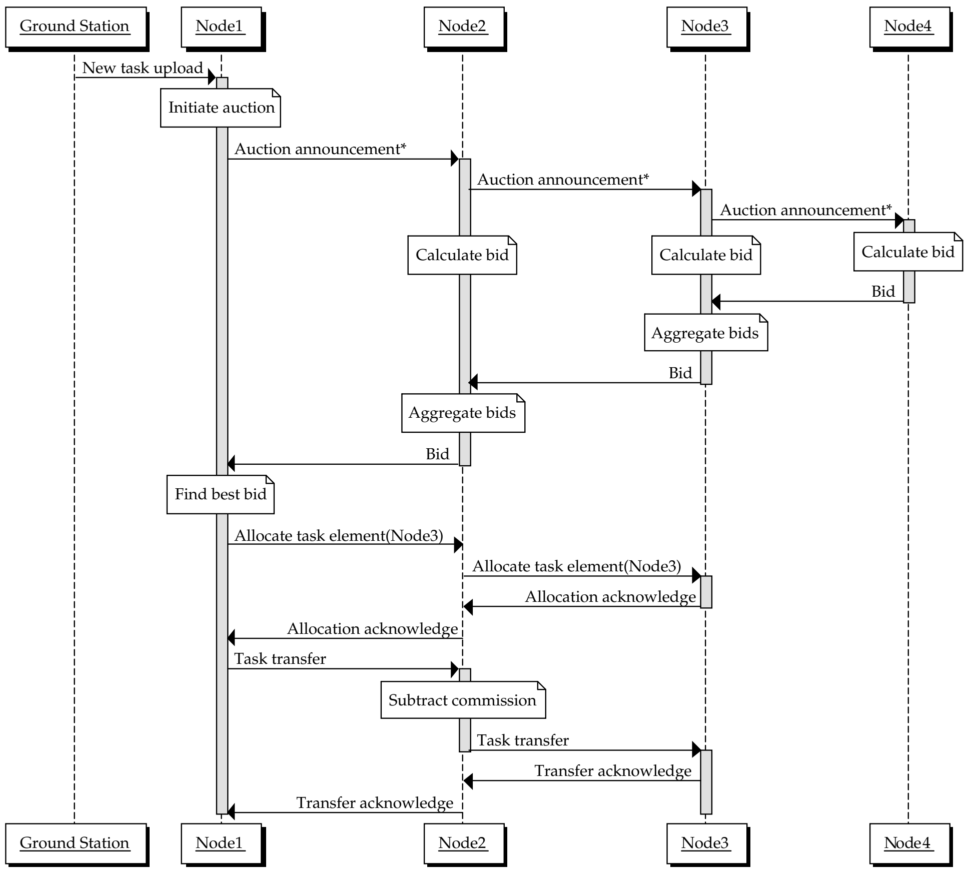


Figure 3.9: Message sequence diagram describing the market-based task allocation flow, with time increasing from top to bottom. Node 1 acts as auctioneer, while Node 3 is the successful bidder. The auction announcement messages are flooded through the network; nodes capable of executing the task respond with bids that convey their suitability. Bids are aggregated on the return path: only the best bid is forwarded. The task is allocated to the lowest bidder and the payment and task are transferred.

Furthermore, this strategy does not take the topology of the network into account. If the auctioneer receives equal bids from two different nodes, he has to decide between them. From a global efficiency point of view, we require the auctioneer to prefer allocation to the closer node, but there is no way of determining how far away they are.

To remedy these two concerns we need a localising force that will counter the tendency to allocate all tasks in the system to the agent with the highest energy level. Thus, when an agent relays a bid to the auctioneer the bid is increased by a constant *commission* factor. Bids from far away will therefore appear more expensive to the auctioneer, causing it to favour nearby nodes. The resultant bid that is received by the auctioneer can be expressed as

$$B_{\text{rx}} \quad = \quad z\left(\frac{e_{max}}{e_{rem}} + c_{\text{os}}\right)(1+k)^{d_{\text{bid}}} \tag{3.4.2}$$

where k is the commission value (between 0 and 1) and d_{bid} the number of hops the bid has been relayed. Note that, for $k = 0$, distance does not make a difference to the received bid value. This approach does not completely eliminate allocation collisions, but does reduce their occurrence substantially.

Note that commission does not necessarily have to be multiplicative — I chose it based on the similarity to real-world financial transactions, where commission is often levied as a percentage of the transaction cost. To best capture the system-level energy consumption, additive commission could also be used. However, for the transfer distances used in my experiments, the difference between the two approaches is minimal.

In terms of communication cost, the negotiation packets (auction announcement, bid, allocation and acknowledgement messages) are all relatively small, in the order of tens to hundreds of bytes. Task transferral usually involves more data, however, as the state of the task and any associated data must be transferred. For example, if the task at hand involves the merging of images from different sources, the image data needs also to be transferred, which results in orders of magnitude more data than the negotiation packets. In the cost calculations in subsequent chapters I will therefore distinguish between the cost of negotiation packets (c_{tx}) and the cost of task transferral (c_{tf}). To relate these two values, we can express c_{tf} as c_{tx} scaled by a constant factor (α):

$$c_{\text{tf}} = \alpha c_{\text{tx}}$$

α therefore defines the difference in energy cost between negotiating an allocation, and transferring the task for execution.

3.4.2 Optimisation

While the above scenario is promising in its task allocation ability, we can refine it to more efficiently utilise the communication resources available. I achieve this by limiting auction range and by aggregating bids.

The case for limiting the auction range is argued first. If an auctioneer wants to allocate a task in a very large network with n workers, an auction announcement will be propagated to all individuals (n messages). A subset of these agents will respond, but in the worst case all n. Their bid messages are relayed back to the auctioneer, resulting in up to $\frac{n(n+1)}{2}$ messages. The total number of messages sent in the negotiation phase is therefore proportional to n^2. However, due to the high commission on long distance allocations, local workers will be preferred. We can exploit this preference for local allocation by limiting the propagation distance of the auction message through a time-to-live (TTL)[3] mechanism, without significantly impacting on the allocation performance. The set of workers participating in an auction is called the *auction community*, which covers an area up to d_{ttl} hops away from the auctioneer. The number of workers participating in an auction will be n', where $n' < n$, with the exact value of n' determined by the network topology and d_{ttl}.

The second improvement involves aggregating bids to only relay the best bid. If we refer back to Figure 3.8d, note that the auctioneer will receive multiple bids from neighbouring workers that represent more distant bidders. The relaying of these bid messages results in the n^2 term identified above, which in turn is the greatest contributor to the total communication cost. If workers aggregate bids and only forward the winning bid for their local sub-trees, the n^2 factor is replaced by n, drastically cutting the communication overhead. A more detailed derivation of the communication complexity is given in Section 5.3.1.

Note the similarity between the aggregation of bids and data aggregation used by Intanagonwiwat *et al.* (2003). In both cases only the required information is transmitted to the destination: the auctioneer only needs to receive the winning bid to make an allocation. An additional benefit of bid aggregation is that processing is spread across the network, thereby eliminating scaling problems in that regard.

3.4.3 The global view

When the allocation process executes concurrently across the network, we find that the system self-organises into zones of allocation around auctioneers allo-

[3]The term time-to-live is derived from the equivalent TTL mechanism used in internet protocol packets.

cating similar types of task components. The sizes of the zones are determined by the distribution of bidding agents, their individual energy levels, the number of tasks injected into the system, and the topology of the network. This auction mechanism is equivalent to a reverse sealed-bid auction, but it is truly distributed. Not only do all agents have to act as temporary auctioneers, but all agents in the auction area help to calculate the winner.

The topological constraints on information exchange is similar to the discussed in Robinson (2002) — the agents that make up a market are physically co-located. One significant difference between Robinson's work and mine is that my allocation mechanism uses the relaying of bids to increase the size of the auction community, while he treats the auction community as a given with no explicit concept of communication cost. This means that my allocation approach will cope better with sparse networks, where the local community is very small, because relayed bids provide an efficient mechanism for enlarging the auction community.

Currency is primarily used as a communication metaphor; no actual money changes hands, nor can nodes go bankrupt or get rich. Our interest lies in the management of the entire system, not in the success of an individual agent (although agents do try maximise utility by allocating to worker that quotes the lowest price). This view of prices for communicating the state of the system recalls Hayek's concept of price signals: the change in prices in a free market communicates information to agents in the system which allows them to solve the distributed allocation problem, even without knowing exactly what it is (Hayek, 1945). In our case the price signals communicate the availability of resources (agents with skills and enough energy), which in turn determines the allocation patterns. If one area is exploited, prices will increase, causing allocation to shift to another area. The allocation auction can therefore also be seen as a particular type of negotiation mechanism.

3.4.4 Task-centric routing

This allocation mechanism makes no assumptions about the underlying routing architecture and can therefore be implemented in a number of different ways. However, the ad hoc information generated during an auction can be used to implement a novel task-centric routing protocol that is closely integrated with the task allocation mechanism.

In the majority of communication networks, routing is address-based: packets are sent from someone with a unique identifier, to another node with a different identifier. To relay communication between these individuals, intermediate nodes use routing tables that describe how messages should be forwarded. The information in these routing tables needs to be maintained to allow effective packet delivery, which complicates network management in dynamic topologies. For this reason, I propose the use of a task-centric routing for distributed satellite systems; this is supported by the data-centric routing approach proposed by Gnawali *et al.* (2005) for inter-satellite networks.

When it comes to task allocation we are no longer bound to using identities. If an auctioneer wants to allocate a task, it is concerned with the capabilities and resources of the potential bidder, not its name. In my task-centric routing scheme communication is routed according to the task it is associated with, which greatly limits the overhead. Paths are established on an ad hoc basis: under normal conditions this would be prohibitively expensive, but as the required paths can be constructed using information visible during auctions, it has no additional cost in the system. When auction announcement messages are flooded through the network, a node will only forward the packets if:

1. it has enough energy to transfer a task,
2. the announcement message time-to-live has not been exceeded, and
3. the node has not forwarded an announcement packet for the particular auction, i.e., the task identifier associated with the packet has not been seen before).[4]

Packets will therefore propagate across the network over nodes with sufficient energy to ensure reliable communication, forming a spanning tree across the network. By only forwarding one packet for a particular auction, cycles in the routing network are avoided.

In terms of implementation, a task-centric approach to routing can be realised if all nodes maintain two tables. By indexing these tables with the task identifier, the next hop for a packet can be determined as the packet travels through the network. The first table is used for "upstream" routing: it contains the next hop towards an auctioneer. When an auction announcement is flooded across the network, the address of the transmitting neighbour is added to this table, along with the task identifier. When bids and acknowledgement messages are relayed, the node selects as destination the neighbour specified

[4]An alternative propagation rule relays packets if the path they followed to from the auctioneer to the node is via higher capacity intermediaries than that followed by previous announcement packets, with a trade-off for distance. This could help the system performance by distributing communication to under-utilised nodes. In my simulations I only used the shortest path propagation, as preliminary experiments did not suggest a significant improvement for more advanced route construction approaches.

in this table. The second table is the downstream routing table. Entries are added to this table when the node relays a bid back towards the auctioneer. Bids that are not relayed are not entered into the routing table. If an allocation message is received, a node can use the downstream entry of the corresponding task to relay it to the next neighbour, because bid aggregation only requires the winning bid to be transmitted. The same applies to task transferral messages.

Information is only stored for the duration of an auction; the maximum sizes of these tables are therefore fairly small, and the node can safely remove routing information when an auction is complete. The sizes of the routing tables are largely determined by the number of simultaneous auctions in which the node is involved, which in turn depends on the network topology and d_{ttl}. As a result, the processing requirements on intermediate nodes are minimal: the tables are indexed using unique task identifiers, and data is only written to the table on specific packet events.

This routing approach shares a number of similarities to the data-centric routing approach used in directed diffusion (Intanagonwiwat *et al.*, 2003): nodes do not need to know the topology of the network, nor what other nodes are out there. Routing tables are constructed using only local information — the auction announcement and bid messages provide enough information for up- and downstream routing. Whereas directed diffusion constructs a routing landscape in which the gradient shows the shortest path to the sink node, my routing mechanism aims only to maintain a route between auctioneer and bidders until the task is allocated.

3.4.5 Assumptions and limitations

As with any model, a number of assumptions are made in the abstraction process. The model is valid for part of the parameter space for which these assumptions hold. We need to clearly identify these assumptions before exploring the model in any detail.

- The allocation mechanism assumes that a single task component needs to be allocated per auction. If multiple tasks regularly needed concurrent allocation multiple auctions would suffice, but a more efficient mechanism could be designed.

- Communication is treated as deterministic and pair-wise bidirectional: two nodes will communicate successfully if they are within range of each other. In reality though, wireless communication is more accurately modelled as a stochastic process. This will need to be considered if systems with unreliable communication systems are investigated.

- Sufficient bandwidth is assumed to be available to not warrant explicit consideration in the model. The number of transmitted packets is fairly low, confirming the validity of this assumption. This simulation should therefore not be seen as an accurate model of systems with very low bandwidth.
- This task model assumes that agents can estimate their own internal energy cost in completing a task — this information is used in the when bidding for task components. This is not unreasonable, as the resource cost associated with a task can be characterised quite accurately. Note, however, that agents still have to estimate their outsourcing costs, as this information is unknown at the time of bidding.

3.5 Discussion

In this chapter I have described the development of a task allocation mechanism inspired by human labour markets. A scenario of a distributed satellite system was used as a starting point to build an abstracted multi-agent model. A subset of the characteristics of this model is shared with other multi-agent allocation problems, namely multi-component tasks, expensive communication, limited energy, heterogeneous agents, a spatially distributed network, node failure, and a volatile topology. The objectives of maximised allocation, robustness and scalability were identified. Tasks were defined, and a task allocation model described in terms of a labour market. This market serves as an inspiration to the proposed task allocation mechanism for distributed satellite systems, which was described in the final section.

Adam Smith identified markets as encouraging the division of labour and resulting in specialisation of skills which leads to increased productivity. In this system I invert the causal relationship between these elements: specialised agents are a given, as they can increase the efficiency of the system. We then define a market-like mechanism to handle the allocation of labour. The resulting system exhibits the desirable characteristics of "natural" markets: efficiency, robustness and scalability.

Although not explicitly designed as such, this system shows some similarities with the contract net protocol (Smith, 1980) as well as the various other market-based allocation schemes proposed over the years. The key distinguishing features of my allocation scheme are the following:

- The use of a spatially distributed auction to allocate tasks. The auction mechanism is fully distributed, with nodes across the network involved in calculating the winning bid.
- The inclusion of communication cost as a significant factor in system efficiency.
- A novel commission parameter is used to localise allocation, and balance energy expenditure between communication and task completion. This does not require knowledge of the network size or topology.

In the following chapters I will further explore the behaviour of the allocation mechanism.

4

Behaviour of the task allocation mechanism

The previous chapter presented the design process for the task allocation mechanism to be used in distributed satellite systems. This model is fairly complex, with a number of interdependent parameters determined by both the application (e.g., communication cost) and the allocation mechanism (e.g., d_{ttl}). Before I explore the parameter space, it is necessary to discuss the basic dynamics of the allocation mechanism. In this chapter I focus on the behaviour of the task allocation mechanism in a number of basic scenarios. The experiments confirm the suitability of the allocation mechanism for distributed satellite systems, but also help us to understand *how* it works, thus providing a foundation for understanding the more complex dynamics we will encounter in later chapters. Thus, in order to gain familiarity with the allocation mechanism, this chapter looks at both the way tasks are distributed out across the system and how the it responds to node failure.

4.1 Task allocation

The primary objective of the system is to achieve effective allocation of tasks. As discussed in the model design section, this requires that tasks be distributed across nodes in a manner that is sensitive to the available energy of individuals, while also taking the network topology and communication cost into account. In this chapter I explore the qualitative behaviour of the allocation mechanism through simulation.

4.1.1 Single auctioneer

As the behaviour of the system is our primary concern, I restrict the initial scenario in a number of ways that make it easier to follow the allocation process. The simplest allocation example is the case where we have an auctioneer at one end of a linear network, as shown in Figure 4.1. The other nodes act as homogeneous worker agents, with the same initial energy and skills. The tasks

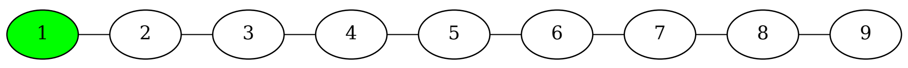


Figure 4.1: Linear network with one auctioneer, every node represents an agent. The shaded node (1) acts as auctioneer and allocates tasks to the other nodes, who all have the same capabilities. The task allocation mechanism favours nodes that are located closer to the auctioneer, due to a combination of the commission parameter and the effect of transmission cost.

are single element tasks, with the structure:

$$A \to a$$

Note that although any node could theoretically act as an auctioneer, this simplified task structure means only the first node will allocate tasks. The network topology is static, without node failures or mobility. Communication has a fixed cost of 0.001 units per negotiation packet ($c_{\text{tx}} = 0.001$), while task transfer messages have a cost of 0.1 units ($c_{\text{tf}} = 0.1$). One task is allocated every 100 time steps, using the auction procedure described in the previous chapter. Tasks initially have a size (z) of 1, at $t = 5\,000$ time steps the task size is increased to 2, thus increasing the load on the system. Commission is set to 5%, i.e., $k = 0.05$. Nodes have a maximum energy capacity of 10 units, if it falls below this level, it regenerates at a rate of 0.003 units per time step per node. Three distinct states can be discerned if we observe the node energy levels over time, as shown in Figure 4.2.

In the *settling* phase ($t < 1\,000$), all the worker nodes start with the same energy, and will therefore initially calculate the same bid value (B). As there is no outsourcing cost, equation 3.4.1 reduces to give the bid value as:

$$B = z\left(\frac{e_{max}}{e_{rem}}\right) \tag{4.1.1}$$

Bid values therefore communicate the energy available to a specific agent. However, the value received by the auctioneer is modified by the commission parameter, which increases the cost of relayed bids with every hop:

$$B_{\text{rx}} = z\left(\frac{e_{max}}{e_{rem}}\right)(1+k)^{d_{\text{bid}}} \tag{4.1.2}$$

The auctioneer will therefore allocate the task to the nearest agent (Node 2 in Figure 4.1). This process is repeated with further tasks and other nodes; changing energy levels will mean that occasionally a distant bid is more competitive than a local one. The net result is that the allocation tends to cluster around

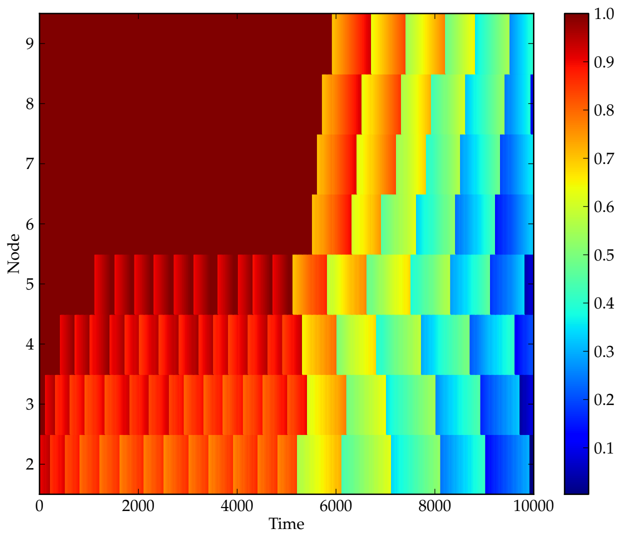


Figure 4.2: Normalised energy levels over time for worker nodes in network shown in Figure 4.1, the auctioneer is not shown. One task is allocated every 100 time steps, at $t = 5\,000$ the size of the task is increased. The step changes in energy show where tasks are allocated, nodes' energy levels recover slowly over time. Three distinct allocation phases can be observed: settling ($t < 1\,000$), steady state ($1\,000 < t < 5\,000$) and overload ($t > 5\,000$).

the auctioneer. The strength of localisation is determined by the commission parameter: a high commission (more than 20%) will concentrate tasks more strongly around the auctioneer, a lower commission results in a flatter allocation over a greater number of nodes. This settling phase is usually observed as a transient at the beginning of a simulation, or when task allocation is bursty.

The system reaches *steady state* when the amount of energy being added to the system (e.g., satellites recharging their batteries using solar energy) is equal to the energy being expended on work and communication. In this state the tasks are allocated to nodes close to the auctioneer with an almost uniform distribution. The energy consumption of the closer nodes is still greater than the ones that are further away. This is because the closer nodes have to expend a significant amount of energy relaying messages and transferring tasks to the more distant nodes. The number of nodes in the allocation area is enough to allow the incident (incoming) energy to match the cost of task execution and communication. The unused nodes remain fully charged and cannot store any of the incident energy.

An *overload* condition can be caused if the workload exceeds the energy

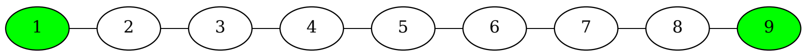


Figure 4.3: Linear network with two auctioneers (nodes 1 and 9). Tasks are allocated from both ends of the network to the worker nodes in between. All workers have exactly the same capabilities (except for position); the resulting allocation pattern sees Node 1 utilising workers 2 to 4, while Node 9 allocates to the workers at 6 to 8. Node 5 accepts an approximately equal number of tasks from both auctioneers.

added to the system: all nodes will eventually be fully exploited. However, as the node energy levels decrease, the $\frac{e_{max}}{e_{rem}}$ term grows dramatically, which in turn decreases the localising effect of commission. As a result, we observe a flattening of the allocation distribution as the mean node energy decreases. The number of active nodes is therefore maximised for as long as possible. If a node's energy falls too low, it cannot accept more tasks or communicate until it has recovered sufficiently. This condition can be seen in Figure 4.2 where $t > 5\,000$.

4.1.2 Multiple auctioneers

To demonstrate the interaction between multiple auctioneers, I briefly discuss the allocation pattern that results from having two auctioneers at either end of a linear network, as shown in Figure 4.3. The auctioneers allocate tasks at the same rate. The other parameters of the example in Section 4.1.1 are kept the same.

Two auctioneers cause tasks to arrive in the system from either end. The allocation dynamics are very similar to the single auctioneer case, with allocation close to an auctioneer preferred. However, we observe that two zones of influence form around the auctioneers: the one on the left only allocates tasks up to the node in the centre of the network, while the area on the right receives tasks from the other auctioneer. If a node is positioned exactly in the centre between the auctioneers, allocation alternates between them (this assumes a regular rate of task allocation that is the same for both auctioneers).

Note that there is no direct communication between the auctioneers. No global knowledge is used to calculate this partitioning, nor does the centrally located node notify either of them of the other's existence. Instead, it emerges from the low-level interaction between the nodes. A feedback loop is formed by basing bids on the available energy: a node that receives work regularly is less likely to be allocated more tasks. The bid values effectively communicate the state of the network to the auctioneer. The shape of the partitioning is determined by the distribution of available skills, the energy levels of the nodes in the vicinity, and the task load of the auctioneer. Variations in these parameters will result in changes in the size and shape of the partitioning. For example, if one auctioneer has twice as many tasks to allocate as the other, its zone of influence will be significantly larger than the other's. The dependence on network topology also means allocation can adapt rapidly to failure of a node. Note that the auctioneer does not keep a map of the agents in his area, instead their energy levels serve as a distributed memory of their utilisation.

4.1.3 The role of commission

This thesis introduces commission as a useful parameter to abstract the spatial properties of the network into a single value that can be used to influence allocation. In the discussions above, I have already highlighted the localising force of commission: it encourages local allocation by making relayed bids from distant nodes more expensive. This can decrease the communication required in task allocation and transfer.

Another effect of adding commission to bids is that it breaks symmetry ties. If two tasks of the same type are auctioned simultaneously to overlapping auction communities, the fittest agent will place the lowest bids in both auctions. With no commission, both auctioneers will allocate to this agent, but only one will succeed. The other auction will therefore have to be repeated. While this may seem like a minor effect, in networks that have mostly homogeneous nodes, simultaneous tasks, multiple auctioneers, and where the auction community covers a significant portion of the network, these conflicting allocations can occur frequently. When commission is not equal to 0%, however, many of these conflicting allocations will be avoided. For the auctioneers to still receive equal bids, the distance in hops from both auctioneers to the agent must be the same too.

A special case occurs when the system has no commission ($k = 0$) and the transmission cost is zero ($c_{\text{tx}} = 0$): the allocation changes to a round-robin pattern. With every allocation a node's energy would be reduced to below that of the other worker nodes, which results in subsequent tasks being allocated to other nodes with a greater remaining energy.

Negative commission, where relayed bid values are decreased with every hop, will amplify the distributive element of allocation, pushing task allocation to the outer limits of the auctioneers' zones. This could serve as a useful mechanism to facilitate long-distance distribution of tasks, but investigating it is beyond the scope of this work.

An optimal value of k could be determined for a particular network, but it is dependent on the topology, workload and energy regeneration characteristics of the system. Even then, the relative benefit of different commission values is small. The establishment of zones of allocation provides the largest component of the performance gain, and this behaviour depends more on the existence of commission than any particular value thereof. In my opinion, only stable topologies will benefit significantly from optimising k. In the majority of cases, a value between 5% and 20% should be sufficient, or at least provide a good starting point for further optimisation.

4.1.4 Discussion

The above examples demonstrate the allocation dynamics that result from the proposed task allocation strategy. Tasks are preferentially allocated close to the auctioneer, in a manner that distributes energy consumption to maximise network lifetime. This verifies the obvious primary requirement of the allocation mechanism, namely that it must allocate tasks effectively.

Implicit groupings were observed to arise due to the spatial proximity of specific agents, where repeated interactions can be observed. This partitioning into zones of influence occurs without global information or decision making; it is the emergent result of the interaction between agents. This could be seen as an emergent form of the transaction cost theories for the existence and structure of firms (Axtell, 1999). The sensitivity of the allocation to topology, workload and agent availability, means the mechanism autonomously adapts to changes.

4.2 Robustness

Tasks are allocated according to the available energy of a node and the node's distance from the auctioneer. With the basic dynamics of the allocation mechanism described above, we can now consider the more challenging scenario of node failures. The objective is once again to improve our understanding of dynamics that result from interactions between agents, while also confirming the allocation mechanism's potential for robustness. It should be noted, however, that this is but a single point in parameter space: not enough to contribute to an emergent taxonomy. I will therefore extend this investigation in the following chapter.

The ad hoc nature of running a new auction for every allocated task means the system will rapidly adapt to changes in topology, such as those caused by

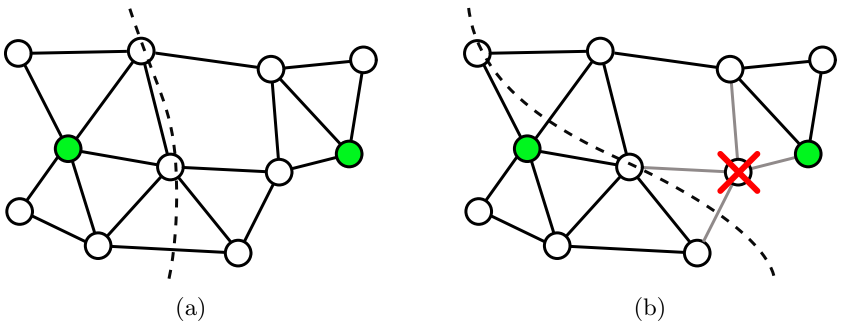


Figure 4.4: Task distribution shifts to reflect changes in the network. In (a) the dotted line shows the area in which the majority of tasks sold by each auctioneer (shaded node) find their allocation. The nodes intersected by the dotted line receive tasks from both auctioneers. If a node fails, we lose both its capacity to complete jobs and its routing functionality. The relative distances (measured in number of hops between nodes) and loading of nodes will change, resulting in a new distribution of labour in the network (b).

node failures. As an example, the linear networks of the previous section are expanded to a more realistic two-dimensional arrangement in Figure 4.4, which demonstrates how the zones of influence will shift in the event of a node failure.

To quantify the robustness of the proposed market-based allocation approach, I compare it to idealised control strategies. I am specifically interested in the effect of failures on the system level behaviour — I therefore use the number of tasks successfully allocated as a metric. Robustness can then be measured as the change in performance due to failures.[1]

4.2.1 Simulation setup

The market-based allocation scheme is used in a network with 100 nodes arranged in a 10 by 10 square lattice formation. Nine of the nodes act as sources of tasks; the remaining 91 are worker nodes. To simplify the description of time-dependent events, and relate the simulation to a space mission scenario, I divide simulation time into "days" of 100 time steps each. Tasks are introduced to the system at the beginning of each day at a rate of 9 tasks per day. Initially each auctioneer receives one task for allocation, but they may fail just as worker nodes can. In this case tasks continue to enter the system at a constant rate, with excess tasks distributed randomly across the pool of remaining auction-

[1]The work in this section formed part of my paper submission to ECAL 2009 (van der Horst *et al.*, 2009).

eers. Nodes have a uniformly distributed failure probability of 0.001 per day per node. Approximately half the nodes fail after 800 days.

Whereas previous discussions used tasks of constant size, here I vary their energy cost according to a Weibull distribution with shape parameter $\beta = 2$ and scale $\eta = 2$. The Weibull distribution is commonly used in simulation to represent task durations. The distribution captures the idea that most tasks will have a characteristic duration, with some outliers taking much longer, and the shortest possible duration being logically bounded by zero (Law and Kelton, 2000). Duration can be treated as being equivalent to size (i.e., energy cost) in this model.

The recharging of batteries from solar panels is implemented by increasing nodes' energy by 0.15 units per day, up to a maximum of 10 units per node. The recharging happens continuously over the course of the day. Transmission cost is set to 0.001 units per packet for negotiation packets (c_{tx}), and 0.1 units for task transferral (c_{tf}). Commission (k) is fixed at 20%. The atomic task structure used in Section 4.2.3 is used again:

$$A \rightarrow a$$

4.2.2 Candidate allocation schemes

Three alternative allocation strategies are used to compare robustness. I have opted for idealised versions of centralised control, as they can provide a common reference. Although the following systems could not be built the real world, they do provide useful measures for comparison with other systems.

The *ideal* case represents the best possible performance; as such it provides a theoretical upper-bound on allocation. This assumes an omniscient controller that has perfect knowledge of the network: both topology and the internal states of nodes. The controller can communicate cost-free with any node, without being constrained by network topology. Allocation is effectively treated as a bin packing problem: for every task, the controller finds the worker node with the most remaining energy and assigns the task to it. The controller is considered immune against failure. This model corresponds to the most abstract, mathematical view of the allocation problem.

