DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA
ESCUELA SUPERIOR DE INGENIEROS
UNIVERSIDAD DE SEVILLA

Market-based distributed task allocation methodologies applied to multi-robot exploration

por

Luis Antidio Viguria Jiménez

PROPUESTA DE TESIS DOCTORAL
PARA LA OBTENCIÓN DEL TÍTULO DE

DOCTOR INGENIERO DE TELECOMUNICACIÓN

SEVILLA, NOVIEMBRE 2009

Director

Dr.-Ing. Aníbal Ollero Baturone, Catedrático de Universidad

UNIVERSIDAD DE SEVILLA

Memoria para optar al grado de Doctor Ingeniero de Telecomunicación por la Universidad de Sevilla

Autor:	**Luis Antidio Viguria Jiménez**
Título:	**Market-based distributed task allocation methodologies applied to multi-robot exploration**
Departamento:	**Departamento de Ingeniería de Sistemas y Automática**

V° B° Director:

Aníbal Ollero Baturone

El autor:

Luis Antidio Viguria Jiménez

A mis padres.

Acknowledgements

The work presented in this thesis would not have been possible without the support, guidance and inspiration of so many individuals: family, friends and colleagues. Their help and good advice constitute the basis of my research. Thanks to all of them.

In particular, I would like to thank my advisor Professor Aníbal Ollero for giving me the opportunity for working in this amazing field. This thesis would not have come to the form and shape it is today without him. Of course, I am truly grateful to the rest of the Robotics, Vision and Control Research Team at the University of Seville; specially to Ivan, Fernando and Luis.

Special thanks go to Dr. Ayanna M. Howard, from Georgia Institute of Technology, for all her support and guidance over the two years that I stayed in Atlanta. I learned a great deal from her motivation and clarity of purpose. Also, I am very grateful to all the members of HumAnS Lab for their unselfish help and useful comments.

I would like to show my gratitude to the Spanish Ministry of Education and Science, the Science and Innovation Section of the Andalusian Government, and the Fulbright Commission in Spain for financial support. Also, I would like to thank the board of directors of FADA-CATEC that supports this thesis during the last year.

My thesis dedication and my most wholehearted thanks go to my parents, Antidio and Fátima. Without their unconditional love, patience, support and guidance, I would have never come this far. I cannot thank them enough for all they have taught me.

Finally, my most special thanks go to my friend and partner Pepa, I feel very lucky to have her by my side.

Resumen

A medidada que los robots se van integrando en la vida diaria de las personas, se les pide que realicen tareas cada vez más complejas. Muchas de estas tareas se podrían ejecutar más eficientemente por un grupo de robots, en vez de por uno sólo. Trabajando conjuntamente, el equipo de robots puede completar tareas de forma más rápida, incrementando la robustez del sistema e incluso llevando a cabo tareas que son imposibles por un único robot. Sin embargo, coordinar un equipo de robot aún requiere superar importantes retos a nivel científico.

Dentro del campo de estudio de los sistemas multirobot, esta tesis se enfoca en el problema de la asignación de tareas. Este problema intenta responder a la pregunta: ¿qué robot debería ejecutar cada una de las tareas? Este problema tiene una gran importancia en misiones de exploración que hacen uso de varios robots. Por ejemplo, en futuras misiones científicas, se quiere mandar distintos robots instrumentados a lugares de interés científico, que nos permita ampliar nuestro conocimiento sobre el origen de la vida. A la hora de establecer las configuraciones de estos robots, hay que determinar como asignar las posiciones de éstos para que finalmente se obtenga la topología deseada. Este mismo objetivo también se trata cuando se estudia el problema de la asignación de tareas con múltiples robots.

Esta tesis presenta recientes contribuciones en el campo de la cooperación entre múltiples robots. En particular, la investigación llevada a cabo se centra en algoritmos distribuidos de asignación de tareas basados en reglas de mercado. Con mayor detalle, las aportaciones de la tesis se pueden resumir en cuatro puntos fundamentales. En primer lugar, se ha desarrollado un nuevo concepto denominado *servicios*, que permite asignar tareas de las que se desconoce el número de robots necesarios

para ser ejecutadas. El número final de robots necesario es decidido durante un proceso de negociación y depende de las capacidades de los robots. Con respecto a los algoritmos MRMT (Multiple Robots Multiple Tasks), esta tesis presenta un nuevo algoritmo distribuido de asignación de tareas que combina la resasignación y combinación de pujas, que posee tanto una alta eficiencia como tolerancia a fallos. Por otra parte, se han desarrollado mejoras de los algoritmos MRST (Multiple Robots Single Task) ya existentes. La principal diferencia es que se aumenta la información compartida, de manera que los robots eligen tareas que, no sólo son mejores para ellos, sino para el grupo en general. Finalmente, esta tesis desarrolla un marco de trabajo probabilístico para calcular medidas de eficiencia en algoritmos MRST. El análisis consiste en el cálculo del valor esperado de la función objetivo, que es usado más tarde como métrica para comparar diferentes algoritmos. Como el valor esperado de una variable aleatoria indica su valor *en media*, éste supone una medida más informativa que las utilizadas usualmente en la literatura para medir la eficiencia del algoritmo, como por ejemplo el peor caso posible con respecto a la solución óptima.