In the *centralised* approach, we have an intelligent central controller that controls a network of simpler workers. The level of realism is increased by reintroducing the network topology and transmission costs. This allocation scheme approximates the performance of a central controller in a realistic network, and is subject to most of the same constraints as the market-based approach. The single controller node is positioned in the centre of the same lattice. The remaining 99 nodes are workers, as opposed to 91 workers for the market-based control case. The controller has perfect information about the energy levels of nodes in the network, as well as the topology of the network — we ignore the cost of maintaining this information. Tasks are again assigned as in the ideal case, with the additional constraint that to allocate a job to a node, a valid path must exist between the central controller and the selected worker node. A path is valid if all nodes along it are active and have enough energy for transmission. Just as in the market-based control case, nodes along the path have their energy decreased by 0.1 units when communicating the allocation of a task, however no negotiation packets are transmitted.

As the controller node in the centralised approach is a single point of failure, we assume that in a real mission scenario, it would incorporate redundancy to decrease its vulnerability. I therefore model this node as being immune to failure in the *centralised with immunity* (CI) case. All other variables are the same as used in the centralised approach.

4.2.3 Performance

The system was allowed to settle into steady state behaviour, as described in Section 4.1.1, before enabling the failure of nodes. The number of tasks successfully assigned was measured at intervals of 100 days and normalised with respect to the steady-state performance of the ideal allocator. This was repeated 80 times to obtain an average behaviour; the resulting performance is shown in Figure 4.5.

The ideal system can be seen to form an upper bound on the allocation success. It deteriorates over time as the number of failed nodes increases and the system's capacity to complete jobs decreases. The market-based approach displays lower initial performance: due to the energy cost of communication it accommodates only 76.9% of the tasks the ideal case does. The energy used on communication could not be utilised in executing tasks, resulting in the lower performance. Steady-state performance drops to 68.9% for both centralised control schemes, because of the larger portion of the energy budget spent on communication (the average path length when allocating tasks is greater than with distributed allocation). Progressive node failure decreases the total capacity of the network: allocation paths become longer and use more energy, and the network is fragmented when all routes to functioning nodes are cut. This is reflected in the steep slope of the centralised, CI and market-based control data. The sensitivity of the network to failure of the central controller is significant, as can be seen when comparing the centralised approach to CI.

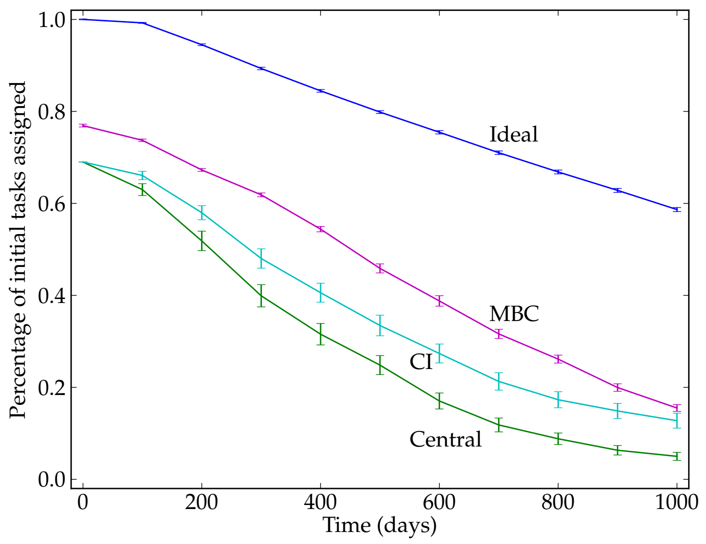


Figure 4.5: Performance and robustness of different allocation strategies. Network performance is measured by the total number of tasks allocated, normalised with respect to the steady-state performance of the ideal system. The amount of work stays constant, while nodes fail with a uniform probability. The ideal case provides an upper bound on the performance because it does not take communication cost or network topology into account. The market-based control allocation scheme (MBC) deteriorates faster than the ideal case, but performs more efficiently than the centralised and centralised with immunity (CI) approaches. The error bars indicate the standard error over 80 runs.

4.2.4 Robustness

We define robustness as the ability of the system to maintain steady-state performance despite node failures. To compare the robustness of the different systems, the results from Section 4.2.3 are normalised with regards to their respective steady-state values (Figure 4.6).

The ideal case again provides an upper bound. The centralised case deteriorates rapidly, largely due to the whole network collapsing if the controller node fails. The CI approach performs better, almost as well as the market-based approach. Both these approaches are subject to network fragmentation, but CI is definitely more sensitive. Remember that I am proposing the market-based approach as an allocation mechanism for systems where centralised control is not suitable. Seeing it outperform an allocator with global knowledge of the system is therefore a strong confirmation of the viability of the distributed approach.

To express these results in terms of satellite mission reliability, we define a

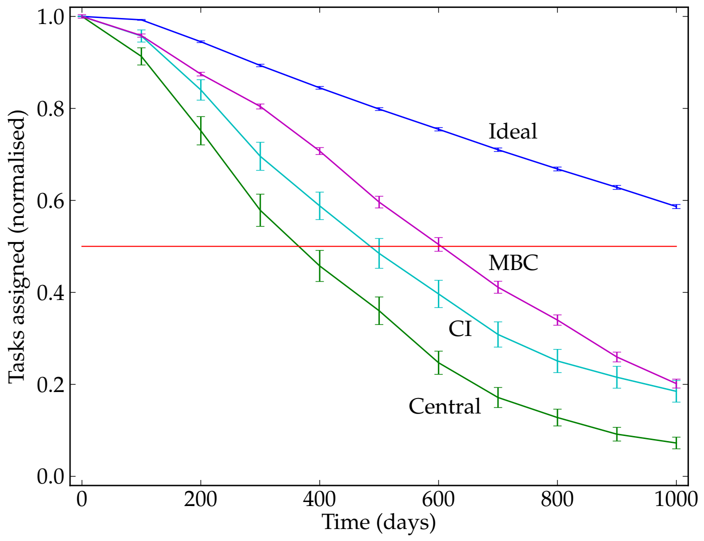


Figure 4.6: Robustness of different allocation strategies. In this experiment the ability of the system to maintain its performance despite node failures is measured. The performance data from Figure 4.5 is normalised with respect to the steady-state performance of the respective allocation strategies. The ideal case shows the theoretical maximum obtainable, if topology and transmission cost have no influence. Market-based allocation (MBC) shows a more gradual deterioration than either of the centralised approaches. The vulnerability of the network due to failure of the controller node is clearly visible when comparing the centralised case to CI case. The solid horizontal line indicates 50% of the initial throughput, which can serve as an operational threshold for satellite missions. The error bars indicate the standard error over 80 runs.

mission as operational while it delivers more than 50% of its initial throughput. This is analogous to a satellite system that was designed to perform a certain amount of work, and will remain useful until its capacity is decreased by half. The centralised system reaches the 50% limit after 360 simulation days, the immune-centralised at 525 days and the market-based approach at 605, making it the most reliable of the three. In spite of having fewer worker nodes, the performance of the market-based system is superior. This is not only related to efficiency, but also to robustness. In particular, this is a result of having multiple auctioneers which are able to adapt their allocation to changes in the network topology and node utilisation. These results are especially promising for distributed space applications. Launch mass will always be the dominant factor in total mission cost and, assuming a given launch mass and spacecraft of equal size, our results show that more work can be done more robustly using a market-based approach.

Additional experiments confirmed that the qualitative behaviour of the system is robust to variations in parameter values. The qualitative observations still hold, although some quantitative changes occur. For smaller networks, the centralised and market-based performance results converge, because the allocation distance is decreased. If the ratio of transmission cost to task size changes, the performance will increase (for smaller packets) or decrease (for larger packets) accordingly. These relationships are explored in more detail in the next chapter.

4.3 Discussion

This chapter presented the dynamics of the task allocation mechanism designed for the distributed satellite problem. The objectives were to demonstrate the validity of the market-based allocation mechanism in terms of effectiveness and robustness. In addition, this chapter serves as a foundation for the more complicated scenarios presented later in the thesis, by building an understanding of the allocation dynamics that result from the interactions between agents.

The discussion first focused on the distribution of tasks in a simple linear network. The task allocation mechanism balances the distributive force of energy-aware allocation against the localising effect of commission to allocate tasks locally, but not over-exploit the nearest nodes. As a result the allocation pattern is partitioned into zones around auctioneers, reflecting the available worker nodes, energy and task loads. The novel commission parameter plays a significant role in this by encouraging auctioneers to allocate tasks locally.

The second half of the chapter focused on the efficiency and robustness of the system, by measuring the behaviour of the market-based allocation mechanism against a number of idealised central allocation models. Node failure decreases the maximum number of tasks that can be allocated through decreased resource availability (fewer workers are available) and network fragmentation. The market-based approach was shown to perform well under these circumstances, slightly outperforming a centralised allocator with full knowledge of the network, and faring significantly better than a central controller that is subject to failure. The improvement in performance is a result of lower system-level communication costs when assigning jobs, as well as improved robustness due to the distributed and adaptive nature of the control system.

From a methodological point of view, it is important to note that the auctioneers do not maintain a model of the system to keep track of where tasks are allocated, or how nodes perform. Instead the bid values tell them enough about the part of the system they can influence to achieve a good allocation. The system itself is therefore used as a model, by accessing the states of the nodes when required. To take liberties with Alfred Korzybski's famous expression, by using auctions for allocation, our territory becomes the map.

When should system designers use the system as a memory? For it to be feasible the cost of accessing the system should be low when compared to the effort of building and maintaining a model. This implies a low communication cost, or a highly dynamic system, or a system that requires a very complex model. It also helps if an allocator only needs to take a part of the system into account when calculating an allocation, as it reduces the amount of information that needs to be transmitted. One drawback of this approach is a speed penalty: for spatially distributed system accessing a model is always faster than accessing the real system.

The approach is warranted in the case of distributed satellite systems, primarily due to the unpredictable dynamics of the system. I would therefore expect the approach to also be suitable to mobile robotics applications. Routing in computer networks usually relies on a hybrid model: a system model, in this case routing tables, is used by default, but if routing fails the physical system is queried to update the model. The model speeds up routing significantly, and the network is stable enough to make it useful. At the other end of the spectrum we find very stable systems that can best be managed using detailed models, such as physical geography or telephone numbers. These systems change very slowly, and searching on the actual system is prohibitively expensive. The use of models such as road maps and telephone directories is therefore well-suited to help us navigate these systems.

5 Scalability and robustness

In this chapter I explore the parameter space between the areas where we know centralised allocation should be used, and where we think distributed control in the form of my proposed market-based allocation approach is applicable. This represents an early branch in the design process described in Section 2.6.3: the selection of a specific control process has a significant impact on the final system.[1]

The objective is to find ways of determining where each approach is most suited to my generic model of task allocation, and demonstrate it specifically with the distributed spacecraft problem. I do not want to demonstrate the absolute superiority of one approach over the other — each has its own region of applicability which I would like to define by mapping the relevant part of parameter space. Although the number of nodes in the network is the most obvious parameter to explore, performance also depends on the communication cost and network volatility. This contributes to the exploration of the problem space: the work in this chapter relates problems that are controllable using centralised mechanisms to systems that are better suited to distributed control.

5.1 Motivation

Robustness and scalability are frequently used to motivate the usage of decentralised approaches to controlling multi-agent systems. In recent years, however, distributed control has become an end in itself: a significant portion of recent literature claims to be of merit primarily because it demonstrates the application of such a technique. A separate community supports traditional centralised approaches, where global knowledge of the system is assumed to be indispensable. This is usually motivated by claims of verifiability or optimality. The schism between these opposing viewpoints is exacerbated by enthusiasts of

[1]This chapter is based on work that was accepted to the Workshop on Self-Adaptive Networks at the IEEE International Conference on Self-Adaptive and Self-Organising Systems (van der Horst and Noble, 2010).

either approach constructing problems that suit their preferred control architectures.

Distributed algorithms are implicitly designed for infinitely large systems (e.g., Shehory and Kraus, 1998), but practical demonstrations of self-organising control methodologies frequently only employ a modest number of agents. For example Tripp and Palmer (2010) use up to eighteen satellites, while Krieger *et al.* (2000) look at foraging with up to twelve robots. The performance of these agents is extrapolated to make optimistic claims about large-scale behaviour. While this approach is valid in a purely mathematical sense, the practicalities of dealing with increased complexity are being neglected (Durfee, 2004).

At the same time there is an assumption that small systems will be adequately addressed using centralised control. But this raises an important question: when should a system be regarded as "small", and when is it "large"? In other words, when should we use centralised control, and when is distributed control better? We can easily categorise the extremes, but the central space is much less clear. This hazy middle-ground is also where a number of real- world systems can be found, including the problem of task allocation in distributed satellite systems.

As satellites have traditionally been monolithic entities, the centralised control mindset dominates the design of these systems. Subsystems are largely autonomous in function, but not in decision making: for example the attitude and orbit determination subsystem independently measures and computes the spacecraft's attitude and orbit parameters, but only when allowed to by the central computer. This centralised mindset can easily be transferred to a distributed satellite system: a central "mother ship" monitors the other modules and instructs them on what to do. While it offers attractive possibilities for verification and testing, we also know intuitively that this approach will struggle to scale to thousands of modules. However, for the first generation of distributed satellite systems, which is envisaged to be much smaller (ranging in size from tens to hundreds), centralised control presents a viable approach for these systems: the optimal task allocation methodology remains unclear.

In this chapter I therefore compare centralised and distributed task allocation with the goal of finding this transition zone and understanding how it is influenced by the scale of the system, the transmission cost, and robustness to node failures.

5.2 Centralised allocation

In this section I develop a centralised allocation mechanism to compare market-based allocation against. This centralised approach is designed to serve as a fair comparison: it is subject to the same transmission cost constraints as the market-based allocation version, with reasonable fault-tolerance mechanisms. A central controller uses global information to compute the best allocation. The cost involved in building and maintaining the model used to calculate allocations is of particular interest in this experiment. As stated previously, the objective is not to prove one approach superior to the other, but rather to find the regions of parameter space where either approach dominates. Note that this centralised allocation scheme was designed to be comparable to the market-based approach in all aspects. It is therefore significantly more detailed and realistic than the approaches used to test robustness in Section 4.2. The key distinction between the two implementations is that the previous one did not take the communication cost of maintaining a system model into account; instead it was assumed that this information was freely available to the manager node. As I will demonstrate in this experiment, this cost can be significant if a certain level of robustness is required in a system with failures.

The network consists of a manager that calculates allocations using a model of the network, and a number of worker agents that execute tasks on command from the manager. The workers have the same capabilities as for market-based allocation: each has a specific skill and limited energy, some of which will be spent on communication. The manager polls agents to keep track of the available energy of nodes, their respective skills, and the connections between them. These messages are flooded through the network at regular *status polling intervals* (τ_{status}); every worker responds with its status information. These packets set up routes between the manager and all workers, and provides the manager with the necessary information to update its network map, which includes the capacity of nodes. As the manager knows the effect of all communication and allocation in the network, it can accurately adjust its view of the network as it progressively allocates tasks. However, external influences on the system, such as node failures, cannot be predicted and require detection. Workers do not maintain a map of the network: instead they only need to know how to reach the manager, who will in turn provide them with routing instructions for a task when they need it.

In describing the allocation process, we will rely on four classes of agents, distinguished by the roles they fulfil. The *originating agent* is the one that detects a new task and initiates the allocation process; the *manager agent*

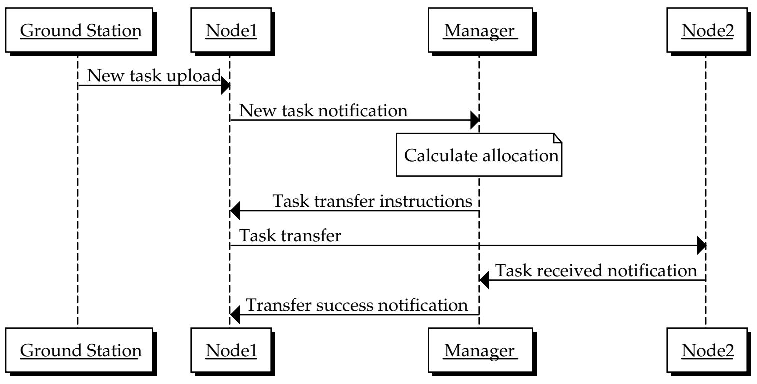


Figure 5.1: Message sequence diagram describing the task allocation flow for centralised control. Node 1 acts as originating node in this task allocation sequence, while node 2 is the recipient. All messages are addressed messages, no broadcast messages are sent. Messages can be relayed via intermediate agents (not shown). The manager is updated at every step of the allocation flow to maintain an accurate network model. Time increases from top to bottom.

maintains a model of the network topology and node states, which it uses to determine allocations; the *recipient agent* is the agent that the task is allocated to; while *intermediate agents* primarily relay messages between the other agents.

As with market-based allocation, tasks are initially injected into the system by the ground station. In the case of compound tasks, new task components appear in the system when the preceding task element has been completed. The relevant agent notifies the manager via intermediate agents upon detection of a new task component that needs allocation. The manager then uses its model of the network to determine the best allocation. The allocation instruction is then transmitted back to the originating agent. The agent proceeds to transfer the task to the receiving agent along a route specified by the manager. The receiving agent acknowledges receipt by transmitting a task-received message to the manager. The manager .now updates its model to reflect the energy used in transmission and allocation, before relaying the acknowledgement to the originating node, thereby closing the allocation loop. If the originating node does not receive the allocation acknowledgement message within a predetermined time, it will repeat the allocation process. A message sequence diagram describing this exchange is shown in Figure 5.1.

The manager agent relies on its model of the network, agent energies, and resources to determine the best allocation. Upon receipt of a task allocation request, the manager calculates, for every agent that possesses the necessary skills and sufficient energy, a figure of merit (B_{cc}):

$$B_{\text{cc}} = z\frac{e_{\max}}{e_{\text{rem}}}(1+k)^{d_{\text{tf}}} \tag{5.2.1}$$

where z is the task size, $e_{\max}$ the maximum energy of the node, e_{rem} the remaining energy, k is the commission and d_{tf} is the distance a potential task transferral will have to travel. The manager selects the agent with the lowest B_{cc} value for task allocation. As in the market-based control case, the allocation therefore favours agents that are relatively underutilised and are close to the originating agent. Note the similarity between this function and Equation 3.4.2, although Equation 5.2.1 does not estimate future outsourcing costs. The elements of compound tasks are allocated one at a time on an ad hoc basis, with the manager searching for the next suitable worker upon completion of the preceding task element.

The ability to allocate tasks successfully in a changing environment depends on the accuracy of the manager's network model and on the workers knowing the correct route to the manager. Both these factors are determined by the interval between status polling messages (τ_{status}) sent by the manager agent. These messages refresh workers' routes and update the manager's network model. With a large τ_{status} the manager is effectively implementing open-loop control, because it relies purely on its model of the system to allocate tasks. Smaller values of τ_{status} provide more feedback, which improves the accuracy of the model, but it also requires more energy for communication. τ_{status} therefore acts as a robustness parameter, which should be adjusted to fit the node failure rate.

The packet time-to-live distance (d_{ttl}) plays a similar role when using market-based allocation. Large d_{ttl} values increase the size of the auction community, which in turn improves the resilience of the allocation mechanism to failures. I will use these two parameters to draw robustness comparisons in Section 5.5.

5.3 Bounds on task allocation cost

Calculating upper and lower bounds for the task allocation costs of the different approaches allows us to make some predictions on where they might be relevant.

5.3.1 Market-based allocation

I start by developing an analytic description of the energy cost required to allocate a single task component when using the market-based allocation approach. The most expensive allocation case occurs when all the nodes in the auction community place bids and the winning node is the maximum distance away from the auctioneer. From the message sequence diagram (Figure 3.9), we see that this will result in n' auction announcement messages being broadcast. If all nodes in the auction community place bids, this will also result in n' bid messages, because each node only forwards the best bid it receives. To allocate to the most distant node requires d_{ttl} messages for each of the allocation, acknowledgement, task transferral and task acknowledgement steps. The worst-case total communication cost (c_{auc}), measured on system level, required to allocate a single task element is therefore:

$$c_{\text{auc}} = 2n'c_{\text{tx}} + 3d_{\text{ttl}}c_{\text{tx}} + d_{\text{ttl}}c_{\text{tf}} \tag{5.3.1}$$

As discussed in Chapter 3, the transferral of tasks between nodes usually requires more information, and therefore energy, than the transmission of negotiation packets (i.e., auction announcement, bid and acknowledgement packets). We can express the task transfer cost (c_{tf}) in terms of the transmission cost (c_{tx}), by scaling c_{tx} by a constant factor (α):

$$c_{\text{tf}} = \alpha c_{\text{tx}} \tag{5.3.2}$$

The auction cost can than therefore be expressed in terms of c_{tx}:

$$\begin{align} c_{\text{auc}} &= 2n'c_{\text{tx}} + 3d_{\text{ttl}}c_{\text{tx}} + \alpha d_{\text{ttl}}c_{\text{tx}} \tag{5.3.3} \\ &= c_{\text{tx}}\left[2n' + d_{\text{ttl}}(\alpha + 3)\right] \tag{5.3.4} \end{align}$$

In Equation 5.3.3, the first two terms represent the negotiation overhead, while the last term describes the task transferral cost.

If n_t tasks consisting of n_k components each are allocated, an upper bound on the total system cost over the duration of the experiment is given by:

$$\begin{align} c_{\text{mbc}_{\text{max}}} &= n_t n_k c_{\text{auc}} \tag{5.3.5} \\ &= n_t n_k c_{\text{tx}}\left[2n' + d_{\text{ttl}}(\alpha + 3)\right] \tag{5.3.6} \end{align}$$

We can see from this expression that the total cost of allocating tasks grows linearly with the number of tasks (n_t), and linearly with the number of agents in the auction community (n'). The auction community is in turn a function of the size of the auction community and the network topology — this will be explored in more detail in Chapter 6. For a constant d_{ttl} and a fixed topology, n' will also be constant: the system cost of task allocation is therefore independent from the number of nodes in the system. Communication cost c_{tx} has a linear relationship to $c_{\text{mbc}_{\text{max}}}$, making it a significant driver of overall system efficiency. Note that this analysis disregards the decrease in auction community size due to the edges of the network. If this is taken into account, Equation 5.3.6 will decrease further. This effect will be more significant for small networks, where d_{ttl} is comparable to the network radius.

We can similarly find the lower bound on allocation cost by considering the case where the auctioneer allocates the task to an immediate neighbour, who is also the only bidder. This decreases the number of bid messages to one, while the task transferral distance in Equation 5.3.6 is also reduced to 1. A lower bound on the system level allocation cost is therefore:

$$c_{\text{mbc}_{\text{min}}} = n_t n_k c_{\text{tx}}(n' + 4 + \alpha) \tag{5.3.7}$$

We can therefore conclude that this market-based task allocation mechanism will scale well, because the number of nodes in the network does not determine the system level allocation cost. Instead, the size of the auction community is the primary driver of the allocation cost, and this parameter is frequently controllable by the system designer.

5.3.2 Centralised allocation

The worst-case cost for centralised allocation consists of two parts: an allocation component, and the cost of updating the model of the system used for allocation.

The allocation component can be derived from Figure 5.1. The worst-case cost will occur in a linear network topology, where a task needs to be transferred from one end to another, and the auctioneer is located next to the receiving node (Figure 5.2). For a single allocation, this configuration maximises the task transferral distance (n), as well as the distance between the originating and the manager nodes (n-1). The allocation cost is therefore given by:

$$\begin{aligned} c_{\text{alloc}} &= c_{\text{tx}}(n-1) + c_{\text{tx}}(n-1) + c_{\text{tf}}n + c_{\text{tx}} + c_{\text{tx}}(n-1) & (5.3.8)\\ &= c_{\text{tx}}[3(n-1) + \alpha n + 1] & (5.3.9)\\ &= c_{\text{tx}}\left[n(\alpha+3) - 2\right] & (5.3.10) \end{aligned}$$

If n_t tasks of n_k components each are allocated, an upper bound on the total allocation cost over the course of an experiment is given by:

$$c_{\text{alloc}_{\text{total}}} = n_t n_k c_{\text{tx}}\left[n(\alpha+3) - 2\right] \tag{5.3.11}$$

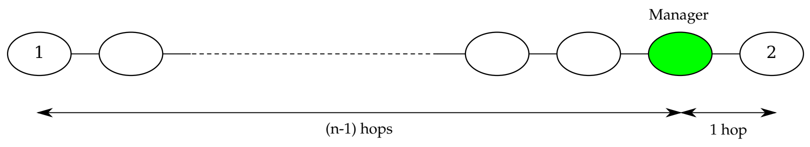


Figure 5.2: Worst-case topology when using centralised allocation. The shaded node acts a manager, node 1 is the originating node, and node 2 is the recipient. This topology maximises the task transferral distance (n) and the communication distance between the manager and the originating node.

The centralised manager also requires status updates to maintain its model of the system. A polling packet is flooded from the manager through the network to update routes between nodes and the manager — this requires up to n transmissions. Nodes reply to the manager with their skills and energy levels which provide the data for updating the network model. The worst-case update scenario again occurs in a linear topology, where the number of status packets is given by $\frac{n^2-n}{2}$. If we assume the update packets are the same size as negotiation packets, the energy cost of a single update (c_{status}) is:

$$c_{\text{status}} = nc_{\text{tx}} + c_{\text{tx}}\frac{n^2-n}{2} \tag{5.3.12}$$

$$= c_{\text{tx}}\left(n + \frac{n^2-n}{2}\right) \tag{5.3.13}$$

$$= \frac{1}{2}c_{\text{tx}}(n^2+n) \tag{5.3.14}$$

The fact that these status updates occurs at intervals of τ_{status}, allows us to calculate an upper bound on the total energy spent on updates of the course of an experiment with duration T:

$$\sum_{t=0}^{T} c_{\text{status}} = \frac{c_{\text{tx}}T(n^2+n)}{2\tau_{\text{status}}} \tag{5.3.15}$$

When combined with the total allocation cost (Equation 5.3.11), this gives the worst-case, system-level communication expenditure:

$$c_{\text{cc}} = c_{\text{alloc}_{\text{total}}} + \sum_{t=0}^{T} c_{\text{status}} \tag{5.3.16}$$

$$\Rightarrow c_{\text{cc}_{\text{max}}} = n_t n_k c_{\text{tx}}\left[n(\alpha+3)-2\right] + \frac{c_{\text{tx}}T(n^2+n)}{2\tau_{\text{status}}} \tag{5.3.17}$$

Note that $c_{\text{cc}_{\text{max}}}$ is dominated by the n^2 term that results from status updates. This means that in systems with large n a great deal of energy will be expended on maintaining the manager's system model. In the case of open-loop control ($\tau_{\text{status}} \rightarrow \infty$), the allocation cost is linear with respect to all the parameters in the allocation term, including the number of nodes in the system. As with market-based allocation, the system is also sensitive to the value of c_{tx} — in fact it will be even more sensitive due to the linear relationship with system size (n) rather than the with the constant n'. For a small number of nodes and a significant α, the first term of the expression will dominate, but as the network scales the maintenance cost grows quadratically.

The centralised approach is also significantly more sensitive to the value of α than market-based allocation. For centralised control, α is multiplied by n, while market-based control is limited by d_{ttl}. In other words, the task transferral distance is bounded by network size in the centralised case, instead of being capped by the packet time-to-live distance.

For the lower bound, we consider the case where the originating node, manager node and recipient node are all direct neighbours of each other. All communication happens between these nodes, which results in an allocation cost of:

$$\begin{aligned} c_{\text{alloc}} &= 4c_{\text{tx}} + c_{\text{tf}} && (5.3.18) \\ &= c_{\text{tx}}(\alpha + 4) && (5.3.19) \end{aligned}$$

The lowest status update cost results when the network is fully connected — no relaying of status packets is necessary. In this case, as status update will consist of n status request messages, and n responses, giving the status update cost as:

$$\sum_{t=0}^{T} c_{\text{status}} = \frac{2c_{\text{tx}}Tn}{\tau_{\text{status}}} \tag{5.3.20}$$

A lower bound on the system-level cost is therefore given by the sum of Equations 5.3.19 and 5.3.20:

$$c_{\text{cc}_{\text{min}}} = n_t n_k c_{\text{tx}}(\alpha + 4) + \frac{2c_{\text{tx}}Tn}{\tau_{\text{status}}} \tag{5.3.21}$$

This results in a much smaller allocation cost, but the maintenance cost remains dependent on the number of nodes in the system, regardless of where tasks are allocated. Note that this is a rather loose lower bound — in realistic scenarios, allocation and communication will never be confined to the space immediately surrounding the manager node. If the characteristics of a particular network topology are known, the status update cost can be estimated more accurately.

Table 5.1: Compound task structure used in experiments. The execution of task elements (left-hand column) results in another task element (right-hand column) that must be executed. Tasks are only considered complete if all the task elements are successfully executed. Nodes have skills that are randomly selected from the set a, b, c, d, e.

A	$\rightarrow$	aB
B	$\rightarrow$	bC
C	$\rightarrow$	cD
D	$\rightarrow$	dE
E	$\rightarrow$	e

5.4 Experimental setup

The above analysis provides us with some insight on system performance under ideal conditions, with a stable network topology. To measure the response of the respective allocation strategies to node failure, I use simulation.[2] The objective of the experiment is to determine the relationship between robustness and failure rate, and the system performance in terms of tasks allocated and energy efficiency. The intention is not to optimise for a specific scenario: it is my opinion that the resulting map of trade-offs parameters provides more useful information to system designers, and offers a more complete view of the problem space.

The test network initially consists of 225 agents in a 15 by 15 lattice topology. Agents can communicate directly only with their immediate neighbours to the north, south, east and west; communication to other agents must be relayed via these neighbours. Every agent has a skill or specialization randomly selected with a uniform distribution from the set of types $\{a, b, c, d, e\}$. Two compound tasks are injected into the network every 100 time steps, over a total duration of 2 000 time steps. These tasks consist of five sequential components, each of which requires one unit of energy to execute. The causal dependencies between the components are shown in Table 5.1.

Transmission cost was fixed at 0.01 units for a negotiation packet and 1 for a task transfer packet, i.e., the value of α was 100. Even though the basic model includes the possibility that agents will run out of energy, agents in the simulation runs described have effectively been supplied with infinite energy. This was done in order to separate the effects of node failure from node ex-

[2]For the simple network topology under discussion a analytical model of performance could be built, with a stochastic element to capture failures, but it would be specific to the network. The use of simulation allows multiple set-ups to be explored, and allows the dynamic behaviour of the system to be captured.

haustion, due to the large amount of energy used by the centralised allocation approach. Thus the focus is effectively on the allocation overhead, rather than the energy used in task execution.

5.4.1 Node failure

An agent is removed from the network when it fails: not only can it not perform any work, but its connections with other nodes are broken, disrupting routing and network topology. In this experiment an exponential distribution is used to model agent failures, where the shape parameter (λ) is the failure rate.

$$f(t,\lambda) = \begin{cases} \lambda e^{-\lambda t} & t \geq 0 \\ 0 & t < 0 \end{cases} \tag{5.4.1}$$

The failure rate (λ) is varied from 0 for no failures, to 10^{-4} for a mean of 192 observed failures over a full run of 2 000 time steps.

5.4.2 Robustness parameters

A higher failure rate requires greater robustness from the task allocation mechanism. With market-based allocation this is achieved by modifying the size of the auction community. I therefore range d_{ttl} from 3 to 8, which results in an auction community size ranging from 24 up to 144, although communities limited by the edge of the network are smaller. Central allocation is more successful if the model of the network is frequently updated when τ_{status} is small. I explore a range of values from 10 to 200, as well as the case where only an initial update occurs and tasks are allocated in an open-loop fashion. The experiment was repeated 50 times and the mean value of the results used.

5.5 Results

To compare the two allocation approaches, we look at the mean number of tasks successfully allocated, and compare it against the energy used to allocate the tasks.

5.5.1 Task allocation success

A contour plot of the mean number of tasks successfully completed is shown in Figure 5.3a for market-based allocation and Figure 5.3b for centralised control. This is shown for a range of failure rates and a significant variation in the robustness parameter (d_{ttl} for market-based allocation, τ_{status} for centralised control). Standard error values range from 0, when λ is small, to 0.951 for market-based allocation and 0.707 for centralised allocation. Note the similarity in the shape of the landscapes formed by the allocation schemes. Both exhibit a strong deterioration as λ increases, due to the higher number of failures. The decrease in performance is more rapid for centralised control. A top to bottom deterioration is also visible — in both graphs the greatest robustness is shown at the top. We observe that for small values of d_{ttl} market-based allocation fails to allocate successfully even when no failures occur. Centralised control is always successful in a network where no nodes fail.

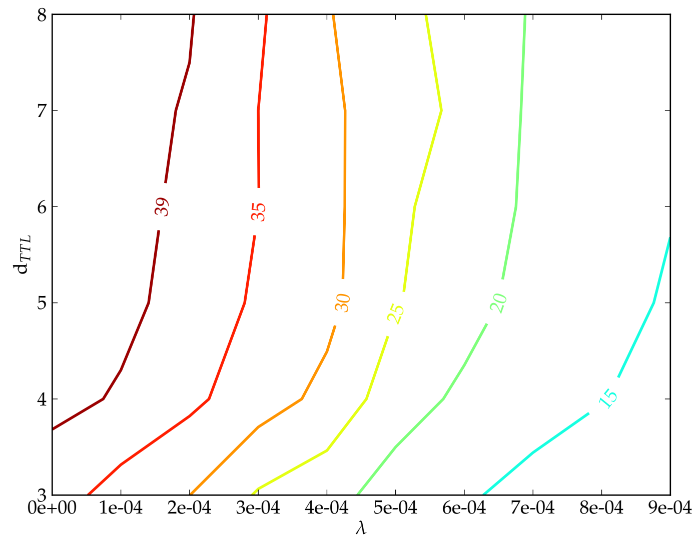
5.5.2 Energy consumption

In some networks the amount of energy used by the system (individually and cumulatively) has a tangible impact on the system's performance and overall cost. I have already described distributed satellite systems as having this characteristic, as larger batteries and solar panels will increase launch mass and mission cost. Although this experiment treats nodes as having infinite energy, we can use the measured energy consumption to predict where the respective allocation approaches will perform well.

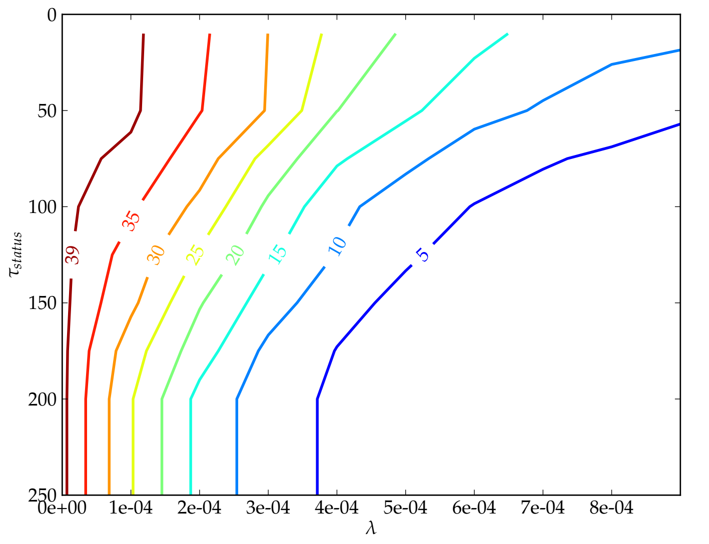
Figure 5.4a shows the total energy usage when using market-based allocation over the same values of node failure rate and time-to-live range as shown in Figure 5.3a. Note that for a given failure rate, increasing robustness by increasing d_{ttl} results in higher energy usage. Figure 5.4b shows the total energy usage for a centrally controlled system over the same values of node failure rate and status update interval as shown in Figure 5.3b. Standard error values range from 0.11 to 8.17 for market-based allocation, and 0.67 to 140.15 for centralised control, with the higher values found in the high energy consumption area. As expected, an increase in robustness comes at the expense of higher energy consumption. In both of these figures it may appear at first glance that a higher failure rate is good news. However, this decrease in energy usage only occurs because fewer tasks are successfully allocated. The main distinction between the two figures is that high levels of robustness in the centralised control case are associated with extreme levels of energy consumption, nearly an order of magnitude greater than for market-based control.

5.5.3 Conditions favouring either approach

When designing a task allocation system, some of the above parameters are imposed on the system designer, while others have to be selected. The cumulative

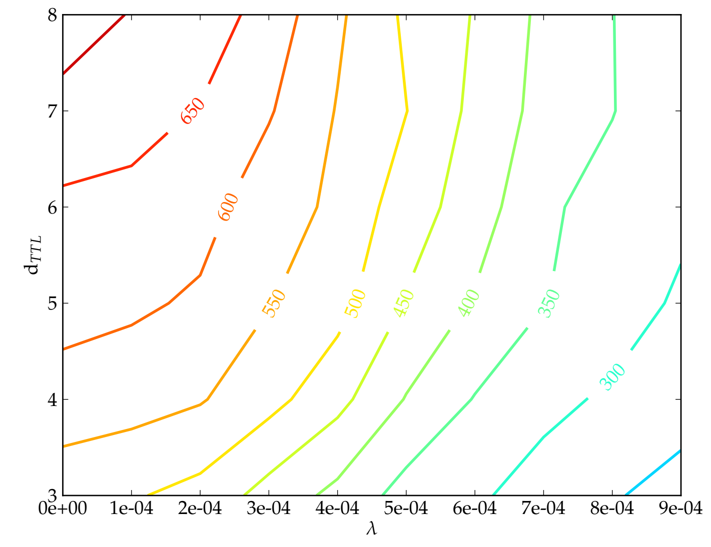


(a) Market-based control

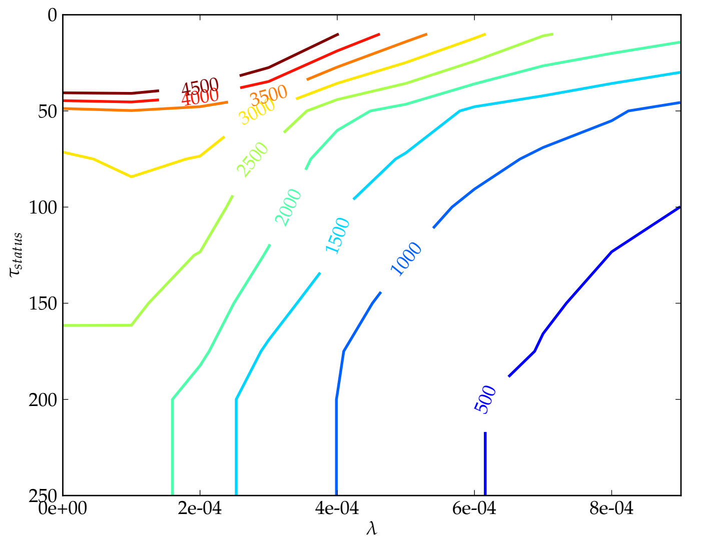


(b) Centralised control

Figure 5.3: Mean number of tasks successfully allocated over 50 runs using market-based and centralised control, plotted for a range of node failure rates (λ). The measurements are plotted against d_{ttl} for market-based control and τ_{status} for centralised allocation, i.e., their respective robustness parameters, with higher robustness at the top of the graph.



(a) Market-based control



(b) Centralised control

Figure 5.4: Mean energy used in allocating tasks over 50 runs using market-based control (Figure 5.3a) and centralised control (Figure 5.3b) for a range of node failure rates (λ). The measurements are plotted against d_{ttl} for market-based control and τ_{status} for centralised allocation, i.e., their respective robustness parameters. Note the large amount of energy required for frequent status updates.

effect of these parameters will determine whether a market-based or centralised approach is to be preferred.

For example, suppose the above system is implemented in an environment where the node failure rate is 1.5×10^{-4} and we require an average allocation success of 97.5% (39 out of 40 tasks). Referring to the task allocation graphs (Figures 5.3a and 5.3b), we find that a market-based allocation approach requires $d_{\text{ttl}} \geq 5$ while the centralised approach requires $\tau_{\text{status}} \leq 50$. On the energy consumption graphs (Figures 5.4a and 5.4b), these points show expected energy consumption of 610 units for market-based allocation and 1 500 for centralised allocation. In this particular case, market-based allocation is clearly the preferred approach.

However, this is not always so: we can use Equations 5.3.6 and 5.3.17 to estimate the network size for which centralised control and market-based control will be equivalent in performance. It follows that for smaller networks centralised control will be favoured. Note that this is an approximate threshold based on the expressions we derived for the bounds on allocation cost in Section 5.3. For the two-dimensional lattice topology in this experiment, the Manhattan geometry gives the number of nodes in the auction community as:

$$n' = \sum_{i=0}^{d_{\text{ttl}}} 4i \tag{5.5.1}$$

For a given time-to-live distance, we can express this sum as:

$$n' = 2\left(d_{\text{ttl}}^2 - d_{\text{ttl}}\right) \tag{5.5.2}$$

As a result, the upper bound on the total allocation cost when using market-based control (Equation 5.3.6) changes to:

$$\begin{aligned}
c_{\text{mbc}_{\max}} &= n_t n_k c_{\text{auc}} && (5.5.3)\\
&= n_t n_k c_{\text{tx}} \left[2(d_{\text{ttl}}^2 - d_{\text{ttl}}) + d_{\text{ttl}}(\alpha + 3)\right] && (5.5.4)\\
&= n_t n_k c_{\text{tx}} \left[2d_{\text{ttl}}^2 - 2d_{\text{ttl}} + \alpha d_{\text{ttl}} + 3d_{\text{ttl}})\right] && (5.5.5)\\
&= n_t n_k c_{\text{tx}} \left[2d_{\text{ttl}}^2 + d_{\text{ttl}}(\alpha + 1)\right] && (5.5.6)
\end{aligned}$$

while the lower bound can be derived from Equation 5.3.7:

$$\begin{aligned}
c_{\text{mbc}_{\min}} &= n_t n_k c_{\text{tx}} \left[2\left(d_{\text{ttl}}^2 - d_{\text{ttl}}\right) + 4 + \alpha\right] && (5.5.7)\\
&= n_t n_k c_{\text{tx}} \left[2d_{\text{ttl}}^2 - 2d_{\text{ttl}} + 4 + \alpha\right] && (5.5.8)
\end{aligned}$$

The upper bound on allocation cost in a system with centralised control can be similarly modified to incorporate the effect of the lattice topology. The maximum distance between the manager node and any worker node is $\sqrt{n}$, while the maximum transfer distance is $2\sqrt{n}$. This gives the upper bound on allocation cost as:

$$c_{\text{alloc}_{\text{total}}} = n_t n_k c_{\text{tx}} \sqrt{n}(2\alpha + 4) \tag{5.5.9}$$

The lower bound is still given by Equation 5.3.19:

$$\begin{aligned} c_{\text{alloc}_{\text{total}}} &= n_t n_k (4c_{\text{tx}} + c_{\text{tf}}) && (5.5.10) \\ &= n_t n_k c_{\text{tx}}(\alpha + 4) && (5.5.11) \end{aligned}$$

The total status update cost can be determined much more accurately than in Section 5.3.2, because the topology of the network is known, and does not change. For a square lattice, $(n-1)$ status request messages are flooded through the network. If all nodes send a response, all the messages need to be relayed back to the manager. For a two-dimensional lattice, the total number of messages is given by the sum of the Manhattan-distances between the manager and every worker node in the network. In the case of a square lattice with the manager located at the centre, the number of packets transmitted for a status update can be expressed as:

$$n_{\text{status}} = \begin{cases} (n-1) + \sum_{i=0}^{\sqrt{n}} 4i^2 - \sum_{i=0}^{\frac{\sqrt{n}-1}{2}} 2(2i+1)\left(\sqrt{n} + 2i + 1\right) & \text{if } n \text{ is odd} \\ (n-1) + \sum_{i=0}^{\sqrt{n}+1} 4i^2 - \sum_{i=0}^{\frac{\sqrt{n}}{2}} 2(2i+1)\left(\sqrt{n} + 2i + 2\right) \\ \quad - \sum_{i=\frac{\sqrt{n}}{2}}^{\sqrt{n}-1} (4i+4) - 4\sqrt{n} & \text{if } n \text{ is even} \end{cases} \tag{5.5.12}$$

An upper bound on the energy overhead when using centralised control is therefore given by:

$$c_{\text{cc}_{\max}} = n_t n_k c_{\text{tx}} \sqrt{n}(2\alpha + 4) + \frac{c_{\text{tx}} n_{\text{status}} T}{2\tau_{\text{status}}} \tag{5.5.13}$$

while the lower bound becomes:

$$c_{\text{cc}_{\min}} = n_t n_k c_{\text{tx}}(\alpha + 4) + \frac{c_{\text{tx}} n_{\text{status}} T}{2\tau_{\text{status}}} \tag{5.5.14}$$

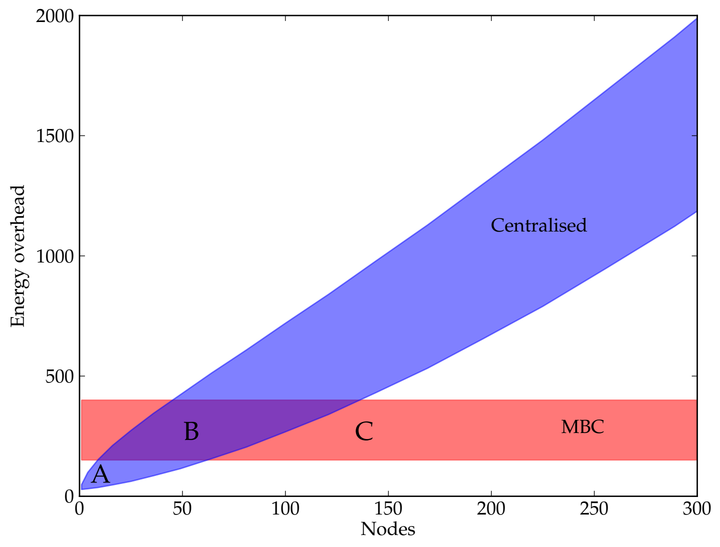


Figure 5.5: The expected energy overhead for market-based (red) and centralised allocation (blue) as a function of network size. The shaded areas indicate the difference between the upper and lower bounds. In large networks (the area marked C), market-based allocation will be more energy efficient than a centralised approach. For small networks (A) nodes, centralised allocation will probably be preferred. The area where the ranges overlap (B) is more ambiguous, and simulation will be required to determine the best strategy.

To visualise the effect of network size on the performance of the different allocation mechanisms, I plotted the areas between the upper and lower bounds for an example configuration in Figure 5.5. In the market-based allocation case d_{ttl} is set to 5, while the centralised approach uses a status update interval of 50. In both cases α is 10 — this relatively small value puts the predicted energy overhead in a relatively narrow band. All other parameters are kept as in the experiment above. The intersection of the bounds for the different approaches show where each would be preferred.

The scalability of the market-based approach is clearly demonstrated in the figure: the energy overhead remains constant for all network sizes.[3] In contrast, the energy required by the centralised approach grows at approximately $n^{\frac{3}{2}}$, primarily due to the increase in the number of status update packets.

For large networks, in the area marked C in Figure 5.5, the market-based approach is clearly preferred. In contrast, small networks (A) are more suited

[3]The bound calculations do not take the effects of network edges into account; we can therefore expect the overhead for the market-based approach to decrease in small networks.

to centralised control. The area where the two approaches overlap (B) is ambiguous as to which one would be preferred. In this area simulation is required to determine the best task allocation mechanism. No special meaning should be attached to these numbers: they are the result of the parameters of the scenario and as the parameters change, the intersection will also shift.

An increase in node failures will increase the required energy for both allocation approaches. However, as the previous section indicated, the centralised approach is much more sensitive and therefore requires more energy. As a result, the intersection area in Figure 5.5 will move to the left. A very stable network topology will similarly shift the point of ambivalence to the right, making centralised control attractive on larger systems.

The effect of the topology is also demonstrated in this example: the $\sqrt{n}$ and d_{ttl}^2 terms are determined by the lattice topology. For alternative topologies, e.g., scale-free networks, these terms would be different, again shifting the point of ambivalence. I will investigate the costs associated with different topologies in more depth in Chapter 6.

For both allocation approaches, total energy usage scales linearly with communication cost (c_{tx}). This means that there will be no point of ambivalence with respect to this dimension, except when $c_{\text{tx}} = 0$. However, low c_{tx} values will make the energy difference between the two approaches less significant, potentially allowing other parameters to dictate the best approach.

These calculations only apply to a specific failure rate ($\lambda = 1.5 \times 10^{-4}$). For systems with a different λ, the required values for d_{ttl} and τ_{status} will follow the contours in Figures 5.3a and 5.3b, which will in turn shift the curves in Figure 5.5 up for increased communication, and down for decreased communication. A lower failure rate will increase parameter range where centralised control would be preferred, while a more dynamic network will tend to favour market-based allocation.

5.6 Discussion

Centralised control is traditionally used for controlling small systems, while self-organising approaches, such as market-based control, have been proposed for large systems. In this chapter I explored some of the parameters that determine whether we should regard a system as “small”, or suited to central control; or “large”, where distributed approaches will be better. The results confirm that both distributed market-based control and centralised control have a place in the tool kit of a system designer.

In terms of distributed satellite systems, this chapter demonstrated the scalability and robustness of the market-based task allocation mechanism. In situations where the network is volatile, a larger auction community is required to ensure successful allocation. The system-level allocation cost is largely determined by the packet time-to-live value. From a mission analysis perspective, this implies that money can either be spent to decrease the failure rate of individual satellites, thus decreasing the size of the required auction community; or on larger power systems to accommodate the increased communication needed for higher robustness on the level of the allocation mechanism.

An analysis of allocation costs was used along with simulation to map the trade-off between successful task allocation and energy consumption for a range of failure rates. This allows for unbiased selection of the most efficient allocation approach. In addition, I demonstrated a method for estimating the boundaries in parameter space that divide regions where one architecture or the other is the most attractive.

Which parameters turned out to be the most important in determining the appropriateness of each approach? The size of the network is a critical factor, because the allocation costs of centrally controlled systems scale at a rate of up to n^2. In contrast, the market-based allocation approach is independent from the number of nodes in the network, as it is primarily determined by the size of the auction community. In this example system, I found that 10 to 150 agents (zone B in Figure 5.5) was the zone of ambivalence between market-based and centralised approaches. However, there is no suggestion that these figures are any sort of magic number dividing small from large systems. The point is only to demonstrate that it is possible to find this threshold for any given problem.

Network volatility, in the form of node failures, is also an important parameter, because the robustness measures required to deal with high volatility consume a significant amount of energy. This is especially true for centrally controlled systems. On the other hand, if the system is very stable, e.g., agents are extremely reliable, robustness measures are not required which changes the point of ambivalence dramatically: centrally controlled systems become much more competitive. This is not only applicable to agent failures, but also to changes in topology, or nodes changing their capabilities in a manner that is unpredictable to the manager node. For market-based control, the packet time-to-live determines the size of the auction community, and as a result the system's robustness to failure.

Whether communication cost should be regarded as significant depends on the relative size of tasks and communication costs. If the total communication cost is orders of magnitude less than task sizes, or communicating does not directly impact the ability of an agent to do work, the choice of allocation system will be determined by other factors, such as implementation and verification effort. When communication comes at some cost we will prefer one approach to the other, but increases in communication cost will never lead to a switch in the preferred option.

What does this tell us about related systems? It is clear that relatively small and stable systems are best controlled using a centralised approach. Small groups of mobile robots and smart electricity meters, where the communication topology doesn't change dramatically, fall into this category. Although other design constraints can influence the point of ambivalence between approaches, I would treat claims about the importance of distributed control for these applications with caution. On the other hand, wireless sensor networks, for example, are clearly better addressed using distributed control, due to the scale and volatility of the system.

6

Topology and communication cost

Communication cost is a persistent theme in this thesis. I use it to distinguish distributed satellite systems from other networked computing systems, and to relate them to wireless sensor networks. The cost of communication impacts on the management of the system by making information expensive. Tasks are preferentially allocated to local nodes, to minimise the cost associated with multi-hop transfers. The propagation characteristics of radio waves mean that multi-hop routing is more energy efficient than direct, long-distance communication; therefore I focus on communication networks that consist mostly of local links. Communication cost forces us to take topology into account, but at the same time the network topology also determines the impact of communication cost on the system's performance.

But what does this mean for a system designer? How connected should a network be? When dealing with physical systems, how should nodes be distributed? And how does the task allocation mechanism deal with a changing topology? In cases where the topology is controllable, the design decisions revolve around choosing a suitable topology for the system at hand. If the topology is not under control it acts as a design constraint; the system designer then faces the question of what the best management approach for a given topology will be.

To be able to judge when a task allocation mechanism suits a particular network topology, we need a better understanding of the interplay between communication cost and topology. At one extreme we have fully-connected networks, where information can be exchanged directly between any two nodes without significant impact on the whole system. Networks with a large diameter, where many hops are required to traverse from one end to the other, are potentially much more sensitive to communication cost. Note also that in systems where communication is effectively free, e.g., the internet, the underlying topology ceases to matter and all nodes appear equally accessible. In systems with expensive communication, such as road transport networks, the topology becomes the dominant concern.

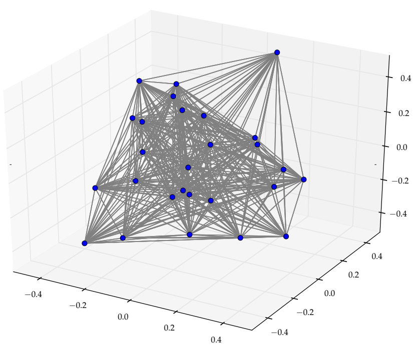
To effectively answer these questions I investigate the relationship between communication cost and topology in this chapter, with the focus on systems managed using the task allocation mechanism described in Chapter 3. In the case of distributed satellite systems, managing the system is challenging because of the constant change in topology: not only are links continuously made and broken, but the characteristics of the topology can vary greatly over the course of an orbit. In the first section I measure the performance of different topologies to identify the characteristics that contribute to system-level cost. The following section investigates the trade-off in communication power against network connectivity to determine how connected a network should be. Finally, I focus more closely on distributed satellite systems: how can we accurately describe the dynamic topology of such a system, and how does the constantly changing topology impact the market-based task allocation mechanism developed in the previous chapters.

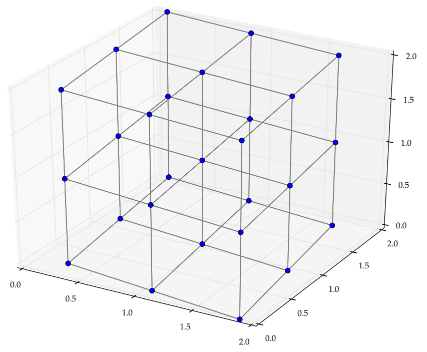
6.1 The cost of different topologies

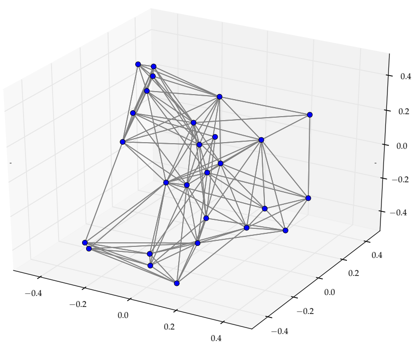
The topology of a network determines the number of packets transmitted during allocation. For different topologies, the size of the auction community and average task transferral distance vary. To better understand what makes some topologies more attractive than others, I compare the allocation cost of four different topologies and analyse their differences in performance. I specifically focus on networks that are embedded in space, and where communication links depend on this space, as this corresponds closely with the wireless communication model used by the distributed satellite system. The objective of this section is to develop an understanding of why different topologies result in different system-level allocation costs. This is used to explain the observed behaviour in subsequent sections.

6.1.1 Candidate topologies

I will consider four topologies: a fully-connected network, a cubic lattice, and two random geometric graphs, one in a cubic space and the other in an elongated rectangular prism. The first two topologies are very regular, resulting in allocation patterns that are easy to visualise and verify. The choice of networks is determined by the fact that nodes are embedded in a three-dimensional space, and that their connections are a function of this space, as will be discussed in more detail in Section 6.2. This constraint render many other topologies ir-


(a) Fully-connected


(b) 3D lattice


(c) Cubic random geometric graph

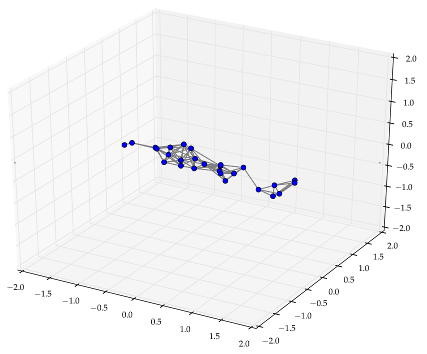

(d) Elongated random geometric graph

Figure 6.1: Different static network topologies used. For clarity networks consisting of only 27 nodes are shown.

relevant: the networks that frequently receive attention in the graph theory community, such as random graphs, scale-free and small-world networks, are often impossible to realize in a pure three-dimensional space.

One reference case is provided by a fully-connected network. In this topology all nodes are directly connected to all other nodes, i.e., the network diameter is equal to one. For a network with n nodes, this results in $\frac{n(n-1)}{2}$ links, while all nodes have a degree of $n-1$. The system-level allocation cost is therefore constant: all negotiation packets reach their destination in one hop; the task transferral similarly happens directly between the auctioneer and successful bidder. An example of such a network consisting of 27 nodes is shown in Figure 6.1a.

In a physical system, this represents a scenario where all nodes are within direct communication range of each other. For nodes using broadcast communication, such as the radios used in a distributed satellite system, the one-to-many nature of communication adds no extra cost to the transmitter. Communicating with one node is therefore just as expensive as communicating with everyone. However, the increase in traffic volume can require significant processing on the part of the receivers, and can cause congestion due to the shared bandwidth.

The second network under consideration is the three-dimensional analogue of the lattice formation used in previous experiments. This cubic lattice structure approximates a situation where only local communication is possible. The lattice structure results in a network diameter of $\sqrt{3}\sqrt[3]{n}$, while the degree of most nodes is six, with only node degrees on the edges of the lattice being smaller. The formation has many redundant routes, making it relatively robust to topological disruption. Figure 6.1b shows the topology.

The above networks present two cases at opposite ends of the parameter space: one is densely connected, while the other is much more distributed but still ordered enough for the dynamics to be understandable. A better approximation of the topology of distributed satellite systems is given by a random geometric graph (RGG, Dall and Christensen, 2002). The network is constructed by placing a number of nodes at random locations in three-dimensional space; two nodes are connected if the distance between them is not greater than some threshold communication distance R_{th}. Connections in this spatially distributed network are therefore deterministic and symmetric.

The network characteristics are determined by the node distribution and communication range. For a large R_{th}, the network will resemble the fully-connected network above, while the case where nodes are barely connected is much more similar to the cubic lattice. Decreasing R_{th} further will result in a sparse network, with a low number of connections and a high sensitivity to topological change. Random geometric graphs can therefore be seen to traverse the space between the fully-connected and three-dimensional lattice networks. This topology is of course not restricted to distributed satellite systems — it can also be applied to other distributed systems where the ability to communicate is determined primarily by the spatial distance between nodes.

I consider two instances of random geometric networks: in the one, nodes are distributed with a uniform probability in a cubic space (Figure 6.1c), while in the other nodes are contained in an elongated rectangular prism, as shown in Figure 6.1d. The first network is relatively well connected, with a large degree and small diameter. In the elongated case the diameter increases significantly, while the number of redundant paths in the network decreases. As I will show in section 6.3, these networks reflect actual topologies encountered in distributed satellite systems.

6.1.2 Performance comparison

The efficiencies of the different topologies are compared by simulating task allocation. A fixed number of tasks are allocated and the energy used is measured. The allocation overhead, i.e., the energy that was spent on message-passing instead of task execution, is used to compare the different topologies.

For every run, a network consisting of 125 nodes was used to allocate 500 tasks using the market-based mechanism described in previous chapters. Each task consists of 5 sequential components, where every component requires 1 unit of energy. Node skills are chosen uniformly at random from the 5 possible task types at the start of a run.

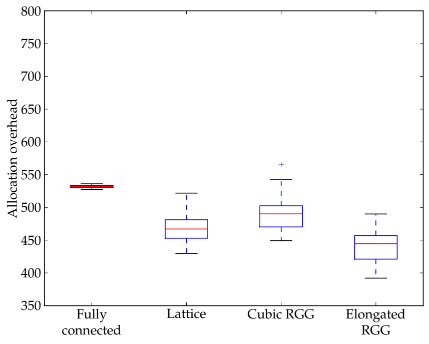
The cost of transmitting a packet is 0.001 units of energy, while transferral of a task requires 0.1 units. If a node relays a bid from a neighbour, it adds 10% commission to the bid value. Node energy regenerates at a rate of 0.006 units per time step, with every node capable of storing up to 10 units of energy. These values were selected to represent a system where task execution requires a significant amount of energy, task transferral is an order of magnitude cheaper, and sending a negotiation packet is cheaper still. The energy regeneration rate is enough to make node exhaustion unlikely, which allows us to focus solely on topological effects.