Por otra parte, la tesis está organizada de la siguiente manera. El primer capítulo introduce y motiva el problema a tratar. A continuación, el Capítulo 2 aborda el problema de la asignación de tareas que necesitan un número desconocido de robots para ser ejecutadas. El número de robots depende de sus capacidades y se calcula durante un proceso de negociación. El concepto de *servicio*, es definido en este punto. Un *servicio* se genera a partir de una tarea con la que está relacionado, y éste puede generar otro servicio creándose una relación jerárquica entre las tareas y los servicios. En este capítulo también se estudia el impacto sobre la ejecución de las tareas que dicha relación provoca.

En el caso en el que la eficiencia de la asignación sea de interés, ésta se puede aumentar, como se muestra en el Capítulo 3, mediante la combinación de los conceptos de reasignación con el de pujas combinatorias. Las tareas se agrupan y se asignan como si fueran una única, en lugar de asignar cada tarea por separado. La eficiencia de la asignación se ve aumentada puesto que los robots calculan sus costes con un horizonte de ejecución más largo y se benefician de las sinergias entre las tareas. Como contrapartida, se necesita un nivel de confianza más alto en los costes de las tareas.

La última parte del Capítulo 3 estudia problemas de sincronización para algoritmos de asignación de tareas basados en reglas de mercado.

En el Capítulo 4 se describen diferentes algoritmos MRST de asignación de tareas. Este tipo de algoritmos no usan planes de ejecución locales, y por tanto, son apropiados para aplicaciones donde los costes pueden cambiar con el tiempo. En el mismo capítulo se discuten varias estrategias para la obtención de soluciones más eficientes que las usuales. Por otra parte, el Capítulo 5 desarrolla un marco de trabajo probabilístico que permite comparar diferentes algoritmos MRST de asignación de tareas. Dentro de este análisis probabilístico, el objetivo es calcular el valor esperado del la función objetivo, que proporciona una medida de la eficiencia del algoritmo. Los resultados teóricos obtenidos son validados mediante simulaciones y experimentos con robots reales. También en este capítulo, se desarrolla un estudio de la eficiencia de un algoritmo en comparación con el valor óptimo. Este capítulo finaliza con una extensión de la metodología propuesta, mediante el cálculo de la distribución probabilística estimada de la función objetivo. Finalmente, las conclusiones y el futuro trabajo a desarrollar de esta tesis se exponen en el Capítulo 6.

Abstract

As robots become an integral part of human life they are charged with increasingly difficult tasks. Many of these tasks can be better achieved by a team of robots than by a single one. By working together, robots can complete tasks faster, increase system robustness, improve solution quality, and achieve tasks impossible for a single robot. Nevertheless, coordinating such a team requires overcoming many formidable research challenges.

Within the multi-robot field of study, this thesis focus on the task allocation problem which tries to answer the question: which robot should execute each task? This problem has a mayor impact in exploration applications using multiple robots. For example, in future science exploration missions, there is a desire to send multiple, instrumented rovers to scientific sites of interest to expand our understanding of both the history and future of life. Establishment of the sensor configurations involves determining how to allocate sensor positions to mobile sensor agents in order to achieve a desired topology, a similar research objective is found when focusing on the task allocation problem with teams of robots.

This thesis proposes different distributed algorithms to solve the multi-robot task allocation (MRTA) problem. These algorithms must be as efficient as possible trying to obtain solutions close to the optimal. Also, the algorithms have to be robust enough to be highly fault tolerant. In order to fulfill the commented characteristics, the market-based approach has been chosen to develop the novel algorithms presented in this thesis.

These algorithms are divided into two categories: MRMT (Multiple Robots Multiple Tasks) and MRST (Multiple Robots Single Task). The former makes use of local execution plans, and it obtains more efficient solutions. While the latter only allocates one task per robot, and it is suited for applications where task cost may change through time. For MRST algorithms, this thesis presents a novel approach to study the performance of task allocation algorithms. This theoretical analysis is based on a probabilistic approach which is used to obtain metrics that model the algorithm performance.

Finally, the thesis is supported by an extensive experimental work where the proposed algorithms have been tested and validated using real robots.