For the cubic random geometric graph, nodes are positioned uniformly at random in a cube with side lengths 1. The elongated random geometric graph is generated by distributing the nodes randomly throughout a prism with dimensions 0.5:0.5:4 (compared to the unit cube). The volume is the same in both cases, giving a uniform node density. A communication range (R_{th}) of 0.3 units determines which nodes are connected: this is large enough to ensure the network usually consists of a single component. In the rare cases where a single component did not result, a different network was generated.

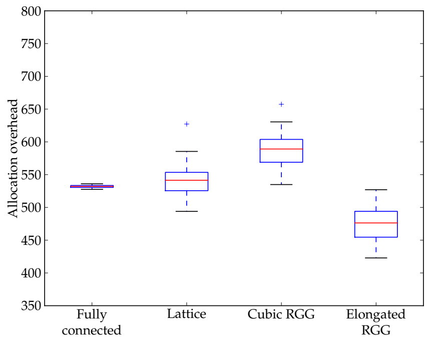
An experiment consists of 50 runs, where each run has a different set of node skills and, for the random topologies, node positions. The experiment was repeated using transmission time-to-live (d_{ttl}) values ranging from 3 to 6 hops.

6.1.3 Discussion

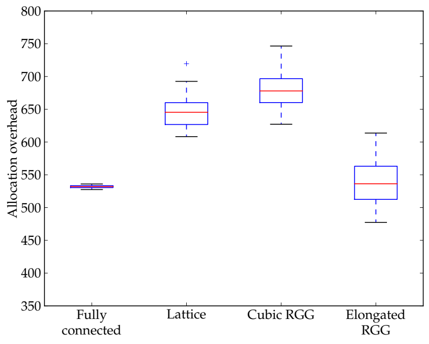
The energy overheads for task allocation in the different topologies are shown in the box and whiskers plots in Figure 6.2. The boxes extend from the lower to the upper quartile of the data, with the red line indicating the median. The whiskers show the extent of the data, with outliers marked separately. The average allocation overhead is around 550 units. Note the compounding



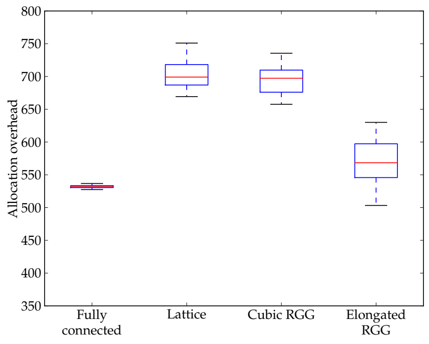
(a) $d_{\mathrm{ttl}} = 3$



(b) $d_{\mathrm{ttl}} = 4$



(c) $d_{\mathrm{ttl}} = 5$



(d) $d_{\mathrm{ttl}} = 6$

Figure 6.2: Energy used for different topologies over 50 runs. The box extends from the lower to the upper quartile of the data, with the red line indicating the median. The whiskers show the extent of the data, with outliers marked separately. Increasing d_{ttl} increases the size of the auction community, which causes the allocation overhead to go up. The cost of the fully-connected network stays constant, because the entire auction community is directly connected to the auctioneer.

effect of repeated communication: an apparently small c_{tx} value adds up to a significant amount when compared to the energy used in task execution.

The fully-connected network shows a narrow band of energy usage which does not change with d_{ttl}. This distribution reflects the fact that the size of the auction community remains the same across runs. The small variation that can be observed in the allocation cost is due to the different compositions of the networks used for test runs, which in turn changes the number of bid messages.

If we consider the other topologies, the allocation cost for the elongated RGG is the lowest, while the cubic RGG is the most expensive. The lattice topology performs somewhere in-between. For $d_{\mathrm{ttl}} = 3$, allocation in all three networks is less expensive than in the fully-connected case, but increasing d_{ttl}

raises the cost, especially for the lattice and cubic RGG. By increasing d_{ttl}, the size of the auction community is increased for the lattice and RGGs. This in turn causes an increase in the number of negotiation messages, which makes up the bulk of the increase in allocation cost. The task transfer cost similarly shows an increase, but the vast majority of allocations are still close to the auctioneer. This is characteristic of the relatively small set of node types in this network — a node with the required skill can usually be found in the vicinity of the auctioneer.

The elongated RGG exhibits a small increase in cost for increased d_{ttl}. This is the result of the large diameter and small radius of this topology: an increase in d_{ttl} adds a relatively small number of nodes to the size of the auction community when compared to the other topologies. The high cost of well-connected topologies should come as no surprise — the task allocation mechanism was after all designed for sparse networks. The primary culprits for the sensitivity to auction community size are the flooding of auction announcement messages and the large number of bids that result.

In conclusion, these results demonstrate how different parameters determine the allocation overhead. For random geometric graphs, the connection range (R_{th}) and node density determine the connectivity. The network connectivity, combined with the packet time-to-live (d_{ttl}), in turn governs the size of the auction community, which ultimately decides the allocation overhead. By taking these parameters into account, a system designer can influence the allocation overhead. While a network needs to be sufficiently connected to allow successful task allocation and to be robust to changes, excessive connectivity will use a significant portion of the available energy. The high allocation overhead in well-connected networks suggests that a different approach may result in better performance; however, the market-based allocation mechanism is well adapted to sparse topologies.

6.2 Communication range and connectivity

When dealing with spatial networks, communication range affects the network in three ways. Firstly, the communication range needs to be large enough to ensure a reliable network is formed, on which tasks can be effectively allocated. Secondly, the communication range is determined by the transmission power: increasing the range in radio networks requires an increase in the transmission power, which in turn increases the energy cost of communication. Finally, as shown in the previous section, networks that are well-connected generate a large number of negotiation packets which affects the system's performance negatively. Previous work on optimising communication distance between satellites (Wu *et al.*, 2008) primarily focused on balancing transmission delays induced by packet relay against the communication cost of direct communication. In this section, the focus is instead on the topological implications of optimising communication distance.[1] What is an appropriate communication range, given that range is determined by the transmission power and that the network requires a sufficient degree of connectivity to function?

I use a simplified deterministic radio propagation model to decide when two nodes can communicate. In this model transmission is isotropic and symmetric, i.e., insensitive to direction and the same for all nodes. Communication can therefore be visualised as taking place in a sphere with radius R_{th} around a transmitting node. If the transmitting node transmits with power P_{tx}, the power received by the second node (P_{rx}) is dependent on the distance between the nodes (r):

$$P_{\mathrm{rx}} = \frac{P_{\mathrm{tx}}}{r^2} \tag{6.2.1}$$

If we assume that the receiving node has a threshold power level (P_{th}) below which it cannot reliably communicate, the communication range of a node is given by:

$$R_{\mathrm{th}} = \sqrt{\frac{P_{\mathrm{tx}}}{P_{\mathrm{th}}}} \tag{6.2.2}$$

Furthermore, the energy expended in communication is proportional to the transmission power; and as communication cost (c_{tx}) describes the energy consumed to transmit a packet, we find:

$$R_{\mathrm{th}} \propto \sqrt{c_{\mathrm{tx}}} \tag{6.2.3}$$

The resulting network topology is a random geometric graph, with the connectivity determined by the communication power. This relationship between communication cost and distance means that the energy impact of long distance communication between nodes can become significant, while shorter distance, multi-hop routing provides a more efficient option. However, if the communication distance is too small, the network will consist of multiple components which will affect performance adversely. I use simulation to find the balance between these two extremes. Although analytic methods could in theory be used to derive stochastic models of connectivity, when additional constraints such as compound tasks and orbital dynamics are considered, mathematical

[1] For a detailed discussion on communication architectures and link design in satellite systems, refer to Chapter 13, Larson and Wertz (1999).

Table 6.1: Compound task structure used in topology experiments. The execution of task elements (left-hand column) results in another task element (right-hand column) that must be executed. All components must be executed before the task is complete.

$A \rightarrow$	aB
$B \rightarrow$	bC
$C \rightarrow$	cD
$D \rightarrow$	dE
$E \rightarrow$	e

methods quickly become unwieldy and often seemingly intractable. Simulation offers an attractive way to capture the messy nature of real-world systems.

6.2.1 Experimental setup

In this experiment I map the effect of communication range by measuring the allocation cost and the number of tasks completed for a range of R_{th} values. An increase in R_{th} causes a corresponding increase in c_{tx}, as described above.

A three-dimensional cubic space is used for this experiment; all dimensions are scaled to fit in a unit cube. 125 nodes are placed uniformly at random in this space, and connected based on their spatial proximity. A range of communication distances (R_{th}) from 0 to 1.7 units is used: at the lower end, no nodes will be connected, while the upper end results in a fully-connected network. The energy capacity of all nodes is 10 units. Batches of 10 tasks are uploaded every 100 time steps, with 50 batches (500 tasks or 2 500 task components) submitted over the course of one simulation run. System performance is measured by the number of tasks completed. The tasks consist of five components, executing in a linear fashion, as defined in Table 6.1. Node skills are selected from this set of possible task elements with a uniform distribution; we therefore have approximately 25 of each of the five types of nodes.

Transmission cost is calculated by using the communication range according to the inverse square relationship described above. P_{th} is set to 10^{-4}, and R_{th} ranges from 0 to 1.7 units (the length of the diagonal of a unit cube), so the transmission power can be calculated as:

$$P_{\text{tx}} = P_{\text{th}} \times R_{\text{th}}^2 \tag{6.2.4}$$

To relate P_{tx} to the transmission cost c_{tx}, we scale it to so that the maximum task transfer cost is equal to the cost of executing a single task component (1 unit), thus making these costs comparable. This gives:

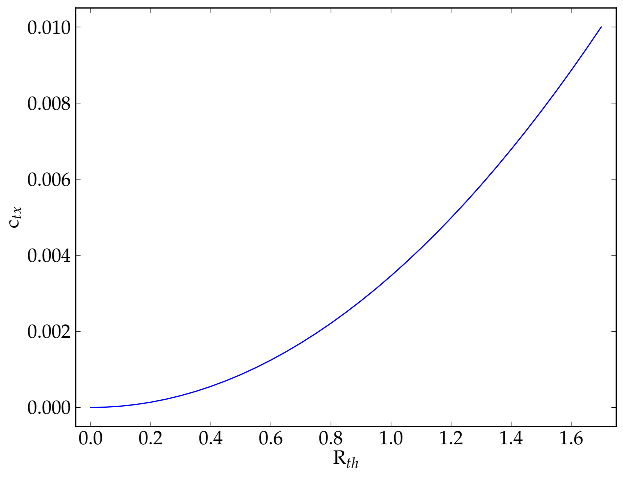


Figure 6.3: Communication cost (c_{tx}) as a function of communication distance (R_{th}). c_{tx} is proportional to the square of R_{th}^2, due to the underlying radio propagation characteristics.

$$1 = c_{\text{tx}} \times \alpha \tag{6.2.5}$$

$$\Rightarrow c_{\text{tx}} = \frac{P_{\text{tx}}}{\alpha \times P_{\text{tx}_{\text{max}}}} \tag{6.2.6}$$

The transfer packet size (α) is set to 100 and $P_{\text{tx}_{\text{max}}}$ to 2.89 to ensure coverage of the entire space. The value of c_{tx} therefore ranges from 0 to 0.01, proportional to the square of the communication distance, as shown in Figure 6.3.

Nodes' energy levels increase by 0.006 units per time step. The maximum amount of energy the system can therefore gain per time step is 0.75 units. In comparison, the submitted tasks on average use $\frac{5\times10}{100} = 0.5$ units of energy per time step. We therefore expect the system to successfully allocate tasks in cases with low communication cost (if the topology allows it), but higher communication costs will exhaust nodes and cause task allocation to fail.

As the node positions are random, the resulting network topology and system-level communication cost will vary between networks. Fifty runs were used to generate the results presented below.

6.2.2 Experimental results

I measure global system performance by the number of tasks completed. The box plot in Figure 6.4 shows how the communication range influences the

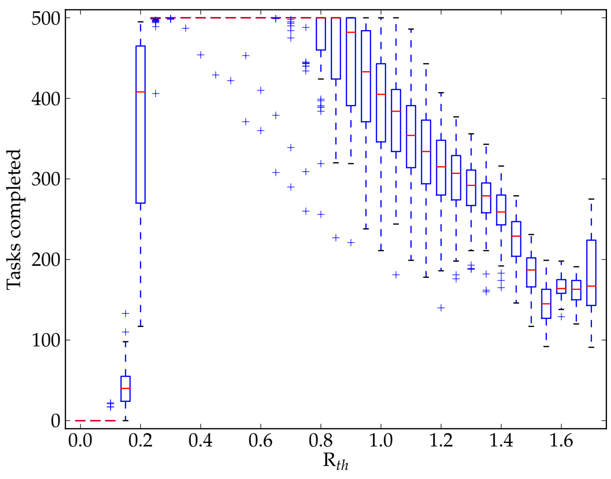


Figure 6.4: Box plot of the number of tasks allocated for a range of communication distances (R_{th}) over 50 cubic random geometric graphs. The boxes show the range of the lower to the upper quartile of the data, with the red line indicating the median. The whiskers show the spread of the data within 1.5 quartiles of the median, while the markers indicate outliers. For low communication range values, no tasks are allocated. At $R_{\mathrm{th}} \approx 0.25$ the network reaches the required degree of connectivity to successfully allocate almost all tasks. For values of $R_{\mathrm{th}} > 0.8$, the task allocation performance again starts to deteriorate. In between these values we find a plateau of good performance.

number of completed tasks. The median is marked in red, the boxes indicate the 25th and 75th percentiles, the whiskers show the lowest and highest data within 1.5 times the interquartile range, with the markers showing outlying data points. For very low values of R_{th}, no tasks are allocated, due to the network being too disconnected. At an R_{th} of around 0.2 units, the network connectivity reaches the threshold where many tasks can be allocated. Task allocation quickly reaches a plateau where all 500 tasks are completed; this stretches from 0.25 to 0.8. The allocation performance then deteriorates for larger values of R_{th}.

To better understand this deterioration, we inspect the allocation overhead graph over the same range of R_{th} (Figure 6.5). The allocation cost (c_{alloc}) is calculated by subtracting the energy spent on successful task execution from the total energy used by the network over the course of a run. At low values of R_{th} we observe a small bulge in the allocation cost. This is due to task components being allocated, but tasks not being completed. Although work is being done, the energy expended does not contribute towards completed tasks.

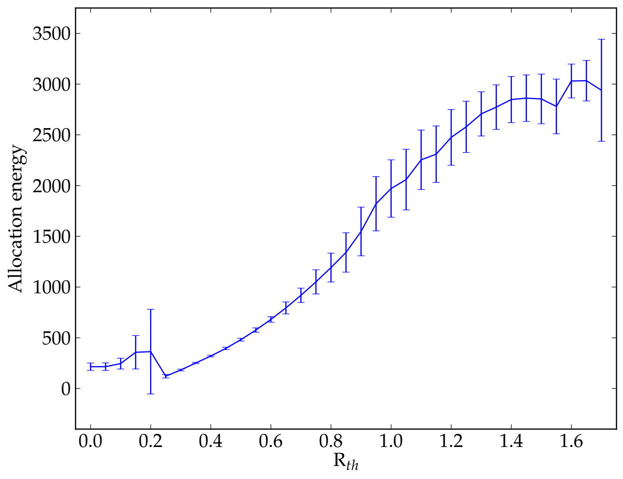


Figure 6.5: Mean allocation cost for range of communication distances (R_{th}). For $R_{\text{th}} < 0.3$, some energy is spent on executing task components, but the network is too sparse to successfully complete tasks, hence the small bump. From $R_{\text{th}} = 0.3$ to $R_{\text{th}} = 0.8$, the network is connected enough to reliably allocate tasks, but the increase in transmission power causes growth of the allocation cost. For $R_{\text{th}} > 1$, the available energy in the system is exceeded, decreasing the number of tasks completed and flattening the allocation overhead graph. The error bars show the standard error over 50 runs.

From $R_{\text{th}} = 0.25$ onwards, the system enters the plateau where all tasks are allocated. At this point the energy overhead is only 100 units, compared to the 2 500 units expended performing the 500 tasks. However, with increasing R_{th}, this overhead continues to grow, until the energy used in task allocation and execution exceeds the total incoming energy in the system, around $R_{\text{th}} = 0.8$. This results in the decrease in task allocation visible for large R_{th} values in Figure 6.4. Once again, this behaviour can be ascribed to the allocation mechanism not assuming a well-connected network, which results in a very large auction community. The correspondingly large number of bids, as well as nodes' attempts at propagating auction announcements, result in a significant amount of traffic. For a well-connected network (large R_{th}), the growth in the size of the auction community for an increase in communication range decreases, resulting in the flattening out of the allocation energy curve for $R_{\text{th}} > 1$.

In terms of the network topologies discussed in the previous section, large values of R_{th} correspond to the fully-connected network topology. Low values of R_{th} result in barely connected components, making the system more similar to the elongated RGG.

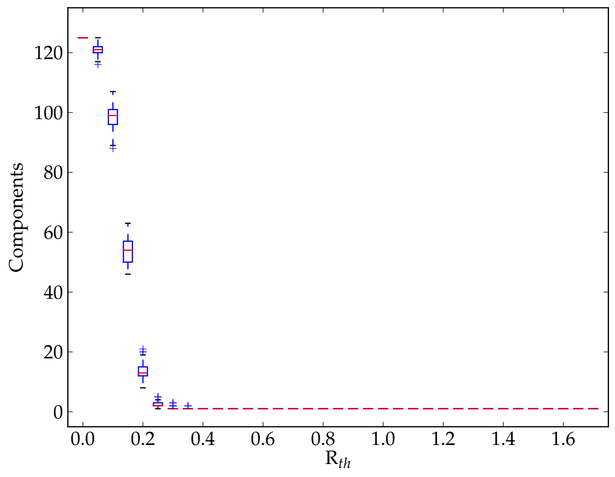


Figure 6.6: Mean number of connected components against the communication distance (R_{th}) for the 50 networks under test. Note the rapid change in connectedness for $R_{\mathrm{th}} < 0.3$: the network usually consists of only one component for R_{th} greater values.

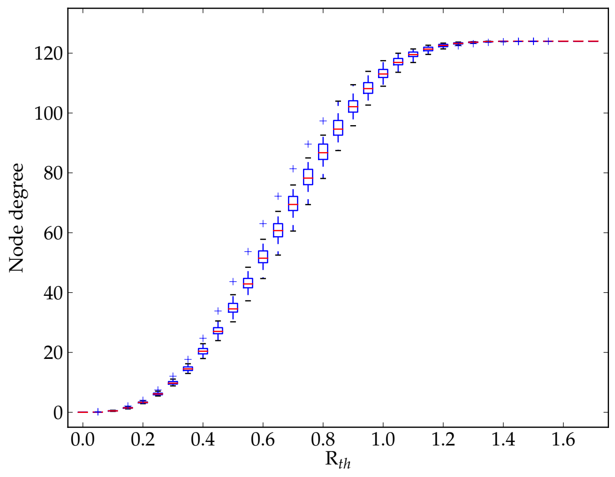


Figure 6.7: Mean node degree against communication distance (R_{th}) for the 50 networks under test. Note that for $R_{\mathrm{th}} < 0.2$ nodes have very few neighbours, while for $R_{\mathrm{th}} > 1.1$ the system is effectively fully connected.

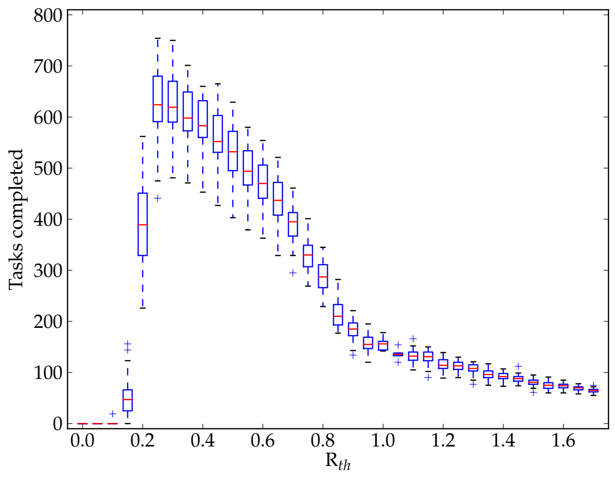


Figure 6.8: Box plot of the number of tasks allocated for a range of communication distances (R_{th}) over 50 cubic random geometric graphs with an increased task load. The boxes show the range of the lower to the upper quartile of the data, with the red line indicating the median. The whiskers show the spread of the data within 1.5 quartiles of the median, while the markers indicate outliers. For low communication range values, no tasks are allocated. The best allocation can be found at $R_{\text{th}} \approx 0.25$, as the required level connectivity has been reached, but communication cost is still minimal. For larger values of R_{th}, the task allocation performance deteriorates rapidly, due to the increased size of the auction community. Beyond $R_{\text{th}} = 1$, performance deteriorates more slowly because the size of the auction community stops growing.

If we investigate the changes in the network structure across the range of R_{th}, we find that for the 125 nodes randomly positioned in space, the topology changes from being fully disconnected when R_{th} is 0, to fully connected for large values of R_{th}. By looking at the number of connected components (Figure 6.6), we see that the system achieves good allocation because of multiple, distributed auctioneers, even if the network consists of more than one component, as is the case for $0.15 \leq R_{\text{th}} \leq 0.25$. From the distribution of mean degrees (Figure 6.7) we can deduce that the network has a large diameter for these values of R_{th}. We can also see that for values of $R_{\text{th}} > 1.1$, the network is fully connected — the size of the auction community therefore approaches the size of the system. These topologies are confirmed by the network diameter measurements (not shown).

6.2.3 Increased task load

The plateau in Figure 6.4 is due to a total load of 500 tasks being submitted in the experiment. We can gain a better perspective on the true shape of the allocation success graph by increasing the task load to exceed the system's capacity. An example of this is shown in the box plot in Figure 6.8 for the same experimental set-up as before, but with the 40 tasks (instead of 10) uploaded to the system every 100 time steps, for a total of 2 000 tasks. The experiment was repeated 50 times, with different node positions for each iteration.

The best system performance can be found at $R_{\text{th}} = 0.25$, as indicated by the peak in the curve. The wide range of measurements in this area is due to the prominent role played by the network topology, as discussed in the previous section. A rapid decrease in performance follows for $0.3 \leq R_{\text{th}} \leq 1$, again due to the increasing size of the auction community. The curve flattens out for $R_{\text{th}} \geq 1$, because the growth in the auction community stops, with only the cost of transmission still increasing. Note that, because of the overhead associated with the large task load, the system capacity is worse than in Figure 6.4.

6.2.4 Discussion

These results clearly demonstrate the importance of determining a suitable communication range for radio networks where communication has a significant cost, specifically in the context of market-based task allocation. There is clearly a range of R_{th} where good allocation can be achieved with minimal overhead. If R_{th} is too small, the network will be too fragmented to allow successful task allocation. On the other hand, very large values of R_{th} require too much power for transmission, and cause the increase in the number of negotiation packets demonstrated in the previous section. The combination of these factors results in exhausted nodes and more allocation failures. In between these extremes we find an area where allocation succeeds, without spending too much on communication. If we look at the total system capacity, a clear peak can be observed where the network is connected enough for successful allocation, and the communication cost is still minimal — this represents the optimal communication distance.

As the lower end of this zone is determined by the connections between nodes, we observe a rapid transition from poor allocation to excellent allocation. The fall-off on the upper end is more gradual because it is a result of the system running out of energy.

The exact location of this zone depends on a number of variables. The start of this zone is determined by the spatial distribution of the nodes, which in turn influences how easily they can communicate. The upper limit is dependent on the difference between the amount of energy entering the network and the energy used by the workload and the communication overhead. Multi-hop communication means that this zone starts well before the network is fully connected. The distributed nature of allocation means that the allocation can succeed, even if the network is split into multiple components. These results reconfirm the suitability of the market-based task allocation system for sparse or disconnected topologies.

System designers would want to adjust communication range to position the system on the plateau of good allocation. The ideal position is just inside the successful allocation zone, towards the end with smaller communication range. In this position the communication overhead is limited, resulting in the most efficient system configuration. Of course, this opens up the possibility in future work of using the nodes themselves to find their individually optimal communication ranges by exploring the task allocation success graph. The distinct difference in allocation success when the communication range is sufficient for good allocation, in comparison with too little or too much distance, lends itself to adaptation on a local level. By adaptively searching for the smallest communication distance where allocation still succeeds, a node could further conserve energy.

6.3 Dynamic topologies

The above experiments describe the behaviour of different network topologies and the effect of communication range on system-level efficiency. However, all of these networks are still static. In this section I build a more accurate picture of a realistic distributed satellite system by introducing a model of the nodes' movements. The mobility model is then used in simulation to show that the task allocation mechanism is resilient to the changes in the network, i.e., allocation in a dynamic environment does not show a decrease in effectiveness from the static case.[2]

In the discussion of node failure (Chapter 5) we also encountered a changing network, which may lead the reader to ask how this scenario differs. When dealing with node failure, the network was eroded by failing nodes, resulting in decreased connectivity and reduced capacity. In contrast, with orbital mechanics the network capacity stays the same, but structural changes are much more

[2]The work in this section was presented at the AI in Space workshop at the International Joint Conference on Artificial Intelligence (van der Horst and Noble, 2011); an updated and extended version was subsequently published in *Acta Futura* (van der Horst and Noble, 2012).

frequent. Previously, the focus was on the effective utilisation of a shrinking pool of resources; in this section I look at large-scale topological change.

6.3.1 Keplerian mobility model

Mobility models are widely used in mobile ad hoc network (MANET) research to represent the dynamic behaviour of a communication network composed of mobile agents (Bai and Helmy, 2004; Camp *et al.*, 2002). As the agents move around, communication channels are formed between individuals in physical proximity to each other; these links are broken again if they move apart. The mobility model allows researchers to map the physical system to an abstract communication network that changes over time.

Existing mobility models frequently rely on random movement to generate a dynamic environment (e.g., Johnson and Maltz, 1996; Choffnes and Bustamante, 2005; Bandyopadhyay *et al.*, 2007). In the case of a distributed satellite system, however, we have good models that describe the dynamic behaviour of objects in orbit: although the formation changes over time, it is not random. Roughly co-orbiting satellites are subject to similar forces, with the variation in their orbital parameters determining their respective trajectories. As a result, we can expect a greater spatial correlation between satellites than can be expected for random movement. Their interactions are also periodic, with approximately the same formation occurring once per orbit.

My mobility model solves the Keplerian equations to obtain the positions of all satellites at a specific point in time; the relative distances between satellites are then used to find the adjacency matrix of the communication network. This adjacency matrix defines how nodes are connected to each other. It is convenient to use an imaginary point on the reference orbit to position the satellites around. This reference point orbits around the planet along with the satellites and serves as an origin for a local Cartesian reference frame. The use of this reference point is primarily a conceptual and visualisation aid, as it focuses the attention on the position of the satellites relative to each other, instead of the planet they are orbiting. The similarity in the satellites' orbital parameters means they orbit in a cluster, but that formation is not actively maintained. The small differences in individual orbits will cause the satellites to drift apart over time, which will require correction if the group is to stay connected. On shorter time-scales the relative positions change dramatically, thereby potentially posing a significant challenge to packet routing and task allocation.

The steps required to determine the connectivity matrix at time t are the following:

1. Calculate the earth-centred Cartesian position u_{ref} of the reference point at time t.

2. Calculate the Cartesian position u_i for all satellites at time t.

3. Centre the coordinate system around the reference point by translating all the satellite positions by $-u_{\text{ref}}$.

4. Calculate the distances between all satellites to obtain the distance matrix.

5. Apply the connection function to the distance matrix to find the adjacency matrix for the communication network.

The connection function captures the propagation characteristics for the underlying communication medium. I will restrict myself to a deterministic radio communication model: if two nodes are within a specified range of each other, we assume they can communicate successfully. To use a more realistic propagation model that incorporates noise and interference, only the connection function needs to be changed.

By repeating these calculations over the course of an experiment, we find the dynamical communication network topology. Note that steps 1 and 3 can be skipped to optimise the calculation of only the communication network topology.

From Chobotov (2002), the position of a satellite can be described in terms of the Keplerian elements as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = [R] \begin{bmatrix} r\cos\theta \\ r\sin\theta \\ 0 \end{bmatrix} \tag{6.3.1}$$

where

$$[R] = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{6.3.2}$$

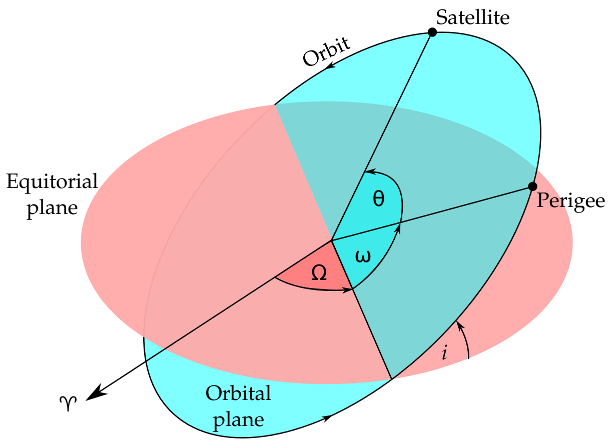


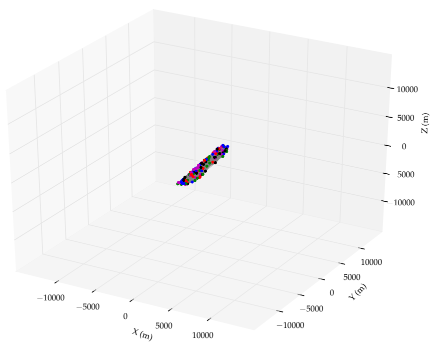
Figure 6.9: Illustration of the orbital plane with Keplerian elements. The orbital plane (blue) intersects the Earth's equatorial plane (pink) with inclination i. The right ascension of the ascending node (Ω) is the longitude of the equatorial crossing measured from the First Point of Aries (♈); while the argument of perigee (ω) indicates the closest point to earth. The position of the satellite is given by the true anomaly (θ).

with

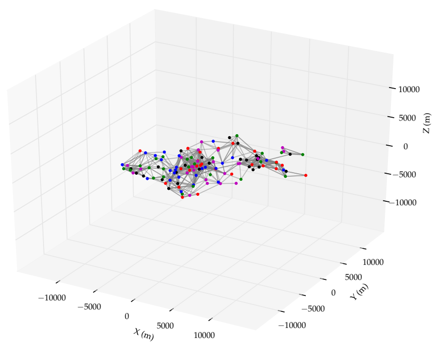
$$\begin{aligned}
R_{11} &= \cos\Omega\cos\omega - \sin\Omega\sin\omega\cos i \\
R_{12} &= -\cos\Omega\sin\omega - \sin\Omega\cos\omega\cos i \\
R_{13} &= \sin\Omega\sin i \\
R_{21} &= \sin\Omega\cos\omega + \cos\Omega\sin\omega\cos i \\
R_{22} &= -\sin\Omega\sin\omega + \cos\Omega\cos\omega\cos i \qquad (6.3.3)\\
R_{23} &= -\cos\Omega\sin i \\
R_{31} &= \sin\omega\sin i \\
R_{32} &= \cos\omega\sin i \\
R_{33} &= \cos i
\end{aligned}$$

where θ is the true anomaly, i the inclination, Ω the right ascension of the ascending node (RAAN) and ω the argument of perigee. The sine term in equation 6.3.1 describes the position of the satellite in its orbital plane (the blue area in Figure 6.9), while equation 6.3.3 defines the orientation of the orbital plane. The size of the semi-major axis (a) and eccentricity (e) of the orbit determines the path circumscribed on the orbital plane.

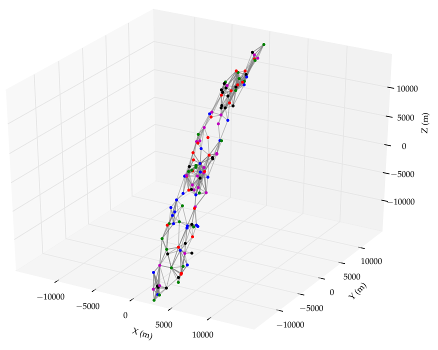
As the Keplerian elements of the satellites vary slightly, they all have different orbital planes which intersect at the centre of the earth. A detailed description on how to calculate the position of a satellite as a function of time



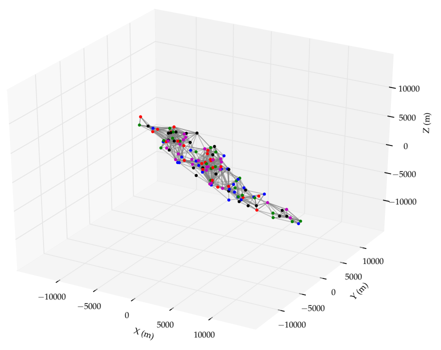
(a) $t = 0$



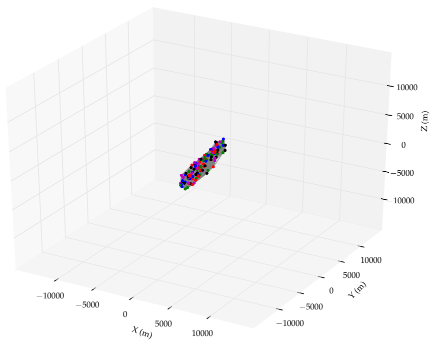
(b) $t = 0.25P$



(c) $t = 0.5P$



(d) $t = 0.75P$



(e) $t = P$

Figure 6.10: Network topology over the course of one orbit for a system of 125 satellites. Note the drastic change in topology, ranging from tightly clustered to sparse and elongated. The colour of nodes indicate specific satellites. P is the orbital period. The parameters used to generate these figures are the same as used for the simulation in Section 6.3.2.

is given in Chobotov (2002), Chapter 4. For the purpose of this discussion, it is sufficient to note the sine and cosine terms in equation 6.3.3. As these terms differ between satellites, their relative positions will change periodically over the course of an orbit.

6.3.1.1 Non-Keplerian disturbances

The above model assumes a spherical potential field, with no external disturbances, which is not strictly true. External disturbance torques, such as geopotential, solar pressure, atmospheric drag, and electro-magnetic forces cause orbits to drift over time. The impact of these forces depends on both the orbits and the physical properties of the satellites. Although these forces have an important effect on the orbits of real satellites, I am primarily interested in the short-term interactions between satellites, over time periods of less than one orbit: how frequently are communication links formed between satellites, and for how long do these links persist? On this scale, the dynamics that result from Kepler's equations dominate the relative positions exhibited by the individual spacecraft.

The Keplerian model therefore captures the communication network dynamics with sufficient accuracy for our purposes; the non-Keplerian perturbations are deliberately ignored.

6.3.1.2 Dynamics

The mobility model displays a complex oscillatory movement of satellites around the reference point, due to the sine and cosine terms in equation 6.3.3. The exact behaviour depends on the orbital parameters and variance thereof for individual satellites. For specific parameter values that suppress the oscillatory terms, stable relationships between satellites can be found, but the dynamic connections dominate the communication network.

The resulting local communication network displays continuous variation in topology and physical scale as spacecraft orbit around the earth, ranging from well-connected to sparse and even disjointed. When the communication range is small compared to the average inter-satellite distance, the network fragments. On the other hand, if the communication range is comparable to the spatial diameter of the satellite cluster, a well-connected network results.

In many cases the network topology ranges between these two extremes within one orbit, as is demonstrated by the network resulting from a slightly elliptical orbit in Figure 6.10. This clearly shows how the satellite formation ranges from tightly clustered, as shown in Figure 6.10a, to being spread over

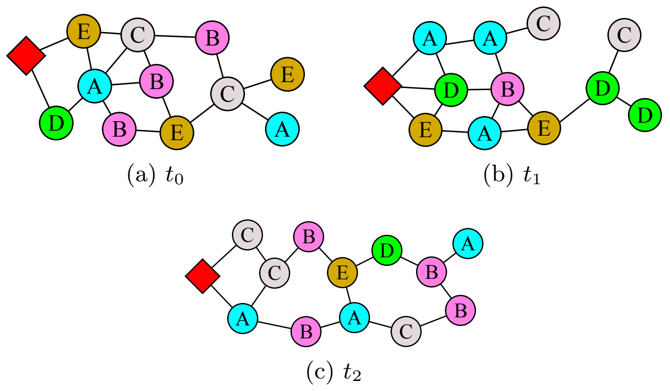


Figure 6.11: Orbital dynamics result in continual change of the local neighbourhood of a node, both in terms of the types of nodes and the topology. The auctioneer is represented by the diamond, while nodes are coloured and labelled according to their capabilities. If the network was static, the community would instead remain constant.

a wide area (Figure 6.10c). Note the similarity between these topologies and the random geometric graphs in Section 6.1 — from this we can deduce that a significant variation in communication cost will be observed at different times during the orbit.

Due to the shifting topology, the local network around a node allocating a task changes continuously in terms of structure and composition, as shown in Figure 6.11. To successfully allocate tasks, these changes need to be detected, or the network mapped, at regular intervals. By using auctions to allocate tasks, and relying on task-centric routing, these changes are transparently detected and the latest network information taken into account.

This mobility model provides the test case for verifying our task allocation mechanism, but it can also serve as a standalone mobility model for researchers in mobile ad hoc networking to test different management and routing algorithms.

6.3.2 Dynamic performance

I am interested in maximising the task allocation and allocation efficiency of the system, because the more efficient the allocation mechanism is, the more energy can be spent on performing payload operations. For this experimental setup, all tasks can be successfully allocated, but the energy overhead due to communication varies. By measuring the communication overhead, i.e., the total energy consumed by the system over time, minus the energy spent on

Table 6.2: Orbital parameters for the simulated distributed satellite system. The reference orbit of the system has the parameters in the reference column, while the values for individual satellites are calculated by adding a uniformly distributed noise with the range shown in the error column.

Parameter	Reference	Error
Semimajor axis a	6878140	± 100 m
Eccentricity e	0.001	$\pm 10^{-6}$ rad
Inclination i	$\frac{\pi}{4}$	± 0.001 rad
RAAN Ω	0	± 0.001 rad
Argument of perigee ω	0	$\pm \frac{\pi}{4}$ rad
Initial true anomaly θ_0	$-\omega$	± 0.01 rad
Orbital period P	5 677	± 0.1 s

tasks, I can measure the impact of network dynamics.

To interpret the simulation results, a fair reference to compare against is required. As the cost of allocation is strongly influenced by the network topology, we need to select a topology that provides a fair comparison, even though the dynamic network changes significantly over time. I therefore determine the topology of the dynamic network at a random time t, then use that as a static network for simulating allocation. If this is repeated for multiple t values, over a number of networks, the mean of the results should give a good indication of the performance without the effects caused by dynamics. This provides the static case.

For additional references I also measure the allocation overhead in two extreme cases: one when the satellites are spread out to the maximum extent to form the sparse case, and the connected case, where they are clustered together and well connected.

6.3.3 Experimental setup

A slightly elliptical, 500km reference orbit is used to define the orbits of 125 satellites, using the orbital parameters in Table 6.2. The initial positions and orbits of the satellites are calculated by adding a uniformly distributed error to the orbital parameters, as listed in the error column in Table 6.2. At $t = 0$, the satellites are clustered around the perigee of the reference orbit, but as they travel around the earth they spread out before clustering together again one orbit later, as shown in Figure 6.10. The connection function uses a simple thresholding comparison: if two satellites are within four kilometres of each other, it is assumed that both parties can communicate with each other, i.e., $R_{\text{th}} = 4\,000$m. A communication delay of 100 milliseconds is assumed: this is generous enough to include a realistic transmission and processing time, even for busy or low-bandwidth nodes. Note that the use of physical orbital parameters necessitates the use of a real time value as well; I therefore depart from the abstract time units used in previous experiments.

Five new tasks are introduced to the system every 100 seconds; each task consists of five components that are executed sequentially by different types of satellites. All five components need to be executed for the task to be considered complete. The system is simulated for one orbit, during which 280 tasks (1 400 task components) are allocated. Executing a task component uses 1 unit of energy, transmitting negotiation packets uses 0.005 units, while transferring a task between two nodes requires 0.5 units. Satellites' energy regenerates at an average rate of 0.005 units per second, to represent the recharging of spacecraft batteries from solar panels. The maximum energy that can be stored by any spacecraft is 10 units. These energy values represent a scenario where communication is cheaper than task execution, but the cumulative energy cost of communication forms a significant portion of the total energy expenditure.

The skills of the satellites are selected with a uniform probability from the set of 5 task component types. Every satellite has only one skill, so the system consists of approximately 25 satellites of each class.

The dynamic case uses the Keplerian movement model to modify the network topology over time. The static case uses 20 different time values to generate different topologies for every run. As the orbits of individual satellites are elliptical and satellites are clustered together at the start, the connected case is found by using the network at $t = 0$. The sparse topology is found halfway through the orbit, at $t = 2\,883$. Fifty runs were used to generate the results presented below. Note that both the composition of the network and the positions of individual satellites were varied between runs.

6.3.4 Results

The total energy used in every run is measured, and the amount of energy spent on successful task execution is subtracted. This allows the calculation of the mean energy used in allocation by the negotiation and transfer packets, as shown in Figure 6.12. The dynamic case required 1 323 units of energy, while the static case required 1 347 units, approximately the same amount. The sparse network used only 1 164 units, while the well-connected case required 1 512 units. The standard error on the measurements ranges from 1.6 to 2.8. For comparison, task execution required 1 400 units of energy. Approximately half the allocation energy is used for transferring tasks, while the remainder is

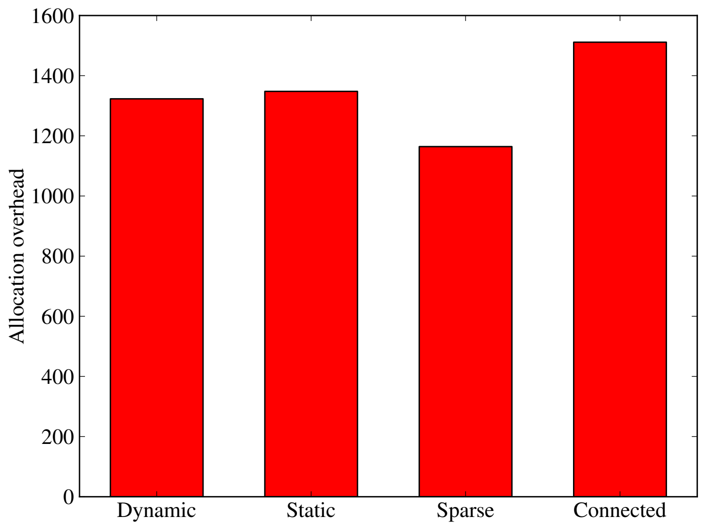
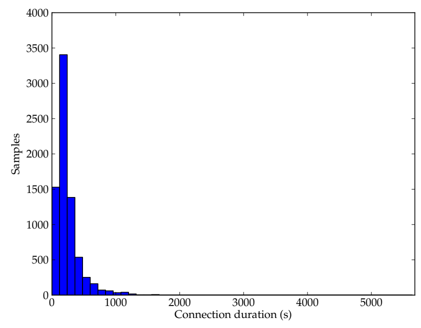


Figure 6.12: Mean allocation overhead using dynamic, averaged static, sparse and well-connected topologies over one orbital period. The dynamic network requires approximately the same amount of energy to allocate as the average static case,indicating that the . Standard error on the measurements ranges from 1.6 to 6.2 (errors bars are not plotted due to their negligible size).

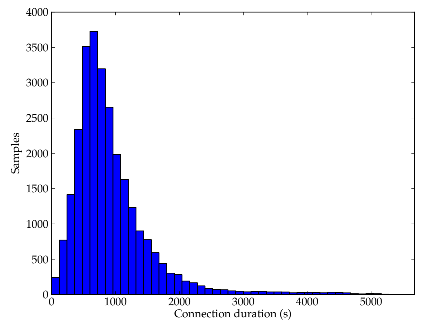
required by the large number of negotiation packets.

The results show that the market-based task allocation mechanism is not adversely affected by the changing communication network; in fact, it slightly outperforms the averaged static case. The reference cases clearly demonstrate how the cost of allocation can vary within one orbit. The high cost associated with the well-connected case can be ascribed to the large auction community resulting from the small network diameter, as discussed in Section 6.1.

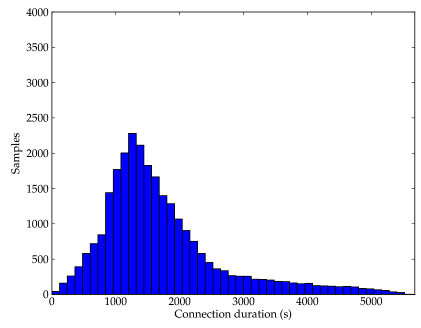
To understand how the allocation mechanism manages to maintain performance in spite of the changes in the network, it helps to look at the lifetimes of connections in the system. Figure 6.13b shows a histogram of connection lifetimes: connections between nodes generally last on the order of hundreds of seconds, orders of magnitude more than the auctions require. The network state information is therefore valid for the duration of the auction, allowing allocation to succeed. For comparison, the connection lifetimes for R_{th} values of 1 000m and 8 000m were also calculated, and are displayed in Figure 6.13. For smaller values of R_{th} the peak of the histogram moves to the left: there are fewer connections made and most only last for a short period, resulting in a relatively volatile network. As R_{th} increases the number of links and their



(a) $R_{\text{th}} = 1\,000\text{m}$



(b) $R_{\text{th}} = 4\,000\text{m}$



(c) $R_{\text{th}} = 8\,000\text{m}$

Figure 6.13: Histogram of connection lifetimes for a group of 125 satellites with 1km, 4km and 8km communication ranges, over the course of one orbit. Note that the vast majority of connections in the 4km case last for hundreds of seconds or more, thereby allowing auctions to succeed. For a smaller R_{th} value, the number of connections and their lifetimes decrease. For larger values of R_{th} we observe an increase in the average connection lifetime, to the point where some connections persist for the duration of an orbit or longer.

stability increase too, shifting the distribution to the right, even resulting in links that persist for longer than the duration of an orbit. If the communication range is increased even further the network will become fully connected.

Tests using a different number of satellites, or placing them in other orbits, show similar results, as long as the component satellites have approximately similar orbits. The network volatility is a result of the error with respect to the reference orbit, which determines how far satellites move apart, and the communication distance, which determines for how long connections can be maintained.

6.3.5 Discussion

This experiment demonstrates that ad hoc, decentralised task allocation and task-centric routing allow us to successfully manage task allocation in distributed satellite systems with realistic network dynamics.

The similarity in performance between the static and dynamic cases can be ascribed to the ad hoc nature of the task allocation mechanism, as well as the differences in time scales between allocation and network changes. In addition, the allocation mechanism has limited memory about the system state (only an average price for a task type), therefore it can adapt much faster than the network changes. By holding an auction for every task component, the current state of the local network is always used to determine allocation. This approach is best suited to scenarios where the network changes more frequently than tasks are allocated — the cost of holding an auction should be less than the cost of incrementally tracking changes in the local community.

Despite the continuous movement of satellites, the average connection lifetime between two nodes in the communication network is significantly longer than the duration of an auction. From the point of view of the auctioneer, the local network is therefore effectively static during an auction, thus allowing successful allocation. If network changes were to disrupt an auction, the auctioneer can restart the auction to retry, with a high probability of success.

6.4 Discussion

This chapter investigated the impact of topological structure and dynamics on system-level performance. I measured the allocation cost associated with different network topologies and explored the trade-off in communication range versus allocation overhead. Finally, a mobility model of a multi-satellite system was developed to measure the impact of realistic orbital dynamics on the communication network topology and the task allocation mechanism.

The experiments demonstrate the significant role played by the size of the auction community in determining the total allocation cost — this is a direct result of the flooding of auction announcement packets through the network. Systems with a large auction community will be more expensive in terms of task allocation, while sparsely-connected systems exhibit a significant saving in communication cost, which leaves more energy for completing tasks. In terms of communication range, we observe a distinct plateau of good allocation of a range of R_{th} values, where all the tasks in the experiment could be allocated. The lower end of this plateau is determined by the physical properties of the network, while the upper end is largely a result of the energy used by the task allocation mechanism. The system capacity shows a prominent peak at the point where the network is connected enough to achieve good allocation, yet still has low communication cost. The mobility model shows that the communication topology for a distributed satellite system can vary greatly over the course of an orbit. The differences in orbital parameters result in a continuous but cyclic change in the network. Despite the constant change, the task allocation mechanism shows no deterioration in performance when compared to a static equivalent. This result seems counter-intuitive, but makes sense when we consider that there is no model of the network that might become outdated, and the time scale of each auction is short enough that it effectively occurs on a static network.

These experiments demonstrate how a system designer can go about measuring the trade-offs in terms of topology and communication cost. Ideally, the communication range and auction community size should be just large enough to ensure reliable allocation. The lower bound on the zone of good allocation is clearly visible as you move along the axis of increasing communication power; increasing it significantly further will not result in an improvement in system performance.

From a satellite engineering perspective, the primary result is that my task allocation mechanism copes very well with the dynamic topologies that result from realistic orbital mechanics: no negative impact is observed. Furthermore, this resilience of the allocation mechanism to a changing network topology implies that the requirements for fine-grained control of individual spacecraft can partially be addressed on a network management level: instead of accurate formation maintenance, coarse positioning of spacecraft to stay part of the network is sufficient. This makes the use of smaller, simpler spacecraft viable in distributed satellite scenarios, making it a more affordable enterprise. For system design, the ease with which the task allocation mechanism manages network dynamics suggests that a large part of the modelling can be done using a representative set of random geometric graphs as topologies. These will provide a significant saving in computational cost if heuristic optimisations are performed.

The above results also relate to other distributed system problems. The explicit focus on communication cost as a result of topology also features prominently in wireless sensor networks, where similar trade-offs are required. The adaptability to a changing network topology that stems from the use of ad hoc information should apply equally to dynamic networks in WSNs. This raises the question of when the benefits of ad hoc information are outweighed by the cost of repeatedly gathering this information. The other relevant link is to work on mobile ad hoc networks. The mobility model for distributed satellite systems now provides us with a test platform to compare different routing and control protocols developed specifically for terrestrial MANETs in our system.

The high cost observed for well-connected topologies suggests that the allocation mechanism can be made more efficient by using an adaptive time-to-live range – this will however require a mechanism to learn more about the network on a node level. Monitoring the number of successfully allocated tasks, or the volume of auction announcements received should provide a useful starting point. The periodic nature of topologies produced by the Keplerian mobility model also raises the question of whether the periodicity can be exploited to optimise management of the system.

Although I believe that it is premature to draw firm conclusions, the slight improvement observed in the task allocation cost of the dynamic network could be a beneficial effect of systems with a constantly changing topology. Although dynamics are usually regarded as a complicating factor in system management (Durfee, 2004), the resulting change in the auction community means that failed auctions are retried on a new set of nodes. Therefore, retried auctions can utilise different nodes on every attempt, instead of waiting for nodes to recover to a usable level. Further investigation is however required to determine whether this phenomenon will have a significant effect on system performance.

7 Competitive markets

The preceding chapters are primarily concerned with the specifics of the proposed market-based task allocation mechanism; in this chapter the focus shifts to address broader questions around market-based control as a methodology. The allocation mechanism in previous chapters is very much under the control of the system designer, and can be seen as a rather artificial market. The designer defines the utility function of individual nodes, and forces them to bid in an informative manner. Although this results in the desired allocation, it also raises the question of how a more "open" market with true competition would perform. If agents could determine their own bidding strategy, would this result in a better or worse allocation when compared to the utilisation-based approach explored in previous chapters?

Proponents of the free market would suggest that a competitive market could lead to further optimisation and increased adaptability. However, markets can under certain conditions exhibit behaviours that are undesirable from a control point of view (Hogg and Huberman, 2002). The simulation platform developed over the course of this thesis allows us to put this to the test: how does the management of a system using "enforced socialism" compare to self-interested capitalism?[1]

The objective of this chapter is to compare the behaviour of a competitive market to one using utilisation-based bidding. I am interested in how and why they differ, and what the methodological implications are for market-based control in general. To achieve this I develop an adaptive agent for a first-price sealed-bid auction with common-value goods. The agent is subject to the same communication constraints and auction mechanics used in previous chapters. To investigate the relative performances of the two approaches, the number of tasks allocated is measured for a fully-connected communication topology, as well as for a range of less well-connected networks.

[1]In recent years this question has received an increasing amount of attention as "the price of anarchy", especially in the context of network routing and congestion games, e.g., Roughgarden (2005)

To distinguish between the task allocation mechanism presented in previous chapters and the competitive agents presented in this chapter, I introduce the terms *competitive* and *cooperative*. Competitive or profit-based allocation indicates a market where agents adapt their prices to maximise their own revenue, within the bounds allowed by the available energy. If, however, agents use only their energy levels to calculate bid values, as discussed in previous chapters, I will refer to cooperative or utilisation-based markets.

7.1 Motivation

My exploration of competitive systems has two distinct motivations, which are discussed in turn below. Firstly, from a pragmatic, engineering point of view, open systems are sometimes unavoidable: they are a fact of life. Secondly, as part of a methodological investigation, competitive systems represent a natural conclusion as we move along the axis of increasingly distributed control.

There are cases where the system designer does not have the level of control required to define the interactions, motivation, strategies and behaviour of individuals in his system. As part of the qualification effort that flows through this thesis, I am therefore forced to ask how this constraint of limited control impacts on the design. More specifically, how does using a competitive market to allocate tasks differ from a cooperative one in terms of performance and suitability to different parts of the problem space? For example, it is quite possible that multiple stakeholders might collaborate in constructing and managing a distributed satellite system. As the size of the system increases, the likelihood of different, self-interested parties joining it increases too. For these multi-stakeholder scenarios a system of fair compensation according to the work done must exist. A natural way to express agents' contributions, while still allowing for individual strategies, is to relate the bid values to actual revenue for their owners. Compensation is therefore related to the demand for certain skills, and the efficiency with which agents convert energy to work. Because stake-holders are self-interested we can expect them to optimise their agents' bidding behaviour to maximise the revenue they earn, while presumably minimising their contribution to the welfare of others. This is in contrast to cooperative markets where bidding is used purely as a signalling mechanism to coordinate system-level behaviour.

The second motivation stems from a view of open systems as the natural conclusion to distributed control. If we have a multi-component system we want to control, the most centralised solution involves a single controller directly managing all the other agents in the system. A more distributed approach allows local decision making, but with a clearly defined global objective. If we continue along this axis, we find open systems, where individuals have no unified goal, but instead everyone “is continually making to better his own condition”, to use Adam Smith’s phrasing (1776). Individuals decide how to bid, and have their own private incentives that they optimise for. I therefore compare competitive to cooperative markets to determine whether some parts of the parameter space are more suited to one system or another.

7.2 The structure of a competitive market

To convert the previously described market-based task allocation mechanism to a competitive market, the bid calculation function needs to change to allow agents to actively try to win tasks, instead of conveying their degree of utilisation. To keep the two systems comparable in terms of performance, the types and number of communication packets allowed must remain the same. I therefore retain the structure of the utilisation-based market, including the first-price auction structure, communication constraints and optimisation features such as bid aggregation; the only change is in how agents decide how much to bid.

The auction structure remains the same as before: an agent with a task announces it to his immediate neighbours, who relay it to their neighbours until the time-to-live of the announcement packet is exceeded. Agents with enough energy and the necessary skills submit bids back towards the auctioneer. At every node along the way, bids are aggregated and the best bid forwarded. With every relay of a bid message, the bid value is multiplied by a fixed commission value. The auctioneer selects the lowest bid, and assigns the task to the corresponding agent, and the same sequence as before follows: task acceptance, task transferral and acknowledgement packets. If the task consists of multiple components, they are once again outsourced to other agents in subsequent auctions. The commission structure and distributed winner calculation is therefore still applicable, and allows the different approaches to be compared. I restrict the discussion to tasks of a constant size for all agents, which results in particular market dynamics — I will point these out through the course of this chapter. The same measures of success as before are used again: how many tasks are successfully completed, and how much energy is required to achieve the allocation?

One significant implication of this market structure is that very little information is available for agents to base their bids on. In most other work using sealed-bid auctions all agents are informed of the closing price of the auction (e.g., Bagnall and Toft, 2006), but due to the communication cost associated with notifying the network, this information is not available in my system.

Agents are paid for contributing to completing tasks; in other words, converting energy to work. As explained in the previous section, these payments are used to calculate the revenue of the agents' owners. Energy is replenished at a constant rate, but the agent can only store a finite amount. There is no incentive to store energy for future use, or to abstain from bidding in the hope that prices will rise. The best strategy is to win as many jobs as possible, as long as an agent still makes a profit completing them.[2]

Instead of distributing tasks to evenly share the load, a competitive market favours the agent most adept at winning jobs. Due to competition and constant task sizes for all agents, we expect the competitive market to converge to an equilibrium price, with effectively no profit for the agents involved. For identical agents in the equilibrium condition, the allocation should be random with a uniform distribution, as no agent has a competitive advantage over anyone else.

Note that I am primarily interested in the performance of the system as a whole (measured in terms of task completion) for the purposes of decentralised control. This is in contrast to a focus on agent strategies, auction revenues, or social welfare measurements that frequently dominate in agent-based economics. The performance of the system is defined in terms of the work completed, not the utilities of the individual agents.

The competitive agents focus on maximising their own profit, which can result in sub-optimal system-level performance. For example, if a successful bidder exhausts itself by winning a series of jobs to the point where it has insufficient energy to transfer tasks, it can prevent access to a part of the network that could otherwise be used, thus reducing the total system capacity. This behaviour is discouraged in the utilisation-based allocation mechanism by having agents increase their bids to reflect decreased energy levels. In economic terms this decrease in performance can be seen as an externality, i.e., a cost that is not captured in the allocation prices. This incentive structure, where selfish behaviour is incentivised, possibly to the detriment of the global system, is reminiscent of the "tragedy of the commons" described by Hardin (1968), and much analysed in the economic literature (e.g., Marwell and Ames, 1979; Mason and Phillips, 1997) and evolutionary biology (e.g., Axelrod and Hamilton, 1981; Killingback *et al.*, 2006) since then.

In economic terms, the competitive market is analogous to a hyper-capitalist system with purely self-interested actors, where no-one is responsible for the global good, yet everyone's actions determine it. In contrast, the utilisation-based market is close to a socialist utopia, where everyone contributes according to their abilities. In both cases the system uses local information for decision making, but in the socialist system, everyone has the global good at heart.

[2]This approach views the energy used in completing tasks as a cost, because the same energy could in theory have been spent on a different task for which the agent would have been paid.

7.2.1 Why not mechanism design?

A reader with a background in market design may at this point wonder: "Why not simply use mechanism design to achieve the desired allocation?" While mechanism design is undoubtedly a valid approach to the problem, it is not the objective of this chapter. A mechanism design approach would focus on incentivising desirable behaviours, usually by changing the rules of interaction. However, this is not always possible, for example in established systems, or in the case of distributed satellite systems, where we want to minimize communication. While mechanism design could be used to find a market-structure that incentivises the desired behaviour, I am more interested in how the performance of the utilisation-based allocation mechanism defined in Chapter 3 changes when we move to an open market, and whether specific dynamic properties emerge as a result.

This chapter should therefore be seen as an investigation into the nature of auctions for market-based control, specifically when taking place in spatial networks. The following statement by Milgrom (1985) neatly summarises my intent with this chapter:

> Too much recent research effort in auctions has been simply applying the latest techniques (principally "mechanism design") to ever more complicated models; too little has been devoted to the very real and important economic questions that auctions raise.

If we were to pursue a mechanism design approach, one possible solution would be to rely on a payment structure similar to the bid calculation used by the utilisation-based agents. To maximise the number of tasks allocated across the system, a payment rule is required that will couple the global system performance to local performance, by scaling the amount of money an agent receives by the total number of tasks completed by the system. One way of estimating the system-level utilisation using only local information, is to take the agent's own energy-levels into account which leads us back to a utilisation-based bidding approach.

7.2.2 Desired allocation behaviour

The desired allocation behaviour requires the careful balancing of three different objectives. The most obvious one is to minimise the allocation cost by allocating tasks close to the auctioneer. Secondly, the system simultaneously needs to prevent node exhaustion by spreading the allocation of tasks across a number of agents. Finally, we need to maximise the incident energy in the system.

Recall that agents have a limited energy storage capacity: once their batteries are fully charged, they cannot store any more energy. In order to maximise the incident energy on a system level, we therefore want as many agents as possible to be capable of storing the incoming energy. In other words, it is preferable to have two partially-used agents to a situation where one is fully-utilised and one not at all, because when both are used, the system as a whole can receive more incoming energy. Note that this condition only plays a role while some of the agents are not utilised — when all agents are in use, they can all receive incoming energy.

7.3 Trading agent design

The majority of work on adaptive agents focuses on continuous double auctions, where both the buyers and the sellers adjust their prices (e.g, Vytelingum *et al.*, 2004; Chaggar *et al.*, 2008). These auctions are efficient at allocating resources, and are widely used in the real world. For the purposes of task allocation in distributed satellite systems however, the combination of communication cost and task structure makes the use of a single-sided auction more attractive. As described in Chapter 3, a reverse, first-price, sealed-bid auction allows allocation with minimal communication.