Contents

List of Tables

List of Figures

List of Acronyms

BBCS	Black Board Communication System
BS	BaSic market-based
BS-WR	BaSic market-based algorithm Without Reallocations
CDF	Cumulative Distribution Function
CNP	Contract Net Protocol
GPS	Global Positioning System
IP	Internet Protocol
MRMT	Multiple Robots Multiple Tasks
MRST	Multiple Robots Single Task
MRTA	Multi-Robot Task Allocation
RMA	Robot Mean Allocation algorithm
RTMA	Robot and Task Mean Allocation
SIT-MASR	dynamic SIngle Task negotiations with Multiple Allocations to a Single Robot
SET-MASR	dynamic task subSETs negotiations with Multiple Allocations to a Single Robot
S+T	Services and tasks
TMA	Task Mean Allocation
TSP	Traveling Salesman Problem
UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
WP	WayPoint

Chapter 1

Introduction

This thesis presents contributions in the field of cooperation within robot teams. More precisely, this research has focused on distributed task allocation algorithms based on market approaches.

This first chapter presents the motivation and the main objectives of the research carried out. The task allocation problem is localized within the multi-robot problem domain. Next, an overview of existing works related to the multi-robot task allocation problem is described.

On the other hand, the fundamentals of market-based approaches are explained. These concepts are used through the rest of this thesis. Moreover, a summary of previous works that have used this approach to solve the task allocation problem is exposed. Next, the scope of this research is limited by explaining the characteristics and assumptions considered in the multi-robot system and the task allocation problem. Finally, the thesis outline and main contributions are presented.

1.1 Motivation and objectives

In the following years, it is predicted that systems based on robots will have to solve more complex problems and in a more efficient manner. Although, in the current situation, the majority of these systems are composed of only one robot, this is not necessarily the best solution when a high fault tolerance level is needed or tasks present an important grade of diversity. For these reasons, multi-robot systems has gained popularity as a research topic in the last decade. In (Parker, 2008), it is shown how research works related to the multi-robot field has significantly increased recently.

Multi-robot systems are preferable for tasks that are inherently distributed in space, time, or functionality. For problems that can be separated in independent subproblems, the use of a multi-robot system offers the potential of reducing the overall task completion time. Moreover, multi-robot systems offer the possibility to increase the robustness and reliability of the solution combining redundant systems. This is possible due to the ability for one robot to replace the role or activities of a failing one. A multi-robot system offers a greater flexibility on the system design since each robot does not have to perform all the possible tasks. For many applications, a single robot approach can lead to the design of a very complex and expensive system. However, a multi-robot system can be composed of more specialized units that offer the possibility of reducing the complexity of each robot. Finally, a number of robots can share information and improve their perception of the environment using cooperative perception techniques (Merino, 2008). In the last decade, several successful multi-robot systems have demonstrated the viability and effectiveness of this approach for some specific tasks, for example: logistics on warehouses (Wurman et al., 2008), data recollection in natural environments (Report, 2002), and the ability to play team sports such as soccer (Veloso et al., 1999) and (Vecht and Lima, 2004).

Within the multi-robot field of study, this thesis focus on the task allocation problem which tries to answer the question: which robot should execute each task? This problem has a mayor impact in exploration applications using multiple robots. For example, in future science exploration missions, there is a desire to send multiple, instrumented rovers to scientific sites of interest to expand our understanding of both the history and future of life. Mars exploration missions are focused on finding signs of life to expand our comprehension of where life began. Earth exploration missions are focused on resolving theories on how life evolved and how it might be effected in the future. These mission examples all have one common theme; scientists and autonomous rovers must work together to navigate in extreme environments in order to collect scientific measurements of interest. Establishment of these sensor configurations involves determining how to allocate sensor positions to mobile sensor agents in order to achieve a desired topology, a similar research objective is found when focusing on the task allocation problem with teams of robots. These exploration applications require to have a special attention to two main factors: robots energy consumption and fault tolerance. For example, if a group of robots are sent to Mars, it is interesting to have them working the maximum possible time and the exploration mission cannot be conditioned to the failure of one or more robots.

This thesis proposes different distributed algorithms to solve the multi-robot task allocation (MRTA) problem. These algorithms should be as efficient as possible trying to obtain solutions close to the optimal. Also, the algorithms have to be robust enough to be highly fault tolerant. The main objective of this thesis is to develop novel distributed task allocation algorithms that fulfill the commented characteristics. Also, this thesis has the objective of studying these algorithms from a theoretical point of view in order to obtain metrics that model the algorithm performance. Finally, the analysis of the problem takes into account practical implementation issues in order to use these algorithms on a team of real robots and demonstrate the system functionalities.

1.2 Task allocation within the multi-robot problem domain

Within the multi-robot domain there are two important issues to take into account: coordination and cooperation (see (Lima and Custodio, 2004) for a global view of the different problems related to the multi-robot field). In general, coordination can be understood as a process that arises within a system when given resources are simultaneously required by several components. In the case of a multirobot system, a classic coordination issue to deal with is space sharing between different robots. Its goal is to ensure that each robot is able to perform its plan safely and coherently regarding the plans of the other robots. Another important issue is time synchronization. This