While a large number of agents with varying degrees of complexity have been developed for continuous double auctions, single-sided auctions have received much less attention. Brandt and Weiss (2002) investigated the effects of antisocial agents in a simple task allocation scenario to demonstrate the vulnerability of second-price sealed-bid auctions to competitors whose main goal is not maximising profit, but inflicting losses on the other agents in the system.

In work that is highly relevant to this chapter Bagnall and Toft (2004, 2006) adapted agent learning strategies used in continuous double auctions to sealed-bid auctions. Their implementations of ZIP-traders (Cliff and Bruten, 1997, 1998) and the history-based agents proposed by Gjerstad and Dickhaut (1998) take the different information revelation and allocation processes for single-sided auctions into account.

As the focus of this chapter is on market dynamics, rather than the strategies of individual agents, I restrict my attention to reactive agents. More specifically, I adapt the single-sided ZIP-traders in Bagnall and Toft (2006) to my task allocation scenario in order to accommodate three significant changes in the market structure. Firstly, for task allocation, we are dealing with interdependent goods, because agents have to outsource subsequent task components.

Agents therefore need to learn an appropriate outsourcing cost in addition to the market price, i.e., what other agents are bidding. Bagnall and Toft looked at sealed-bid auctions where all agents could participate in auctions. In contrast, my own focus on communication costs has led to transactions that execute on a network with a specific topology, which decreases the information available to agents. Furthermore, agents in the task allocation auction are not informed of the winning bid price, which complicates the learning process. Instead of using an estimated optimal bid to update their profit margins, they have to search for the price point where their bids succeed.

Our competitive market should exhibit similar dynamics as would be expected from a real-world market consisting of self-interested individuals. Relevant behaviours include learning of the market price; competition between agents by varying bid prices; sensitivity to changes in the cost of task components over time; and, on a macro-level, price behaviour that corresponds to that predicted by competition economics. When the workload is less than what could potentially be performed by the system, i.e., supply exceeds demand, we expect the market-price to converge to a zero-profit state. Conversely, an overloaded system will be characterised by bid values and task costs increasing until agents run out of energy.

7.3.1 Learning strategy

This study employs a reactive agent that uses only its internal state and the state of the network to determine bid values, without any strategic planning. The trading agent strategy is similar to that used by ZIP-traders, but needs to be adapted for single-sided sealed-bid auctions in which the winning bid value is unknown.

An agent knows its internal energy cost to execute a unit task (c_{int}), but it has to learn the outsourcing cost (c_{os}) and a suitable profit margin (μ) for its position in the network. As the true outsourcing cost is highly dependent on the behaviour of other agents in the market and only observable through actual outsourcing of task components, the agent maintains an estimated outsourcing cost ($\hat{c}_{\text{os}}$), which it uses to calculate the bid value (B):

$$B = c_{\text{int}} + \mu + \hat{c}_{\text{os}} \tag{7.3.1}$$

The only environmental information available to an agent is its own bidding history, whether a particular bid was successful, and the outsourcing cost associated with previous task allocations. It must use this information to learn suitable values for μ and $\hat{c}_{\text{os}}$, while these values are changing due to interaction with other agents. For convenience of analysis I retain the fixed commission factors used in previous chapters: although agents earn money from relaying tasks, they do not have to learn a suitable commission value.

I assume agents to be rational, i.e., they will not knowingly bid values that will cause them to make a loss. They will therefore not bid below their internal cost ($B > c_{\text{int}}$) and they will never lower their outsourcing cost estimate to be negative ($\hat{c}_{\text{os}} \geq 0$). Finally, I also stipulate that the profit margin likewise can never be negative ($\mu \geq 0$).

The learning problem can be illustrated by considering an auction from the perspective of an agent. A new task is announced, so the agent calculates a bid that takes its internal cost and estimated outsourcing cost into account, and adds some profit margin to the bid. It then waits for a response from the auctioneer. After a predetermined period of time the agent still has not heard from the auctioneer, and must therefore assume that its bid has failed. In the next auction, it will try a lower bid, but should this be done by decreasing its profit margin, or its outsourcing estimate? One way to get an estimate of the market price and outsourcing costs is to aggressively submit a low bid that can win the auction, but there is always a danger the outsourcing costs may be high enough to cause the agent to make a loss on the transaction.

ZIP traders face a similar challenge, but only in one dimension. They retrospectively estimate an optimal bid for the precious auction, then update their profit margin by the difference between their actual margin and the desired margin. The margin updates are smoothed by an exponential infinite impulse response filter (the momentum coefficient used by Cliff and Bruten, 1998), which determines the learning rate of agents. For Bagnall and Toft's single-sided-auction ZIP traders, the optimal bid is derived from the publicly known winning bid value.

As my agents do not have this information available, they have to rely on a probing strategy to find a suitable bid value over multiple auctions. A simple gradient descent rule allows them to find and converge upon the market price. This process can be visualised as the traversal of a two-dimensional space, as graphically represented in Figure 7.1. The horizontal axis corresponds to the sum of the agent's internal cost and profit margin, while the vertical axis represents the estimated outsourcing cost. The bid value is given by the sum of the x and y coordinates. If the bid is lower than the other bids in the auction, the agent wins the task. The market value of the task is represented by the blue line in Figure 7.1: it represents the combinations of values for ($c_{\text{int}} + \mu$) and $\hat{c}_{\text{os}}$ that would result in the agent winning an auction. Bids above the line

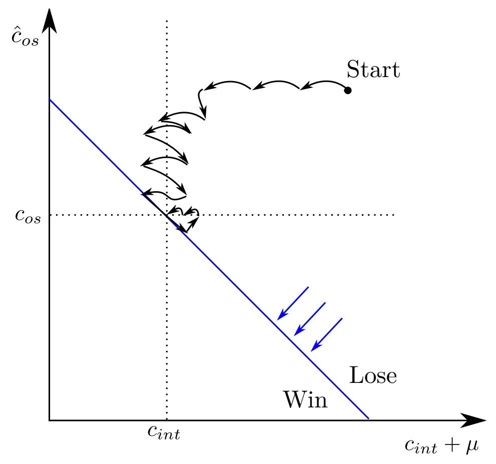


Figure 7.1: Visualisation of the learning strategy used by competitive agents. The horizontal axis represents the sum of the internal cost (c_{int}) and the profit margin (μ), while the vertical axis represents the estimated outsourcing cost ($\hat{c}_{\text{os}}$). The bid value is the sum of the x and y coordinates. The market price is indicated by the blue line: if the bid value is lower than the market price, the agent will win the job, while bids above the line will lose. The arrows indicate a series of changes to μ and $\hat{c}_{\text{os}}$ while an agent searches for the market price. In the zero-profit condition, the market price is the sum of c_{int} and the true outsourcing cost c_{os}.

will lose.

For every auction, the agent computes a bid value based on its previous experiences. At the end of the auction, the agent updates the variables based on whether it managed to win the task or not, thereby moving to another point in parameter space. As this process is repeated over a sequence of auctions, the agent approaches the market equilibrium price. The market-price line will gradually also move downwards until it reaches a zero-profit state ($\mu = 0$), as the agents are forced to bid below the previous market prices.

How can the agent go about learning the correct values? I will start with a simple margin update rule borrowed from ZIP traders adapted to reverse auctions: if an agent wins an auction, it knows it has bid less than all other agents; it therefore increases its bid value in the next auction to increase its profit margin. Similarly, if an agent loses an auction, it knows that it should bid somewhere between its internal cost and its previous bid, i.e., it should bid less in the subsequent auction.

The profit margin μ is used as the primary parameter for adjustment: it is changed more rapidly and is used to explore the space, while $\hat{c}_{\text{os}}$ changes relatively slowly. This is because, although both $\hat{c}_{\text{os}}$ and μ contribute equally to the bid value, the information we have about the variables is asymmetric. The profit margin is expected to fluctuate as agents converge on the equilibrium price, while $\hat{c}_{\mathrm{os}}$ is based on observations of actual task outsourcing. However, if μ is 0, it cannot be decreased any further and therefore the agent adjusts $\hat{c}_{\mathrm{os}}$.

The outsourcing cost estimate is updated whenever an agent allocates a task. If we regard the allocation cost as a noisy signal of the true outsourcing cost once the allocation prices have converged, repeated observations will allow us to find a good estimate of the outsourcing cost. The agent filters the outsourcing cost estimate using a first-order infinite impulse response filter with coefficient $\gamma_{\mathrm{os}} \in [0, 1]$. For $\gamma_{\mathrm{os}} = 1$, the agent has no memory, and always uses the most recent outsourcing cost value as $\hat{c}_{\mathrm{os}}$.

If we look at Figure 7.1, it is clear that the quickest path to the market price is not to be found by moving in a horizontal direction by decreasing the margin and then down by decreasing the outsourcing cost. Instead, it would be found by directly to the nearest point on the win-lose line. This would imply that both μ and $\hat{c}_{\mathrm{os}}$ are decreased simultaneously. However, the relative magnitude of the decrease is unknown. To accommodate this, I therefore introduce a timeout n_{os} which also triggers an update of $\hat{c}_{\mathrm{os}}$: if the agent has failed to win a job in more than n_{os} successive auctions, it decreases $\hat{c}_{\mathrm{os}}$. This is to compensate for the unfortunate state of affairs where the agent has a wildly inaccurate $\hat{c}_{\mathrm{os}}$ value, as it has won no auctions. It therefore has not had the opportunity to conduct an outsourcing auction and thus has no outsourcing cost information.

Finally, to describe the magnitudes of updates, every agent has its own adjustment size variables, one for the profit margin (Δ_{μ}) and one for the outsourcing cost estimate (Δ_{os}). These are used to generate random values with which to increase or decrease the parameters. The magnitudes of the adjustment variables determine how quickly an agent will traverse the parameter space, but larger values can also result in noisy bid values, with significant under-bidding. As this is a purely reactive agent, it is necessarily simple (and sometimes naive) in how it calculates bids. More advanced strategies are certainly possible, but they fall beyond the scope of this chapter.

Note that an agent will only place a bid if it has enough energy to complete the task; its energy level does not contribute to the calculation of its bid value. Once an agent has committed to a task, it has to complete the relevant task components and outsource the others. If it cannot outsource the remaining task components, the agent will suffer the energy penalty of completing a task component and it will not receive any payment, as the task was not completed.[3]

[3]This assumes that an accounting layer exists that tracks where tasks are executed, which

The update rules can be summarised as follows:

1. If the agent wins an auction, μ is increased by a value generated uniformly at random from the range of $[0, \Delta_\mu]$.
2. If the agent loses an auction, μ is decreased by a value generated uniformly at random from the range of $[0, \Delta_\mu]$.
3. If $\mu < 0$, decrease $\hat{c}_{\text{os}}$ by $|\mu|$ and set $\mu = 0$.
4. When allocating a task, update $\hat{c}_{\text{os}}$ with the amount paid (P) to outsource the task:
   $\hat{c}_{\text{os}}\ (t+1) = \hat{c}_{\text{os}}\ (t) \times (1 - \gamma_{\text{os}}) + P(t) \times \gamma_{\text{os}}$
5. If the agent has not won any auctions in the last n_{os} auctions, decrease $\hat{c}_{\text{os}}$ by a value generated uniformly at random from the range $[0, \Delta_{\text{os}}]$. Set $n_{\text{os}} = 0$.
6. If $\hat{c}_{\text{os}} < 0$, let $\hat{c}_{\text{os}} = 0$.

7.3.2 Dynamic behaviour

With the auction structure and learning strategy of the competitive agents defined, we can now consider the dynamics that result from their interaction. If a population of these agents are pitted against each other, they exhibit a market-like dynamic that can serve as a valid model of a competitive market. The winning bid prices for an example market using competitive agents are shown in Figure 7.2.

If a group of agents were to start with random values for μ and $\hat{c}_{\text{os}}$, those with low bid prices will initially win. As they outsource subsequent task components, they learn the outsourcing prices charged by other agents. More accurate estimates of the outsourcing cost therefore propagate from the agents at the end of the outsourcing chain, back up to the first agents. Agents essentially converge to an accurate estimate of the outsourcing cost ($\hat{c}_{\text{os}} \to c_{\text{os}}$), while also competing to win jobs by lowering their margins ($\mu \to 0$).

As a result, the bid prices are driven towards a zero-profit condition, where agents bid the sum of their internal cost and the subsequent outsourcing costs. This is shown in Figure 7.1 where the market price line intersects c_{os} and c_{int}. The imperfect information about the outsourcing cost means that agents will regularly underestimate the market price, and bid too low. Although this allows the agent to learn the correct outsourcing cost, the agent may make a small

agent owners should be compensated, and by what amount.

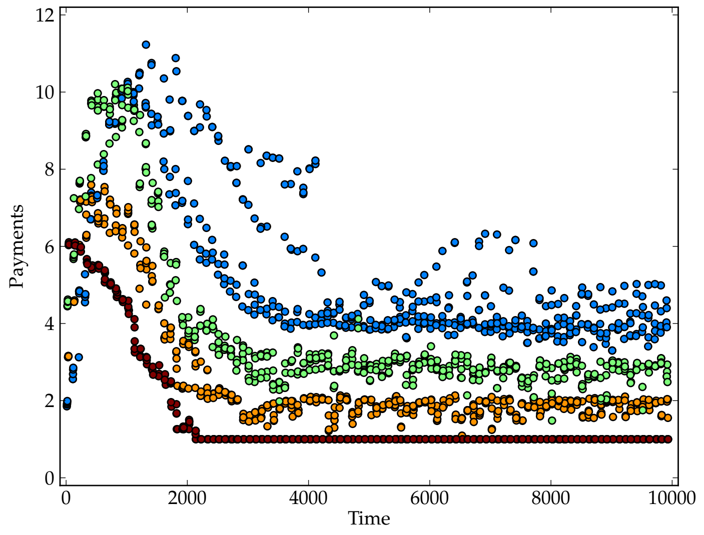


Figure 7.2: Winning bid prices over time for an example market of competitive agents. Tasks consist of four sequential components; the price for each is shown in a different colour. The blue component requires a green one to be outsourced, which in turn results in a yellow one, and finally a red component, at which point the task can be regarded as complete. Competition between agents drives market prices from initially random values to the zero-profit condition, where the bid price is the sum of an agent's internal cost (1 in this case) and the subsequent outsourcing costs. The final component has no outsourcing costs, and therefore converges on a cost of 1. The prices for other components are noisier, due to uncertainty about the outsourcing cost.

loss in the process. As a result of this uncertainty, the bid prices for task components with outsourcing requirements vary around the convergence point. The prices of task components with a longer outsourcing chain will vary more, because of the accumulated uncertainty in the prices of subsequent components. This phenomenon is clearly visible in Figure 7.2. Note that the convergence to the zero-profit condition results from all agents having the same c_{int} values.

If all agents have the same internal costs we find that allocation is randomly spread between them in a round-robin-like fashion, as no agent holds a competitive advantage. This convergence condition is promising from a task allocation perspective because it provides a mechanism for distributing labour between similar agents.

The rate at which the market learns the zero-profit price is determined by the magnitude of Δ_μ and Δ_{os}, as well as the value of n_{os}. The latter variable is however dependent on the number of agents in the system: it reflects the probability that the agent will win a task in the zero-profit state. Once the market has converged to an equilibrium price, for n' in the auction community, the agent should win on average $\frac{1}{n'}$ of the auctions. However, as the number of nodes in the network and the size of the auction community are both unknown, a good value for n_{os} can only be found on very long time scales.

Agents make most of their money by how well they respond to major changes in the market: if they are good at adapting to a new market price, they can use that information to increase their revenue. The bidding success of an individual is determined by the values of its learning parameters as well as its initial guesses for μ and $\hat{c}_{\text{os}}$. A large profit adjustment step, for example, will enable the node to quickly adapt to the market price, but it may also cause aggressive underestimation of the price once prices have converged, decreasing the earnings of the agent. Furthermore, winning a task will give an agent up-to-date information about the market price and outsourcing costs, thus giving it a competitive advantage over agents who have not recently won a task.

There is much scope for optimising the parameters of individual agents to perform well in a market, as done by Cliff (1998). My interest is however with system-level behaviour, so I will limit my discussion to pointing out that some parameter values will obviously result in poorly performing agents. Too much memory ($\gamma_{\text{os}} \to 0$) means that agents will be very slow to learn outsourcing costs. Similarly, if the adjustment parameters (Δ_μ and Δ_{os}) are too small, converge will take many auctions. If, however, the adjustment steps are large, a noisy convergence state will result, with significant underestimation of the market price and oscillation around it. Finally, if the outsourcing timeout n_{os} is too small, agents will aggressively lower their outsourcing estimates, until the market price converges on c_{int} because $\hat{c}_{\text{os}}$ approaches zero. In the following experiments parameter values are randomly generated from ranges that result in suitable behaviour, but they are not actively optimised, as the system-level behaviour does not change significantly.

7.4 Fully-connected market

In the first experiment I compare the behaviour of utilisation-based allocation against competitive bidding. A control case where bidders bid random values is used as a reference. The experiment measures the energy levels in the system over time for the three different allocation approaches and relates it to the number of tasks allocated.

To remove possible topological effects, the allocation takes place on a fully-connected network. This forms a single marketplace, where all agents can be equally involved in bidding, thus maximising the competition. As a result of the fully-connected topology, all agents have equal access to all the information in the system. We therefore expect allocation to be better (globally more efficient) than it will be in a distributed market.

The differences in bid calculation mean that the agents cannot be compared by pitting them directly against each other in a single market: the utilisation-based agents are not competitive at all, and will easily be under-bid by the competitive agents. I therefore investigate their performance by using separate markets, each consisting of only one type of agent.

7.4.1 Setup

The system consists of 50 agents in total: 10 individuals are drawn from five different skill types. Tasks are uploaded to the system from a ground station at a rate of 5 per 100 time steps for the first part of the experiment, representing a lightly loaded system. At $t = 10\,000$, after 100 uploads, the workload is doubled to 10 tasks per 100 time steps. Tasks consist of 5 components that are sequentially executed, as shown in Table 7.1. The run terminates at $t = 20\,000$. All agents have the same task execution cost (c_{int}) of 1 energy unit per task. Negotiation packets (c_{tx}) cost 0.001 units and task transferral (c_{tf}) packets 0.1 units.

The incoming energy (E_{inc}) is set to 0.01 per agent per time step, which is sufficient for the lightly loaded part of the experiment, but not enough for the overload condition. Agents can store up to 10 units of energy, and they start the experiment fully charged. All agents can communicate directly with all other agents, as the communication network is fully connected. The commission and packet time-to-live values therefore do not have any effect.

The competitive agents bid as described in Section 7.3, using a randomly generated set of values for their parameters. The initial values for μ and $\hat{c}_{\text{os}}$ were chosen uniformly at random from $[0, 10]$, while Δ_μ was selected from $[0, 0.2]$. The values for Δ_{os} and γ_{os} were randomly selected from $[0, 1]$ and n_{os} ranged from 15 to 25. These values result in the bidding behaviour seen in Figure 7.2. The utilisation-based agents use their energy levels and expected outsourcing cost to compute bids, as described in previous chapters. The random bidders respond to an auction announcement with a value picked uniformly at random from the range $[c_{\text{int}}, 10]$. They do not calculate outsourcing estimates, nor do they take their energy levels into account beyond checking that they have sufficient energy for task execution. Note that the different types of agents

Table 7.1: Compound task structure used in competitive allocation experiment. The execution of task elements (left-hand column) results in another task element (right-hand column) that must be executed. All components must be executed before the task is complete.

$A \rightarrow$	aB
$B \rightarrow$	bC
$C \rightarrow$	cD
$D \rightarrow$	dE
$E \rightarrow$	e

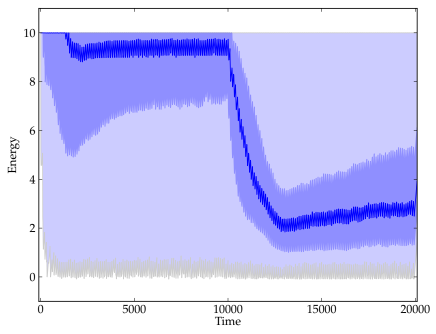
never compete against each other: every type interacts only with a community of the same type.

7.4.2 Energy measurements

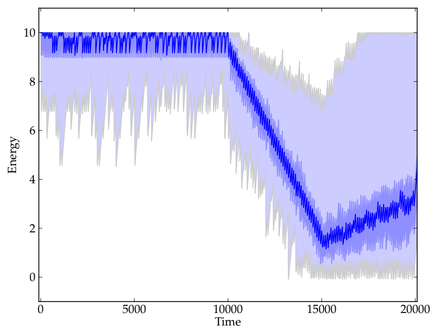
The energy levels of all agents are captured over the duration of a run, and repeated for 50 runs. Figure 7.3 shows a percentile plot of the cumulative results for competitive, utilisation-based and random bidding. The minimum to maximum energy range over all runs is given by the lightly-shaded band, while the darker zone ranges from the 25th to 75th percentiles. The median energy is indicated by the solid blue line.

The graphs display four distinct phases over the course of the experiment. At the start of a run there is a transient initial phase, until allocation settles into the steady-state pattern for a light task load. When the task load is doubled at $t = 10\,000$, another large transient is observed, before the system settles into a new steady state with substantially lower energy. The initial transient is most visible for the competitive market in Figure 7.3a, peaking at $t = 2\,000$. This response reflects the period during which the competitive agents converge on the market price. From $t = 2\,000$ to $t = 10\,000$ we see the 25th percentile recover while the median stays the same. This period corresponds to the zero-profit state in the market. The increase in workload at $t = 10\,000$ causes the median energy in the market to decrease, along with the 25th to 75th percentile band. The decrease is due to the workload exceeding the incoming energy in the system, thus forcing agents to use their stored energy to complete tasks. The decrease stops around $t = 13\,000$ when the stored energy is depleted, before the network settles into the new steady-state condition, where some tasks cannot be completed, due to insufficient energy.

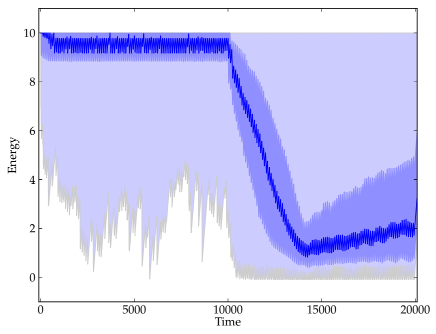
In this overloaded phase, the type A agents from Table 7.1 are always exhausted, and sometimes incapable of executing a task. The subsequent task



(a) Competitive



(b) Utilisation-based



(c) Random

Figure 7.3: Energy over time using three different bidding approaches. The task load is doubled at $t = 10\,000$. The solid line shows the median energy of fifty agents over 50 runs. The lightly shaded area indicates the minimum and maximum range of agents' energy, while the darker zone shows the range from the 25th to 75th percentile. Note the narrow energy range for utilisation-based bidding in comparison to competitive bidding.

components are therefore not executed, thereby decreasing the load on the agents responsible for task components B to E. As a result, the energy of these agents will increase, as can be seen in the gradual upwards curve of the median energy. Experiments where task allocation was run for longer showed the median energy settling at around 4 units, with a minimum of 0 and a maximum of 10, for all three cases. When the system is overloaded, tasks are allocated more on node availability than the value of their bids, because there is so little network capacity available. As a result, the behaviour of the three allocation approaches looks rather similar for $t > 15\,000$.

A qualitative comparison of the graphs for different allocation approaches reveal a number of obvious differences. Broadly speaking, we observe a significant difference between competitive and utilisation-based task allocation, while the random market is somewhere in between.

Most significantly, the median agent energy when using utilisation-based allocation is always higher than for competitive allocation, confirming the hypothesis that the utilisation-based approach would be more efficient. The behaviour of the 25th to 75th percentile band hints at how the improvement is achieved. For utilisation-based allocation, the system acts to equalise the energies of agents. The relative similarity in node energies shows in the narrowness of the 25th to 75th percentile band. For the competitive bidders, this band is much wider, because node energy does not directly influence bidding. When using random bidders, the width of the band is between that of other two cases. With agents choosing bid values from the same range, allocation will be uniformly distributed between agents, in effect approximating a round-robin allocation. It should therefore come as no surprise to see energy dynamics which are somewhat similar to the utilisation-based approach.

The minimum-maximum ranges of the different approaches reinforce what the 25th to 75th percentile band shows. In the competitive market we can see that for almost the entire duration of the experiment, at least one node has no energy, while another is fully charged. For utilisation-based allocation, the minimum energy hovers around 7 units for the lightly loaded part of the experiment, only decreasing significantly when the workload doubles. However, at the same time we also see the maximum energy decrease, from 10 to 7.5 at $t = 15\,000$. Random bidding results in a minimum node energy around 2 for the first half of the experiment, decreasing to 0 when the task load increases. The maximum energy, however, stays at 10, suggesting that the energy distribution is not as equal as with utilisation-based allocation.

Another notable feature is the overshoot visible on the 25th percentile of the initial transient of the competitive market; the other approaches do not display this phenomenon. The overshoot is due to a combination of agent competition and their learning of the network price. Initially, at $t < 1\,000$, only a few agents have succeeded in winning a job. They then use the knowledge of the market price to win more jobs, thus reducing their own energy even further causing the dramatic decrease in both minimum and 25th percentile values. Over time, other agents eventually lower their bids enough to also win tasks, causing work to be distributed more evenly. This can be seen in the recovery of the 25th percentile from $t = 2\,000$ to $t = 5\,000$.

Finally, note that the rate of decrease in the median energy (i.e., the slope of the transient at $t = 10\,000$) is the steepest for the competitive market, followed by the random market, while the magnitude of the slope in the utilisation-based case is smaller still. The difference can most clearly be seen in the steepness of the slope of the energy median in Figure 7.3a when compared to Figure 7.3b. For a fixed task load and incoming energy level (E_{inc}), the slope is determined by the number of agents that can store more energy. The narrow energy distribution in the utilisation-based system means that a greater number of agents have spare capacity to store more energy. For the competitive system, however, some agents are almost empty while others are full: as a result the system cannot store as much of the incoming energy, causing a significantly steeper slope. The random market is, once again, between these two extremes.

7.4.3 Tasks allocated

If we look at the number of tasks allocated, we find that the utilisation-based network outperforms the other two approaches with a mean of 1 419.16 tasks per run (standard error = 0.22) out of a possible 1 500. The random bidder is next with 1 401.62 tasks (standard error = 0.64); followed by the competitive market with 1 371.84 (standard error = 0.65). The small standard error values can be ascribed to this experiment using a fully connected network, i.e., there are no topological effects that influence task allocation success.

If we investigate where performance diverged for the different cases, we find that their performances in the steady-state regimes are the same. However, in the second transient, where the networks move from having sufficient energy for task allocation to not having enough, the competitive approach depletes its stored energy first. This is partly due to it having a slightly lower median energy at the start of the transient, but primarily because its energy decreases much faster than either the utilisation-based market or the one with random bidders. From this we can deduce that, although the performance differences are small for this scenario, for a highly dynamic workload the utilisation-based allocation approach will do significantly better than the competitive one.

7.5 Spatially distributed markets

In this section I investigate what happens when we move from a fully-connected to a spatially distributed market where network topology plays a significant role. While the previous section provides a useful reference for understanding the dynamics of the system, distributed markets are directly relevant to distributed satellite systems as the high cost of communication makes fully-connected markets undesirable.

To determine the effect of topology I measure the number of tasks allocated for a range of network topologies with different degrees of connectivity. By keeping the workload constant and varying the incoming energy, the response of the system to different levels of utilisation can be measured. A utilisation-based market is simulated under the same conditions to serve as a comparison and allow identification of the part of parameter space where one approach would be preferred to the other.

7.5.1 Setup

This experiment uses a very similar configuration as that of the previous section: 50 agents consisting of 10 individuals for each one of 5 task component types. Tasks arrive at a ground station at a constant rate of 5 per 100 time steps, using the same task structure as in Table 7.1, resulting in 25 task components per 100 time steps. All task components cost 1 energy unit to execute. Agents can store up to 10 units of energy, and they start the experiment fully charged. For every competitive agent, the parameter values were randomly generated using the same ranges as in the previous section.

The agents are distributed uniformly at random in a three-dimensional cube with sides of length one. If two agents are closer than the communication range (R_{th}), they are connected. This forms a random geometric network similar to the topologies used in the previous chapter. The three-dimensional position of the agents does not change throughout the simulation to avoid effects due to a dynamic topology. A network that consists of a single connected component is used in all cases.

By varying the communication distance the network changes from barely connected to almost fully connected, allowing us to measure the performance across a range of network connectivity values. A minimum R_{th} of 0.2 results in networks with a mean diameter of 9 and mean degree of 5.09, while the maximum R_{th} gives a mean diameter of 2.96, and a mean degree of 30.23. The packet time-to-live distance (d_{ttl}) is fixed at 4 for all network topologies, while the commission value is fixed at 0.1 per message repeat. Negotiation packets cost 0.001 while task transfer packets cost 0.1, i.e., $\alpha = 100$.

The incoming energy (E_{inc}) ranges from 0.001 to 0.01 per agent per time step. Note that increasing E_{inc} is analogous to decreasing the workload as more energy becomes available to complete the required tasks. Keeping the workload constant, however, simplifies the interpretation of the experimental results, as no additional scaling is required.

The experiment is repeated 50 times for all values of R_{th} and E_{inc}, each time with new agent positions and new agent parameter values. I measure the number of tasks successfully completed and the energy remaining in the system at the end of the simulation.

Based on the previous experiment, we expect to observe an area where the system is under-utilised, i.e., the incoming energy exceeds the workload, allowing all tasks to be completed. At the other extreme, we will observe an overloaded system, where there simply is not enough energy available to satisfy all task requests. We are interested in where the transition between these zones occurs, and how it is influenced by the communication network topology. We can calculate the expected transition point for an idealised system, without topology or communication cost, by setting the energy used per time step equal to the maximum incoming energy per time step, and solving for E_{inc}:

$$\begin{aligned}
(\text{tasks}) \times (\text{task components}) &\leq (\text{incoming energy}) \times (\text{agents}) \\
\frac{5}{100} \times 5 &\leq E_{\text{inc}} \times 50 \\
\Rightarrow E_{\text{inc}} &= 0.005
\end{aligned}$$

For an ideal system, we therefore expect to see the transition between all tasks being allocated and allocation failing around $E_{\text{inc}} = 0.005$. A more realistic model that takes into account the energy spent on communication in a network should therefore show a transition point at a slightly higher value of E_{inc}.

7.5.2 Results

We start by keeping the amount of incoming energy fixed, and focusing on the effect of changing the connectedness of the market. The difference in the number of tasks allocated for competitive and utilisation-based bidding is shown in Figure 7.4. The incoming energy (E_{inc}) was kept at 0.006, while the allocation success was measured for a number of increasingly connected networks. As expected, the number of tasks allocated by both approaches increases as the network becomes more connected. For $R_{\text{th}} > 0.35$, both approaches succeed in allocating all tasks, but the focus of this experiment lies at the lower values of R_{th}. We see that both allocation approaches deteriorate when connectivity decreases, but that the competitive system fares significantly worse.

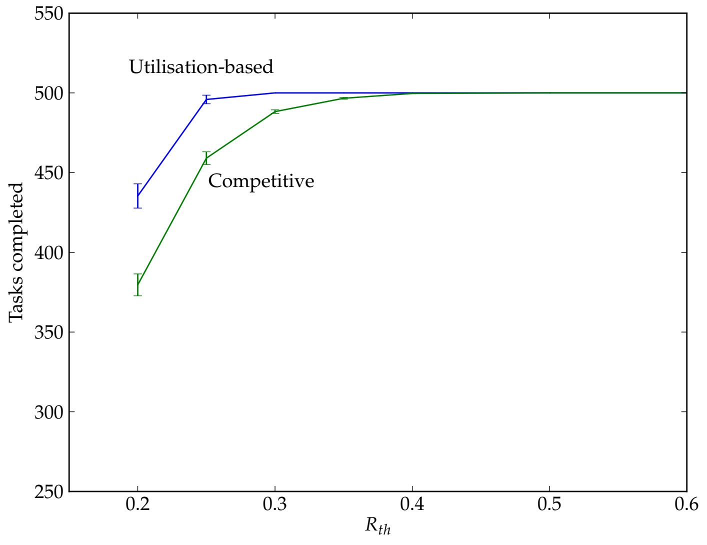
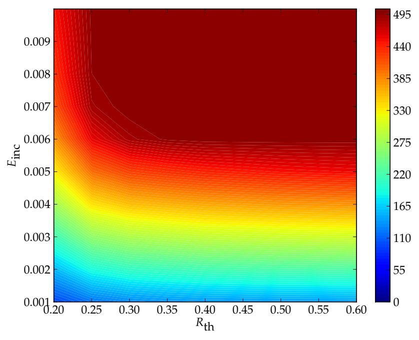


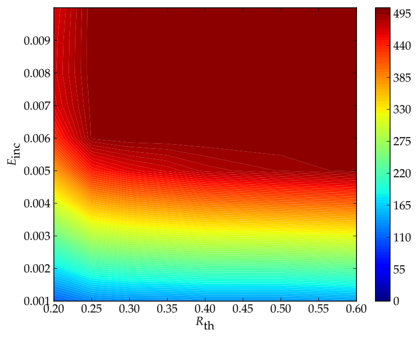
Figure 7.4: Comparison of task allocation using utilisation-based (blue) and competitive (green) task allocation. The connectivity of the network increases along the axis, from relatively sparse to almost fully connected. The y-axis shows the number of task completed. It is clear that using utilisation-based market results in more successful task allocation for less well-connected networks. The error bars indicate the standard error.

This represents but one point on the incoming energy axis — we gain a much better view of the larger landscape if we also consider a range of E_{inc} values, from 0.001 to 0.009. Figure 7.5a shows heat maps of the resulting task allocation; Figure 7.4 should be seen as horizontal section across these graphs at $E_{\text{inc}} = 0.006$. Note once again that a small E_{inc} value is analogous to a large workload, because the amount of work required from the system exceeds the available energy. Conversely, a large E_{inc} value represents a system with a light task load and enough energy to complete all tasks.

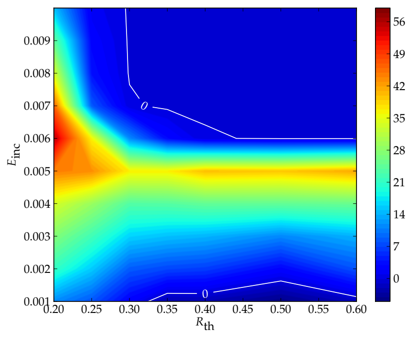
At first glance the task allocation results for both approaches look very similar: when E_{inc} is low, very few tasks are completed, and when it is high all tasks are completed. However, the performance at the transition between these



(a) Competitive bidding



(b) Utilisation-based bidding



(c) Difference

Figure 7.5: Number of tasks allocated for competitive (a) and utilisation-based bidding (b) over a range of communication distance (R_{th}) and incoming energy (E_{inc}) values. The difference between (a) and (b) is shown in (c), which highlights the superior performance of the utilisation-based approach for systems with limited energy and low connectivity.

zones differs, as does the behaviour when the network topology becomes less connected. The expected limit of perfect allocation for an idealised system lies at $E_{\text{inc}} = 0.005$, and we can observe that the performance of the competitive market starts to deteriorate at $E_{\text{inc}} = 0.006$. For the utilisation-based market it ranges from about 0.005 to 0.006, very close to the ideal-case performance.

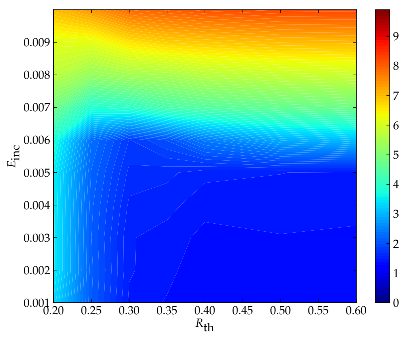
Figure 7.5c literally plots the difference between the two approaches: it shows number of tasks allocated using a utilisation-based market less the number tasks allocated using the competitive approach. This plot clearly demonstrates that utilisation-based allocation performs well in a larger part of the parameter space than the competitive market, because it allocates a larger number of tasks for lower energy and decreased connectivity.

If we look at the 0 contour, we see that the number of tasks allocated is the same for high energy, high connectivity scenarios. With decreasing energy and a relatively high connectivity ($R_{\text{th}} > 0.35$), there is a distinct change and a rapid increase in the difference, peaking at $E_{\text{inc}} = 0.005$. In this area the number of tasks allocated through competitive bidding starts to decrease, while the utilisation-based allocation still performs well. If we decrease the incoming energy even more, the difference between the two approaches also gradually decreases, reaching 0 for $E_{\text{inc}} = 0.001$. At very low values of E_{inc} the majority of the tasks completed successfully used energy that agents had at the start of the simulation.

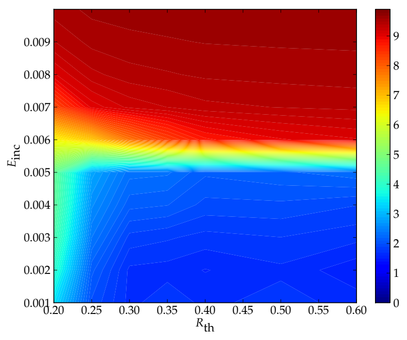
If we traverse the plot from highly connected to sparse, we start in a zone ($R_{\text{th}} > 0.35$) where the connectivity does not influence the performance of either allocation approach significantly. This indicates that the networks are connected enough that more connections do not improve it. For lower connectivity, however, performance does change: relatively slowly for utilisation-based allocation, and faster for the profit-based bidders. This can be ascribed to critical nodes, which provide bridges between different clusters in the network, being over-utilised causing network fragmentation.

By looking at the average energy levels at the end of the simulation, as shown in Figure 7.6, we can better understand the reasons behind the performance results. Competitive allocation shows a gradual decrease from $E_{\text{inc}} = 0.01$ to $E_{\text{inc}} = 0.006$, which corresponds to the area where the system manages to allocate all tasks.

There is a slight increase in the amount of energy remaining in the system as R_{th} increases, due to the lower task transferral costs associated with a better connected network. In comparison, the utilisation-based allocation shows a plateau of high energy which drops rapidly from $E_{\text{inc}} = 0.006$ to $E_{\text{inc}} = 0.005$.



(a) Competitive task allocation



(b) Cooperative task allocation

Figure 7.6: The mean energy levels after allocating 500 tasks for a range of incoming energy (E_{inc}) and communication distance (R_{th}) values. Competitive bidding shows a gradual decrease as E_{inc} decreases, reaching a minimum around $E_{\mathrm{inc}} = 0.006$. When a cooperative market is used, the transition between having enough energy ($E_{\mathrm{inc}} > 0.005$) and not having enough is very sudden, due to the efficiency of this mechanism.

This transition demarcates the point where the incoming energy shifts from being sufficient for the workload, to being not enough. The abruptness of the change is again an indication of the efficiency of the utilisation-based approach — it maximises the total incoming energy to allow the largest number of tasks to be allocated.

In both allocation cases, the slightly higher energy at $R_{\mathrm{th}} = 0.2$ indicates that allocation failed, although enough energy was available. From this we can conclude that the decrease in task allocation in this zone can be directly ascribed to the sparse network topology.

7.6 Discussion

In this chapter I investigated the difference in performance between utilisation-based and profit-based task allocation. By adapting ZIP-traders to reverse, first-price, sealed-bid auctions, I could compare competitive agents to the cooperative ones developed in previous chapters. Simulation results demonstrate that the cooperative agents are always at least as good or better than the competitive ones in terms of the number of tasks allocated. While the allocation success is similar for well-connected markets with sufficient energy, the difference is especially noticeable when we move to less well-connected networks, or limit the energy available to agents in the system. This should come as no surprise: the incentives for the cooperative agents were defined to be aligned with a good global allocation, while the competitive agents were selfish and short-sighted. What is significant, however, are the reasons for the different behaviour, as it can tell us a lot about how we can use market-based control and why it works.

7.6.1 Competitive vs. cooperative markets

Based on these results, we can now return to the questions that motivated this chapter. Firstly, the engineering perspective: how does introducing a competitive market change the system performance? The above results show that the competitive market is comparable to cooperative allocation, as long as the market is well-connected and has sufficient energy. For less well-connected markets, or scenarios where the incoming energy is not significantly more than the work load, the performance of the competitive market decreases conspicuously faster than the cooperative market. The decrease in performance is primarily due to the inability of the competitive market to maximise the incoming energy, as can clearly be seen in the energy measurements. We can expect the difference in performance to be increased by a dynamic task load, which will make the cooperative approach superior in a larger part of the parameter space. A more sophisticated utility function for the competitive agents may improve their performance, but it will require some predictive ability to anticipate future workloads.

One might ask: is this part of parameter space significant? For distributed satellite systems and wireless sensor networks, we find ourselves faced with lower connectivity and critical energy — exactly the part of parameter space where cooperative allocation performs better. I would therefore argue that, where possible, cooperative allocation mechanisms are preferred to competitive markets in distributed satellite systems. However, it should also be noted that many applications do not fall into this part of parameter space, and can therefore be adequately addressed by either approach.

The key characteristic of the cooperative market that allows it to perform better is its tendency to distribute tasks across the network in response to the utilisation of individual agents. This can happen because of the information content in the bid values: high bids indicate agents that are undesirable from a global perspective. In contrast, the information content in the bids of competitive agents is relatively low, because they are bidding to win, not to communicate their state.

Fundamentally, the difference in performance is due to the different feedback loops associated with the allocation approaches. When a competitive bidder wins a job, its information about the system increases relative to the other agents because it learns the market price, which the others do not know. Knowing the market price increases the likelihood that our focal agent will win future tasks, because it can place more accurate bids. As a result of the positive feedback loop, the energy of the winning agents will rapidly be depleted. For a fully-connected network, the primary effect will be that the system-level energy increase is suboptimal. However, for a more distributed communication topology the exhaustion of connecting nodes is also significant, as it leads to network fragmentation. This explains the deterioration observed for low-connectivity networks shown in Figure 7.5.

In contrast, allocation in the utilisation-based market is subject to a clearly defined negative feedback loop: winning a task decreases an agent's energy. The decreased energy means that the agent's next bid will be more expensive, making it less likely to win a subsequent task. As a result, tasks are distributed in such a way as to equalise the energy levels across the system. This can be seen in the narrow energy distribution in Figure 7.3. Furthermore, because the utilisation-based market distributes tasks between the available agents, the number of tasks completed by an agent is related to the state of the network. This forms another informational feedback loop, linking the behaviour of the agent to the system.

7.6.2 Implications for market-based control

What are the implications for us, as users of market-based control? For controlling technical systems, if we have a choice between a cooperative and competitive market, I would recommend using the cooperative configuration. Competitive markets contain no magic: instead they are noisy, subject to positive feedback, and generally hard to control. We are however inundated with good news about competitive markets and their seemingly mystical properties to achieve good allocations. Optimistically believing that all the properties of real markets will save your system is similar to the naive view that evolutionary algorithms are anything more that a special case of stochastic search. While competitive markets certainly have promising features, ideological motivations for their use should be avoided: the system-level behaviour will only ever be as good as we design it to be.

However, sometimes the use of a competitive market is warranted: markets can determine the value of goods or services, offer open-ended incentives to agents in the market, or the system designer might not be in a position to specify all the behaviour. In these cases we should proceed with caution, while paying special attention to the feedback loops in the system. A combination of regulations and incentive engineering approaches, such as mechanism design, have to be used to manage the system. Broadly speaking, we need ways of coupling the utilities of individual agents to the system-level performance.

We should be careful to distinguish between the different features of market-based systems; we can here identify two distinct components relevant to control problems. The first component is the information flow in the system: what is revealed and how it is communicated. This component was identical for the competitive and cooperative markets in this chapter. The second component is the incentive structure used by agents: this determines what decisions they make. Here the two markets differed, with a corresponding difference in the global performance.

Market-like mechanisms are very good at the first component, as they provide highly efficient means to disseminate information for decision making. The amount of information can vary, from the information-poor auctions used above, to continuous double auctions that contain considerably more information. I believe this is one of the most attractive features of using markets for control.

The incentive component uses the information flow component to determine the system-level behaviour. Are agents trying to optimise their own profit, or do they have a system-level goal as a target? For competitive systems, we have seen that this can result in stable equilibria, but that the system's performance would probably have been better had a cooperative mechanism been used. In my opinion, a large number of distributed control problems can be adequately addressed without relying on competitive incentives.

7.6.3 Final considerations

The models of agents and markets used in this chapter have several limitations, some of which point the way to questions for further exploration.

The number of agents in the system and their skills were fixed throughout the experiments. In real-world markets a new agent, with a new skill, can fill a niche at any time if a need is not met sufficiently. This provides an adaptation mechanism that would make the market more flexible than the one described above, and may improve performance.

The cost of task execution (c_{int}) was the same for all agents in the experiments. While this is a reasonable assumption for a man-made system, it does not apply to real-world markets. A variance in the cost would be reflected in the bid values, and it could therefore influence the task distribution. The parameters used by the adaptive agents were not optimised to find the "best" values for the learning algorithm. It would be particularly interesting to see if the optimal parameter values are determined by the other individuals in the market, or whether some depend on the structure of the market.

My usage of a market on a network made me aware of how little research has been done on economies on networks, where communication between agents is determined by the topology. Apart from some work on graph-based economies (e.g., Kakade *et al.*, 2004; Judd and Kearns, 2008; Goyal, 2009) and a few publications on spatial economies (e.g. Fujita *et al.*, 2001; Ladley and Bullock, 2005), the field is strangely empty if we consider the importance of trade links and distribution networks. If information flow in the economy is determined by the topology, the economy effectively consists of a series of overlapping markets. How do prices propagate through this system, what are the implications for convergence, and how do agents cope with the decreased information available to them? Further investigation on spatially distributed auctions and auctions with limited communication is required.

On the market-based control front, it would be interesting to explore the trade-off in increasing the amount of information in a market against the improvement in system performance. In my work I focused on a minimal information scenario, but it is possible that higher information markets exist where the cost of the information is outweighed by the improvement in system performance. An increase in information would also allow for more sophisticated agent strategies, which will in turn modify the performance curves observed above. Finally, in work that is closely related to the evolution of cooperation, the space between the competitive and cooperative markets can be explored. Is it possible to have stable cooperative or hybrid strategies or will markets, when given the opportunity, always tend towards a competitive state?

8 Conclusions

This thesis started with the question of how to best manage a group of interdependent satellites that are constrained in their energy and communication abilities. Instead of focusing on the details of satellite design, however, I approached it as a multi-agent task allocation problem: a problem that can be addressed by using a market metaphor. I quickly came to the conclusion that much of the current work in task allocation in multi-agent systems can be divided into two distinct groups, one general and one specific, with very little work providing a bridge between these extremes. The design process can be viewed as a traversal of this problem space, progressively narrowing the scope of the model we are working with as we move from the general to the specific.

At the highest level, the primary objectives of this thesis can therefore be defined as:

- Development of a task allocation mechanism that takes the constraints of the distributed satellite system environment into account.
- Mapping a section of the problem space of task allocation in multi-agent systems through the above design process.

The first of these involves an engineering problem, and provides a tool which can be used to design and manage multi-satellite systems. The second objective is concerned with the broader space of task allocation, and tries to provide some structure to the segment of the problem space relating to multi-satellite systems and similar applications.

The first objective is met by the market-based task allocation mechanism I developed and verified over the course of this thesis. I not only present the allocation mechanism, but also describe where it should be used, and perhaps more importantly, where it should not be used. As part of the development process, my experiments led to a number of guidelines that should help designers of distributed systems cover the gap between the abstract and applied ends of the spectrum of multi-agent systems, thereby addressing the second objective.

8.1 Overview

I started by proposing a task allocation mechanism that takes multi-component tasks, dynamic topologies, and expensive communication into account (Chapter 3). This mechanism was developed from a reference mission scenario which allowed the identification of constraints and requirements. By applying a model of a human labour market to the abstracted system, an allocation mechanism was derived. The mechanism was then mapped back to the distributed satellite application to define the implementation details of the task allocation mechanism in the rest of the thesis. The mechanism defines the routing and communication layer required to successfully allocate tasks. Agents calculate their bids by taking their own utilisation and skills into account. By adding a commission percentage to bid values, topological information about the state of the network is included in the prices. Routing information is generated on an ad hoc basis as part of the auction. Winner calculation is distributed across the network to minimise communication processing requirements.

In Chapter 4 I verified the suitability of the allocation mechanism by simulating it in a few basic scenarios. The simplicity of these test cases also helped us to develop a better understanding of the system-level behaviour of the mechanism.

The next three chapters broadly examined at where the market-based task allocation mechanism should be used by focusing on the design decisions and constraints related to specific parameters. The behaviour of the allocation mechanism was explored in more detail in Chapter 5. An analytical description of the allocation mechanism was developed to show that the system-level communication cost is constant with respect to the number of nodes in the system, while it grows linearly with the size of the auction community. Simulation was used to compare the allocation mechanism to a reference implementation using a centralised allocator in the context of node failures. The robustness of the mechanism can be adjusted by changing the size of the auction community: the market-based approach performs significantly better in terms of the number of tasks allocated and energy used for large systems where nodes can fail. In smaller systems, or where the nodes and links are sufficiently reliable, an accurate model can be maintained by a centralised controller, thus making it a better solution for that section of parameter space.

Chapter 6 investigated the interplay between communication cost and network topology. This was done by measuring the system-level cost associated with different topologies: results show the potential detrimental impact of having an unnecessarily large auction community. If the transmission power required to create better-connected networks is taken into account, we observe a clear zone of good allocation, bounded on the lower end by nodes' connectedness, and on the upper end by the increasing energy cost associated with a well-connected network. A mobility model that describes the Keplerian trajectories of a group of satellites was developed to measure the effect of realistic orbital mechanics. Due to the short time scales at which auctions happen, the allocation mechanism functions at least as well for the dynamic case as when using a static network.

Finally, in Chapter 7, I investigated the difference between the established cooperative task allocation mechanism and a competitive market. This involved the development of a competitive adaptive agent related to ZIP-traders, but modified to single-sided auctions with outsourced, multi-component tasks. The dynamics of these competitive agents show strong positive feedback in task allocation: nodes that win some jobs tend to win more. This leads to an uneven spreading of tasks and suboptimal system-level energy consumption. When compared to the cooperative agents on a range of network topologies, the cooperative solution outperforms the competitive one in networks with lower connectivity and limited energy. This can be ascribed to the increased efficiency of the cooperative markets because the task distribution is better from a global perspective.

8.2 Traversing the problem space

The preceding chapters dealt with the motivation and design of a market-based task allocation mechanism, followed by the exploration of significant parameters. The focus on the specifics of the application, however, makes it easy to lose sight of the larger problem space. In this section I therefore review the path followed from a general model of task allocation, to the specifics of the distributed satellite system application, with the objective of identifying related systems.

In Chapter 3, I started with a high-level model of labour market. The use of a market mechanism is based on economical theory about the suitability of markets for distributed coordination problems. The use of a price mechanism abstracts information, and allows efficient communication between agents. It should be noted that at this point, the model could be applied to a range of allocation problems, not just distributed satellite systems. By applying system characteristics, either as design decisions or constraints, the general model is pruned to the specifics of the application, as illustrated in Figure 1.2. However, these parameters also provide links to related systems, which allow us to situate distributed satellite systems in a larger problem space.

Communication cost is the most prominent parameter I explored. Distributed satellite systems have high communication costs, but communication is still possible: communication should therefore be minimised. If communication had a negligible impact on performance, factors such as topology and system volatility could easily be detected and compensated for, assuming sufficient computational power was available. System management is thus simplified by the availability of up-to-date state information. One example is networked computer systems with fast, reliable communication between nodes. On the other hand, if communication was very expensive, it would be better to maintain a model of the system instead. However, this assumes that a sufficiently accurate model can be built, and that modelling it is computationally feasible. For systems with a significant level of unpredictable noise (e.g., node failures or a dynamic topology), modelling is simply not practical, and some form of state feedback is required. If the labour market model is applied to a system with inexpensive communication, it moves much closer to work on the interaction of agents in a single marketplace: more information is available, a greater number of agents are involved and multiple bidding rounds become feasible. This is significantly different from my allocation model, yet closely related through the communication cost parameter.

In terms of scale, my allocation mechanism is targeted at large systems. As discussed in Chapter 5, smaller systems can usually be better addressed using traditional approaches such as centralised control. In my opinion, many distributed robotics applications can safely be described as "small", and are therefore best not controlled using market-based or emergent mechanisms — factors such as validation and verification makes centralised approaches particularly attractive. For large systems such as wireless sensor networks, however, distributed approaches can offer the required level of performance, but often without formal guarantees on quality. The boundaries between large and small is often not clear, because other parameters, such as communication cost or volatility, contribute to determine how manageable a system is.

If a network is very stable, a static model can usually be constructed to assist in navigating or managing it. However, network volatility requires a mechanism to detect and adjust to changes, whether it is due to failures or to mobile nodes. My use of an auction provided this network information, other approaches such as intermittently polling nodes for their status fulfil the same function. If access to the system is fast enough, and only partial information is required, as in the case of distributed satellite systems, ad hoc measurement should be sufficient. Slower access and limitations on observability will instead require more sophisticated models — examples include telephone directories and road maps.

Topology plays a role when communication has a significant cost. A system with higher connectivity is more robust to changes in the network, but frequently also requires increased computation on the part of nodes to deal with the increased information. Less well-connected systems are more sensitive to changes: topology therefore plays a more significant role in overall system performance, as illustrated in Chapter 6. Wireless sensor networks are similarly sensitive to topology, as it determines both global performance and how much information is available to manage the system. Electricity distribution grids are also sensitive to changes in topology, but note that their information flow is not dependent on the network structure.

The final parameter I considered was the difference in cooperative and competitive markets. While the former relates the desired global behaviour directly to local utility functions, it requires total control of the system. If we move towards competitive scenarios, such as necessitated by multi-stake holder systems, systems become harder to control and require additional mechanisms to elicit the correct behaviour. These types of systems are naturally much more closely related to real-world social systems.

The above systems all share an axis in parameter space with distributed satellite systems. By applying the discussed constraints to my generic model, I arrived at a specific application. However, the proximity of the other systems is also important: only by mapping out the areas surrounding our design trajectories can we gain the necessary understanding how different systems are related. A thorough map of this space is required to allow us to progress to constructing methodologies for designing multi-agent systems.

8.3 Contributions and implications

This thesis has led to the following contributions:

- For the satellite engineer community, I have demonstrated the suitability of a market-based task allocation mechanism for managing distributed satellite systems. This represents a point on the specific end of the design space. The allocation mechanism is fully distributed, energy efficient, scalable, and robust to node failure and changes in topology. A novel commission parameter was devised in order to take expensive communication into account, while recursive outsourcing is used to address compound tasks. My verification efforts identified the relationships between design constraints and system parameters in order to show the part of the problem space where the allocation mechanism should be used. To allow system designers to navigate through the multiple interdependent parameters, I have provided a map of their interaction in Figure 8.1.

- In terms of market-based control, my comparison of cooperative and competitive markets demonstrated the cost of selfish agents. Although in some parts of parameter space we observe similar performance between the two methods, selfish bidding is prone to suboptimal task allocation due to the positive feedback loops inherent in the market. In contrast, the utilisation-based approach uses negative feedback to distribute tasks evenly, thereby improving the global system performance.

- For those interested in task allocation in distributed systems in general, this thesis provides an analysis of the parameter space in which task allocation mechanisms are embedded. By exploring the effects of network size, robustness, communication parameters, and topology, the area in which the proposed mechanism should be applicable has been identified. This extends beyond the satellite problem to related systems ranging from mobile robotics to wireless sensor networks — this can best be interpreted as a large-scale map of the design space surrounding these systems. A clear message throughout this thesis is that the best management approach is highly dependent on the parameters of the system at hand: a number of different approaches all have a role to play.

- I also argue that there is a gulf between our high-level task allocation knowledge and the actual mapping of it to specific problems. Our domain-specific knowledge is generally tightly coupled to a specific application — as a field, we need a bridge to span the gap. Responsible engineering allows us to use the design process to address this problem, as I demonstrated in this thesis, but persistent effort is required if we want to reach the point where we have a principled methodology for controlling these systems.

8.4 Limitations

As with any work of this nature, a number of limitations exist. Most of these have been pointed out previously, but a brief review helps to define the space

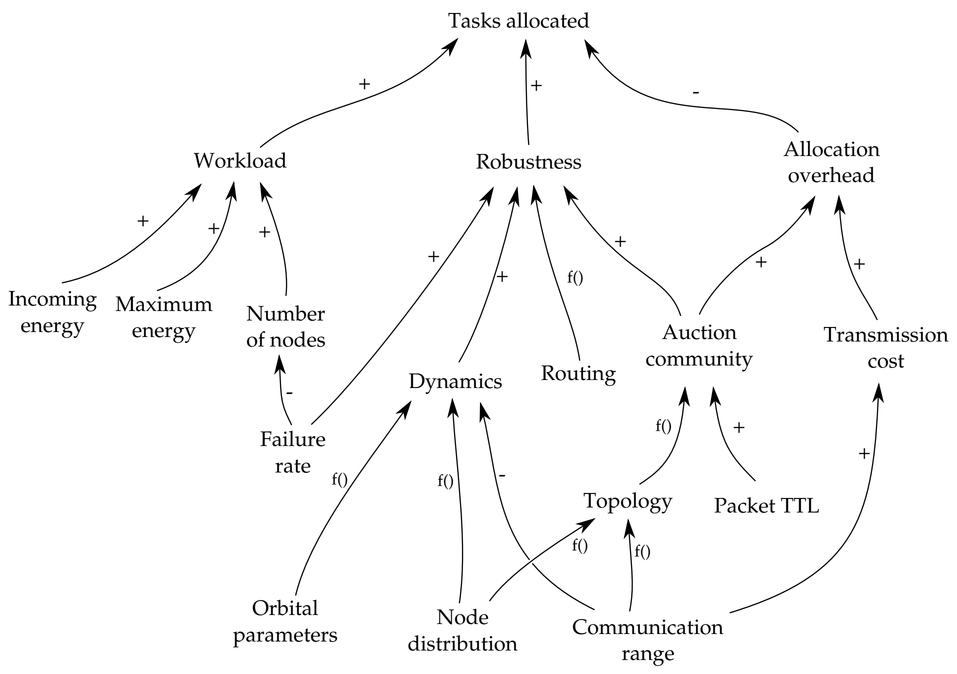


Figure 8.1: Map of how different parameters in the market-based task allocation mechanism interact to determine the number of tasks allocated. The directions of the arrows indicate how the different parameters relate to each other. A positive relationship is marked with a plus sign, while a minus sign indicates a negative relationship. For some parameters the influence is a more complicated function, denoted by f().

where my contributions are relevant, and the areas where this is not necessarily the case.

- A specific type of task allocation was considered: task elements are allocated to individuals. Alternative mission scenarios may require different task structures, for example that all satellites execute a certain function at a specific moment. While many of these scenarios can be realised with the current system, it is possible that other protocols may provide higher efficiency in these cases.

- It was assumed that the execution costs for task elements can be accurately specified. In reality, a significant amount of modelling and measurement is required to determine these values; however, I suspect it is possible to characterise these values for specific systems, especially if a task element model is used.

- In building the abstracted model in Chapter 3, I dealt primarily with small, heterogeneous spacecraft. If larger spacecraft are considered, specifically ones where the available power is less of a constraint, we move to a different position in problem space. From my exploration of the space I believe that the market-based mechanism proposed here will still work for this new case, but I suspect optimisations exist that can better exploit the parameters of the new target system.

- A simple wireless communication model was used for packet transmission. Communication was considered to be deterministic and symmetric, with no accounting for interference, bandwidth, signal-to-noise ratios or asymmetric links. Although this model was sufficient for modelling the network dynamics, much refinement is needed before accurate predictions on the absolute cost of communication can be made.

- In my investigation of competitive markets, I only addressed reactive agents. This does not encompass the entire scope of possible agent strategies.

- Finally, the design process only traversed a relatively small segment of the entire design space, thereby only relating a few problems by means of a limited parameter set. This process needs to be repeated with many different designs to obtain a better view of the family of distributed control problems, and build a bridge that links them with our abstract models.

8.5 Future work

Although my research answered some questions, it also made me aware of a number of ways in which it can still be extended. In keeping with the primary objectives of the thesis, the future work can be divided into avenues that relate to the engineering of multi-satellite systems, and into work that deals with task allocation in general.

On the spacecraft engineering side, an increase in simulation detail and eventual implementation on a hardware platform is required as the next steps in the verification of the task allocation mechanism. This will also allow the limitations around communication cost and task execution cost, as outlined above, to be better understood. From a network management perspective, mechanisms that verify the execution of allocated tasks are also required before the system can be deployed. Adaptive protocols that optimistically conserve energy by limiting communication distance should also be investigated, as they can provide a significant improvement in the performance of the system. All of the above will naturally also impact upon mission design processes, most of which are currently largely untested in this arena.

For readers primarily interested in task allocation, I believe that further investigation of spatially distributed auctions and negotiation with costly communication can lead to valuable contributions. In current literature most auctions are assumed to be localised to a mostly common space, with the associated theory developed for this case. In comparison, both the intelligent agents and mechanism design communities have paid relatively little attention to effective strategies and system dynamics for the case where nodes are distributed across a network. The effects of costly communication in negotiation scenarios have similarly not been adequately addressed. As demonstrated in this thesis, expensive communication changes the utility of agents in the system; sometimes even for those agents not actively bidding or allocating tasks. A formal analysis of how communication cost should influence decision making will be a significant contribution to the field. The belief that similar environments will result in similar allocation mechanisms is a core assumption underlying my work, but it raises the question of how environmental parameters determine which type of market or other allocation system is best suited to a specific allocation problem.

At a more abstract level, I believe further work is also required by all parties to improve the state of the field of task allocation. As argued in Chapter 2, a large gap currently exists between our abstract ideas of task allocation and actual implementations, leading me to believe that we still do not know how to efficiently map from our high-level knowledge to a specific solution. In the work presented here I have contributed by relating some points in the problem space, but to overcome this problem everyone in the field needs to contribute.

Why does this gap exist? In my mind it is to some extent symptomatic of the dynamics of the system in which we research task allocation. Those dealing with the engineering of applications are pushed to deliver working systems, as it is the metric against which they are measured. A compounding factor the large number of fields in which task (and resource) allocation is encountered. Allocation is therefore seen as part of a different problem, not a problem in itself. Those contributing to the general theories frequently come from more abstract fields, which can make it hard to relate their work to applications, especially the problems encountered in realistic settings. Individuals tend to stay in their respective fields — they are required to be field specialists, not task allocation specialists.

However, we cannot be absolved of all blame; we are still rational, independent beings. I believe an important contribution can be made through responsible engineering, where the focus is not only on delivering a product, but also learning about the space in which the problem is solved along the way, as I demonstrated in this thesis. The dissemination of this knowledge is crucial. Similarly, I believe greater emphasis is required on the commonality of task allocation across number of fields. A broad perspective is required; blind allegiance to approaches that are either long established or currently fashionable does not result in fair comparisons, nor does it increase our knowledge of the field.

This thesis applied the ideas initially developed by Adam Smith, and extended by others, to manage our increasingly complex technological systems. However, work like my own exists in parallel with another literature in which those same ideas are combined with computational models to further our understanding of existing social systems. It seems both likely and desirable that the two fields will benefit from greater cross-fertilisation in the years to come.

Bibliography

Anderson, C. (2001). The complexity and hierarchical structure of tasks in insect societies. *Animal Behaviour*, 62(4):643–651.

Anderson, P. W. (1972). More is different. *Science*, 177(4047):393–396.

Axelrod, R. (2006). *Agent-based Modeling as a Bridge Between Disciplines*, volume 2, chapter 33, pages 1565–1584.

Axelrod, R. and Hamilton, W. D. (1981). The evolution of cooperation. *Science*, 211(4489):1390–1396.

Axtell, R. (1999). The emergence of firms in a population of agents. Working Papers 99-03-019, Santa Fe Institute.

Ayre, M., Pettazzi, L., and Izzo, D. (2005). Self-assembly in space using behaviour-based intelligent components. Technical Report ACT (SASUB-BIC05), European Space Agency, the Advanced Concepts Team.

Babaioff, M., Nisan, N., and Pavlov, E. (2009). Mechanisms for a spatially distributed market. *Games and Economic Behavior*, 66(2):660–684.

Bagnall, A. and Toft, I. (2004). An agent model for first price and second price private value auctions artificial evolution. volume 2936 of *Lecture Notes in Computer Science*, chapter 23, pages 281–292. Springer Berlin / Heidelberg, Berlin, Heidelberg.

Bagnall, A. and Toft, I. (2006). Autonomous adaptive agents for single seller sealed bid auctions. *Autonomous Agents and Multi-Agent Systems*, 12(3):259–292.

Bai, F. and Helmy, A. (2004). A survey of mobility models in wireless adhoc networks. *Wireless Ad Hoc and Sensor Networks.*

Bandyopadhyay, S., Coyle, E. J., and Falck, T. (2007). Stochastic properties of mobility models in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 6(11):1218–1229.

Barnhart, D. J., Vladimirova, T., and Sweeting, M. N. (2006). Satellite-on-a-Chip development for future distributed space missions. *ASME Conference Proceedings*, 2006(42541):199–212.

Barnhart, D. J., Vladimirova, T., and Sweeting, M. N. (2007). Very-Small-satellite design for distributed space missions. *Journal of Spacecraft and Rockets*, 44(6):1294–1306.

Bedau, M. A. (1997). Weak emergence. *Noûs*, 31:375–399.

Begg, D. (2005). *Economics.* McGraw Hill, 8th edition.

Bekey, I. (2005). Phase I Study: Extremely large swarm array of picosats for microwave / RF earth sensing, radiometry and mapping. Technical report, NASA Institute of Advanced Concepts (NIAC).

Binmore, K. and Klemperer, P. (2002). The biggest auction ever: the sale of the British 3G telecom licences. *The Economic Journal*, 112(478):C74–C96.

Blumrosen, L. and Nisan, N. (2002). Auctions with severely bounded communication. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 406–415.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems.* Oxford University Press, Inc., New York, NY, USA.

Brandt, F. and Weiss, G. (2002). Antisocial agents and Vickrey auctions. In *Revised Papers from the 8th International Workshop on Intelligent Agents VIII*, ATAL '01, pages 335–347, London, UK, UK. Springer-Verlag.

Bridges, C. P. and Vladimirova, T. (2008). Dual core System-on-a-Chip design to support Inter-Satellite communications. In *AHS '08: Third NASA/ESA Conference on Adaptive Hardware and Systems*, pages 191–198.

Brooks, R. A. and Flynn, A. M. (1989). Fast, cheap and out of control: A robot invasion of the solar system. *Journal of the British Interplanetary Society*, 42:478–485.

Brown, O. and Eremenko, P. (2006). The value proposition for fractionated space architectures. AIAA Paper 2006-7506.

Brown, O., Eremenko, P., and Roberts, C. (2006). Cost-benefit analysis of a notional fractionated SATCOM architecture. In *24th AIAA International Communications Satellite Systems Conference (ICSSC-2006)*.

Brunner, K. and Meltzer, A. H. (1971). The uses of money: Money in the theory of an exchange economy. *The American Economic Review*, 61(5):784–805.

Bullock, S. and Cliff, D. (2004). Complexity and emergent behaviour in ICT systems. Technical Report HPL-2004-187, HP Labs.

Burns, R., Mclaughlin, C. A., Leitner, J., and Martin, M. (2000). TechSat 21: formation design, control, and simulation. In *Aerospace Conference Proceedings, 2000 IEEE*, volume 7, pages 19–25 vol.7.

Cagan, P. (1958). Why do we use money in open market operations? *The Journal of Political Economy*, 66(1):34–36.

Cai, K., Gerding, E., Mcburney, P., Niu, J., Parsons, S., and Phelps, S. (2009). Overview of CAT: A market design competition version 2.0. Technical Report ULCS-09-005, University of Liverpool, Department of Computer Science.

Camp, T., Boleng, J., and Davies, V. (2002). A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC): Special Issue On Mobile Ad Hoc Networking: Research, Trends and Applications*, 2:483–502.

Campo, A. and Dorigo, M. (2007). Efficient multi-foraging in swarm robotics. In Almeida, Rocha, L. M., Costa, E., Harvey, I., and Coutinho, A., editors, *Advances in Artificial Life*, volume 4648 of *Lecture Notes in Computer Science*, pages 696–705, Berlin, Heidelberg. Springer.

Casavant, T. L. and Kuhl, J. G. (1988). A taxonomy of scheduling in general-purpose distributed computing systems. *IEEE Transactions on Software Engineering*, 14(2):141–154.

Chaggar, S., Noble, J., and Cliff, D. (2008). The effects of periodic and continuous market environments on the performance of trading agents. *Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, pages 110–117.

Chakravarti, A. J., Baumgartner, G., and Lauria, M. (2006). Self-Organizing scheduling on the organic grid. *International Journal of High Performance Computing Applications*, 20(1):115–130.

Chang, M. H. and Harrington, J. E. (2000). Centralization vs. decentralization in a multi-unit organization: A computational model of a retail chain as a multi-agent adaptive system. *Management Science*, 46(11):1427–1440.

Chien, S., Sherwood, R., Rabideau, G., Zetocha, P., Wainwright, R., Klupar, P., Van Gaasbeck, J., Castano, R., Davies, A., Burl, M., Knight, R., Stough, T., and Roden, J. (2002). The TechSat-21 autonomous space science agent. In *International Conference on Autonomous Agents*. ACM Press.

Chobotov, V. A., editor (2002). *Orbital Mechanics, Third Edition (AIAA Education Series)*. AIAA, 3rd edition.

Choffnes, D. R. and Bustamante, F. E. (2005). An integrated mobility and traffic model for vehicular wireless networks. In *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, VANET '05, pages 69–78, New York, NY, USA. ACM.

Clarke, D. S., Hicks, M. T., Fitzgerald, A. M., Suchman, J. J., Twiggs, R. J., Randolf, J., and Kenny, T. W. (1996). Picosat free flying magnetometer experiment. In *Proceedings of the Tenth Annual AIAA / USU Conference on Small Satellites*.

Clearwater, S. H., editor (1996). *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific, River Edge, NH.

Cliff, D. (1998). Genetic optimization of adaptive trading agents for double-auction markets. In *Proceedings of the IEEE/IAFE/INFORMS 1998 Conference on Computational Intelligence for Financial Engineering (CIFEr).*, pages 252–258.

Cliff, D. and Bruten, J. (1997). Zero is not enough: On the lower limit of agent intelligence for continuous double auction markets. Technical Report HPL-97-141, HP Laboratories, Bristol.

Cliff, D. and Bruten, J. (1998). Simple bargaining agents for decentralized market-based control. In *Proceedings of the 12th European Simulation Multiconference on Simulation — Past, Present and Future*, pages 478–485. SCS Europe.

Cliff, D. and Bruten, J. (1999). Animat market — trading interactions as collective social adaptive behavior. *Adaptive Behavior*, 7(3-4):385–414.

Coakley, T. P. (1992). *Command and Control for War and Peace*. Diane Pub Co.

Dall, J. and Christensen, M. (2002). Random geometric graphs. *Physical Review E*, 66:016121.

Daniel, K. D. and Hirshleifer, D. A. (1999). A theory of costly sequential bidding. [online]:`http://ssrn.com/abstract=161013`. *Social Science Research Network Working Paper Series.*

DARPA (2010). *DARPA-SN-10-44 System F6 Third-Party Payload Spacecraft Module Request for Information.* DARPA.

Davis, R. and Smith, R. G. (1983). Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, 20:63–109.

De Marco, R. and Menzel, R. (2005). Encoding spatial information in the waggle dance. *Journal of Experimental Biology*, 208(20):3885–3894.

Dias, M. B., Zlot, R., Kalra, N., and Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270.

Dorigo, M., Maniezzo, V., and Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 26(1):29–41.

Dudek, G., Jenkin, M. R. M., Milios, E., and Wilkes, D. (1996). A taxonomy for multi-agent robotics. *Autonomous Robots*, 3(4):375–397.

Durfee, E. (2004). Challenges to Scaling-Up agent coordination strategies. In Wagner, T. A., editor, *An Application Science for Multi-Agent Systems*, volume 10 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, chapter 7, pages 113–132. Kluwer Academic Publishers, Boston.

Ehrentreich, N. (2007). *Agent-Based Modeling: The Santa Fe Institute Artificial Stock Market Model Revisited (Lecture Notes in Economics and Mathematical Systems).* Springer, first edition.

Ellerman, A. D. and Buchner, B. K. (2007). The European Union emissions trading scheme: Origins, allocation, and early results. *Review of Environmental Economics and Policy*, 1(1):66–87.

Emonet, T., Macal, C. M., North, M. J., Wickersham, C. E., and Cluzel, P. (2005). AgentCell: a digital single-cell assay for bacterial chemotaxis. *Bioinformatics*, 21(11):2714–2721.

Farsiu, S., Robinson, D., Elad, M., and Milanfar, P. (2003). Fast and robust multi-frame super-resolution. *IEEE Transactions on Image Processing*, 13:1327–1344.

Ferguson, P. and How, J. (2003). Decentralized estimation algorithms for formation flying spacecraft. In *the AIAA Guidance, Navigation and Control Conference*, pages 2003–5442.

Friis-Christensen, E., Lühr, H., and Hulot, G. (2006). Swarm: A constellation to study the earth's magnetic field. *Earth, Planets and Space*, 58:351–358.

Fujita, M., Krugman, P., and Venables, A. J. (2001). *The Spatial Economy: Cities, Regions, and International Trade.* The MIT Press.

Garnier, S., Jost, C., Jeanson, R., Gautrais, J., Asadpour, M., Caprari, G., and Theraulaz, G. (2005). Aggregation behaviour as a source of collective decision in a group of cockroach-like-robots. In *Advances in Artificial Life*, pages 169–178.

Gerding, E. H., Dash, R. K., Yuen, D. C. K., and Jennings, N. R. (2007). Bidding optimally in concurrent second-price auctions of perfectly substitutable goods. In *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–8, New York, NY, USA. ACM.

Gerkey, B. P. and Matarić, M. J. (2002). Sold!: auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 18(5):758–768.

Gerkey, B. P. and Matarić, M. J. (2003). Multi-Robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3862–3868.

Gerkey, B. P. and Matarić, M. J. (2004). A formal analysis and taxonomy of task allocation in Multi-Robot systems. *The International Journal of Robotics Research*, 23(9):939–954.

Gibney, M. A., Jennings, N. R., Vriend, N. J., and Griffiths, J. M. (1999). Market-Based call routing in telecommunication networks using adaptive pricing and real bidding. In *LNAI n.1699, Proceedings of the IATA'99 Workshop*, pages 50–65.

Gilbert, N. (2004). Agent-based social simulation: dealing with complexity. [online]: `http://www.agsm.edu.au/bobm/teaching/SimSS/ABSS-dealingwithcomplexity-1-1.pdf`, accessed 18 november 2010.

Gjerstad, S. and Dickhaut, J. (1998). Price formation in double auctions. *Games and Economic Behavior*, 22:1–29.

Gnawali, O., Govindan, R., Polyakov, M., and Bose, P. (2005). Data centric, Position-Based routing in space networks. In *2005 IEEE Aerospace Conference*, pages 1–13. IEEE.

Goyal, S. (2009). *Connections: An Introduction to the Economics of Networks*. Princeton University Press.

Hardin, G. (1968). The tragedy of the commons. *Science*, 162(3859):1243–1248.

Hassanein, H. and Luo, J. (2006). Reliable energy aware routing in wireless sensor networks. In *Second IEEE Workshop on Dependability and Security in Sensor Networks and Systems (DSSNS)*, pages 54–64. IEEE.

Hayek, F. A. (1945). The use of knowledge in society. *The American Economic Review*, 35(4):519–530.

Heidt, H., Puig-suari, J., Moore, A. S., Nakasuka, S., and Twiggs, R. J. (2000). CubeSat: A new generation of picosatellite for education and industry low-cost space experimentation. In *AIAA/USU Annual Conference on Small Satellites*.

Heinzelman, W. R., Kulik, J., and Balakrishnan, H. (1999). Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on mobile computing and networking*, MobiCom '99, pages 174–185, New York, NY, USA. ACM.

Hogg, T. and Huberman, B. A. (2002). Dynamics of large autonomous computational systems. In *Proceedings of the Santa Fe Workshop on Collective Cognition*, pages 295–315.

Hovestadt, M., Kao, O., Keller, A., and Streit, A. (2003). Scheduling in HPC resource management systems: Queuing vs. planning. In Feitelson, D., Rudolph, L., and Schwiegelshohn, U., editors, *Job Scheduling Strategies for Parallel Processing*, volume 2862 of *Lecture Notes in Computer Science*, chapter 1, pages 1–20. Springer Berlin / Heidelberg, Berlin, Heidelberg.

Hsieh, C. (2003). Optimal task allocation and hardware redundancy policies in distributed computing systems. *European Journal of Operational Research*, 147(2):430–447.

Huberman, B. A., editor (1988). *The ecology of computation.* Elsevier Science Inc., New York, NY, USA.

Huberman, B. A. and Hogg, T. (1995). Distributed computation as an economic system. *The Journal of Economic Perspectives*, 9(1):141–152.

Hurwicz, L. (1973). The design of mechanisms for resource allocation. *The American Economic Review*, 63(2):1–30.

Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J., and Silva, F. (2003). Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions on Networking*, 11(1):2–16.

Izzo, D., Pettazzi, L., and Ayre, M. (2005). Mission concept for autonomous on orbit assembly of a large reflector in space. *56th International Astronautical Congress.*

Jacyno, M., Bullock, S., Payne, T., Geard, N., and Luck, M. (2008). Autonomic resource management through self-organising agent communities. In *International Conference on Self-Adaptive and Self-Organizing Systems*, pages 475–476, Los Alamitos, CA, USA. IEEE Computer Society.

Jamal and Kamal, A. E. (2004). Routing techniques in wireless sensor networks: a survey. *IEEE Wireless Communications*, 11(6):6–28.

Jilla, C. D. (2002). *A Multiobjective, Multidisciplinary Design Optimization Methodology for the Conceptual Design of Distributed Satellite Systems.* PhD thesis, Massachusetts Institute of Technology.

Johnson, D. B. and Maltz, D. A. (1996). Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers.

Judd, J. S. and Kearns, M. (2008). Behavioral experiments in networked trade. In *Proceedings of the 9th ACM conference on Electronic commerce*, EC '08, pages 150–159, New York, NY, USA. ACM.

Kakade, S., Kearns, M., and Ortiz, L. (2004). Graphical economics learning theory. volume 3120 of *Lecture Notes in Computer Science*, chapter 2, pages 17–32. Springer Berlin / Heidelberg, Berlin, Heidelberg.

Kakde, O. G. (2007). *Theory of Computation.* Laxmi Publications.

Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1):41–50.

Killingback, T., Bieri, J., and Flatt, T. (2006). Evolution in group-structured populations can resolve the tragedy of the commons. *Proceedings of the Royal Society B: Biological Sciences*, 273(1593):1477–1481.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawai, E., and Matsubara, H. (1998). RoboCup: A challenge problem for AI and robotics. In *RoboCup-97: Robot Soccer World Cup I*, pages 1–19.

Klemperer, P. (2002). How (not) to run auctions: The European 3G telecom auctions. *Social Science Research Network Working Paper Series.*

Konolige, K., Fox, D., Ortiz, C., Agno, A., Eriksen, M., Limketkai, B., Ko, J., Morisset, B., Schulz, D., Stewart, B., and Vincent, R. (2006). Centibots: Very large scale distributed robotic teams. *Experimental Robotics IX*, 21:131–140.

Kota, R., Gibbins, N., and Jennings, N. R. (2009). Self-organising agent organisations. In *The Eighth International Conference on Autonomous Agents and Multiagent Systems*, pages 797–804.

Kraus, S. (2001). Automated negotiation and decision making in multiagent environments. *Multi-Agent Systems and Applications*, pages 150–172.

Krieger, M. J., Billeter, J. B., and Keller, L. (2000). Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992–995.

Krugman, P. R. (1996). *The Self-Organizing Economy.* Blackwell Publishers, first edition.

Ladley, D. and Bullock, S. (2005). The role of logistic constraints in termite construction of chambers and tunnels. *Journal of Theoretical Biology*, 234(4):551–564.

Larson, W. J. and Wertz, J. R., editors (1999). *Space Mission Analysis and Design.* Microcosm Press, third edition.

Lau, H. C. and Zhang, L. (2003). Task allocation via multi-agent coalition formation: taxonomy, algorithms and complexity. In *Tools with Artificial Intelligence, 2003. Proceedings. 15th IEEE International Conference on*, pages 346–350.

Law, A. M. and Kelton, D. W. (2000). *Simulation Modelling and Analysis*. McGraw-Hill Education — Europe.

Lee, J. S., Luu, T., and Konangi, V. K. (2005). Design of a satellite cluster system in distributed simulation. *Simulation*, 81(1):57–66.

Lerman, K. (2004). A model of adaptation in collaborative multi-agent systems. *Adaptive Behavior*, 12(3-4):187–197.

Lobosco, D. M., Cameron, G. E., Golding, R. A., and Wong, T. M. (2008). The Pleiades fractionated space system architecture and the future of national security space. In *AIAA SPACE 2008 Conference*.

Marwell, G. and Ames, R. E. (1979). Experiments on the provision of public goods. I. Resources, interest, group size, and the free-rider problem. *American Journal of Sociology*, 84(6):1335–1360.

Mason, C. F. and Phillips, O. R. (1997). Mitigating the tragedy of the commons through cooperation: An experimental evaluation. *Journal of Environmental Economics and Management*, 34(2):148–172.

Matarić, M. J., Sukhatme, G. S., and Østergaard, E. H. (2003). Multi-robot task allocation in uncertain environments. *Autonomous Robots*, 14(2):255–263.

Milgrom, P. R. (1985). *The economics of competitive bidding: a selective survey*, pages 261–292. Cambridge Uniersity Press.

Miller, M. S. and Drexler, E. K. (1988). Markets and computation: Agoric open systems. In Huberman, B. A., editor, *The Ecology of Computation*. North-Holland, Amsterdam.

Modi, P. J., Shen, W.-M., Tambe, M., and Yokoo, M. (2006). ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161:149–180.

Moss, S. (2001). Game theory: Limitations and an alternative, [online]: `http://ssrn.com/abstract=262547`. *Social Science Research Network Working Paper Series*.

Mueller, J. B. and Brito, M. (2003). A distributed flight software design for satellite formation flying control. In *AIAA Space Conference*.

Mueller, J. B. and Thomas, S. J. (2005). Decentralized formation flying control in a multiple-team hierarchy. *Annals of the New York Academy of Sciences*, 1065:112–138.

Niu, J., Cai, K., Gerding, E., McBurney, P., and Parsons, S. (2008). Characterizing effective auction mechanisms: Insights from the 2007 TAC market design competition. In *The 7th International Conference on Autonomous Agents and Multiagent Systems*.

Oster, G. F. and Wilson, E. O. (1979). *Caste and Ecology in the Social Insects. (MPB-12) (Monographs in Population Biology)*. Princeton University Press.

Pinciroli, C., Birattari, M., Tuci, E., Dorigo, M., Marco, Vinko, T., and Izzo, D. (2008). Self-organizing and scalable shape formation for a swarm of pico satellites. In *AHS '08: NASA/ESA Conference on Adaptive Hardware and Systems*, pages 57–61.

QB50 (2010). QB50, an international network of 50 CubeSats for multi-point, in-situ measurements in the lower thermosphere and re-entry research. Technical report. [online] `http://www.vki.ac.be/QB50/`, accessed 7 July 2010.

Resnick, M. (1997). *Turtles, termites, and traffic jams: explorations in massively parallel microworlds*. Complex adaptive systems. MIT Press.

Robinson, N. (2002). Evolutionary optimization of market-based control systems for resource allocation in compute farms. Master's thesis (HPL-2002-284), COGS, University of Sussex/HP Labs, Bristol.

Rogers, A., Dash, R. K., Jennings, N. R., Reece, S., and Roberts, S. (2006). Computational mechanism design for information fusion within sensor networks. In *Ninth International Conference on Information Fusion (Fusion 2006)*.

Rogers, A., David, E., and Jennings, N. R. (2004). Selfish sensors in wireless micro-sensor networks. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1322–1323, Washington, DC, USA. IEEE Computer Society.

Rogers, A., David, E., and Jennings, N. R. (2005). Self-organized routing for wireless microsensor networks. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(3):349–359.

Rosenschein, J. S. and Genesereth, M. R. (1985). Deals among rational agents. In *IJCAI'85: Proceedings of the 9th international joint conference on artificial intelligence*, pages 91–99, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Roughgarden, T. (2005). *Selfish Routing and the Price of Anarchy.* The MIT Press.

Schetter, T., Campbell, M., and Surka, D. (2000). Multiple Agent-Based autonomy for satellite constellations. *Agent Systems, Mobile Agents, and Applications*, 1882:545–581.

Seuken, S. and Zilberstein, S. (2008). Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems*, 17:190–250. 10.1007/s10458-007-9026-5.

Shackleton, M., Saffre, F., Tateson, R., Bonsma, E., and Roadknight, C. (2004). Autonomic computing for pervasive ICT — a whole-system perspective. *BT Technology Journal*, 22(3):191–199.

Shaw, G. B. (1999). *The Generalized Information Network Analysis Methodology for Distributed Satellite Systems.* PhD thesis, Massachusetts Institute of Technology.

Shehory, O. and Kraus, S. (1998). Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200.

Shoham, Y. and Leyton-Brown, K. (2008). *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations.* Cambridge University Press.

Si-wei, C., Jing, C., Lin-Cheng, S., and Yi, T. (2010). ECNP-based method of distributed dynamic task allocation for multiple observation satellite planning. In *2nd International Conference on Advanced Computer Control (ICACC)*, volume 4, pages 325–328. IEEE.

Slater, D. and Tonkiss, F. (2001). *Market Society: Markets and Modern Social Theory.* Polity.

Smith, A. (1776). *The Wealth of Nations: Books 1-3.* Penguin Classics (Original work published 1776, reprint 1986).

Smith, R. G. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12):1104–1113.

Stonebraker, M., Devine, R., Kornacker, M., Litwin, W., Pfeffer, A., Sah, A., and Staelin, C. (1994). An economic paradigm for query processing and data migration in Mariposa. In *Proceedings of 3rd International Conference on Parallel and Distributed Information Systems*, pages 58–67.

Streit, A. (2001). On job scheduling for HPC-clusters and the dynP scheduler. In Monien, B., Prasanna, V., and Vajapeyam, S., editors, *High Performance Computing HiPC 2001*, volume 2228 of *Lecture Notes in Computer Science*, chapter 6, pages 58–67. Springer, Berlin, Heidelberg.

Sujit, P. B. and Beard, R. (2007). Multiple MAV task allocation using distributed auctions. In *AIAA Guidance, Navigation and Control Conference and Exhibit*. AIAA.

Sweeting, M. N. (1992). UoSAT microsatellite missions. *Electronics & Communications Engineering Journal*, 4(3):141–150.

Tesfatsion, L. (2002). Agent-Based computational economics: Growing economies from the bottom up. *Artificial Life*, 8(1):55–82.

Thanapalan, K. K. T. and Veres, S. M. (2005). Agent based controller for satellite formation flying. In *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2005. Proceedings of the 2005 International Conference on*, pages 385–389.

Tripp, H. and Palmer, P. (2010). Stigmergy based behavioural coordination for satellite clusters. *Acta Astronautica*, 66(7-8):1052–1071.

van der Horst, J. and Noble, J. (2010). Distributed and centralized task allocation: When and where to use them. In *Self-Adaptive Networks (SAN) Workshop, IEEE International Conference on Self-Adaptive and Self-Organising Systems (SASO2010)*.

van der Horst, J. and Noble, J. (2011). Task allocation in dynamic networks of satellites. In *IJCAI Workshop on Artificial Intelligence in Space*.

van der Horst, J. and Noble, J. (2012). Task allocation in networks of satellites with Keplerian dynamics. *Acta Futura*, 5:143–150.

van der Horst, J., Noble, J., and Tatnall, A. (2009). Robustness of market-based task allocation in a distributed satellite system. In *Advances in Artificial Life: Tenth European Conference on Artificial Life (ECAL '09)*. Springer.

Vaughan, R., Støy, K., Sukhatme, G., and Matarić, M. (2000). Go ahead, make my day: Robot conflict resolution by aggressive competition. In *Proc. of the Intl. Conf. on Simulation of Adaptive Behavior (SAB)*, pages 491–500.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37.

Vladimirova, T., Wu, X., and Bridges, C. P. (2008). Development of a satellite sensor network for future space missions. In *Aerospace Conference, 2008 IEEE*, pages 1–10.

Vladimirova, T., Wu, X., Sidibeh, K., Barnhart, D., and Jallad, A. (2006). Enabling technologies for distributed picosatellite missions in LEO. In *AHS '06: First NASA/ESA Conference on Adaptive Hardware and Systems*, pages 330–337.

Voos, H. and Litz, L. (2000). Market-based optimal control: a general introduction. In *Proceedings of the 2000 American Control Conference*, volume 5, pages 3398–3402 vol.5.

Vytelingum, P., Dash, R. K., David, E., and Jennings, N. R. (2004). A risk-based bidding strategy for continuous double auctions. In *16th European Conference on Artificial Intelligence, 79-83.*

Waldspurger, C. A., Hogg, T., Huberman, B. A., Kephart, J. O., and Stornetta, S. W. (1992). Spawn: A distributed computational economy. *IEEE Transactions on Software Engineering*, 18(2):103–117.

Wellman, M. P. (1993). A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23.

Wellman, M. P. (1996). Market-oriented programming: Some early lessons. In Clearwater, S., editor, *Market-Based Control: A Paradigm for Distributed Resource Allocation.* World Scientific.

Wellman, M. P., Cheng, S. F., Reeves, D. M., and Lochner, K. M. (2003). Trading agents competing: performance, progress, and market effectiveness. *IEEE Intelligent Systems*, 18(6):48–53.

White, T. and Helferty, J. (2005). Emergent team formation: Applying division of labour principles to robot soccer. *Engineering Self-Organising Systems*, pages 180–194.

Wilson, D. and Sober, E. (1989). Reviving the superorganism. *Journal of Theoretical Biology*, 136(3):337–356.

Wooldridge, M. (2002). *An Introduction to MultiAgent Systems.* John Wiley & Sons, 1st edition.

Wu, X., Vladimirova, T., and Sidibeh, K. (2008). Signal routing in a satellite sensor network using optimisation algorithms. In *Aerospace Conference, 2008 IEEE*, pages 1–9.

Yao, A. C. C. (1980). New algorithms for bin packing. *Journal of the ACM*, 27(2):207–227.

Ygge, F. and Akkermans, H. (1999). Decentralized markets versus central control: A comparative study. *Journal of Artificial Intelligence Research*, 11:301–333.

Zetocha, P., Self, L., Wainwright, R., Burns, R., Brito, M., and Surka, D. (2000). Commanding and controlling satellite clusters. *IEEE Intelligent Systems*, 15(6):8–13.

Zlot, R., Stentz, A., Dias, M. B., and Thayer, S. (2002). Multi-robot exploration controlled by a market economy. In *IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 3, pages 3016–3023. IEEE.

Zlot, R. M. (2006). *An Auction-Based Approach to Complex Task Allocation for Multirobot Teams.* PhD thesis, Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA.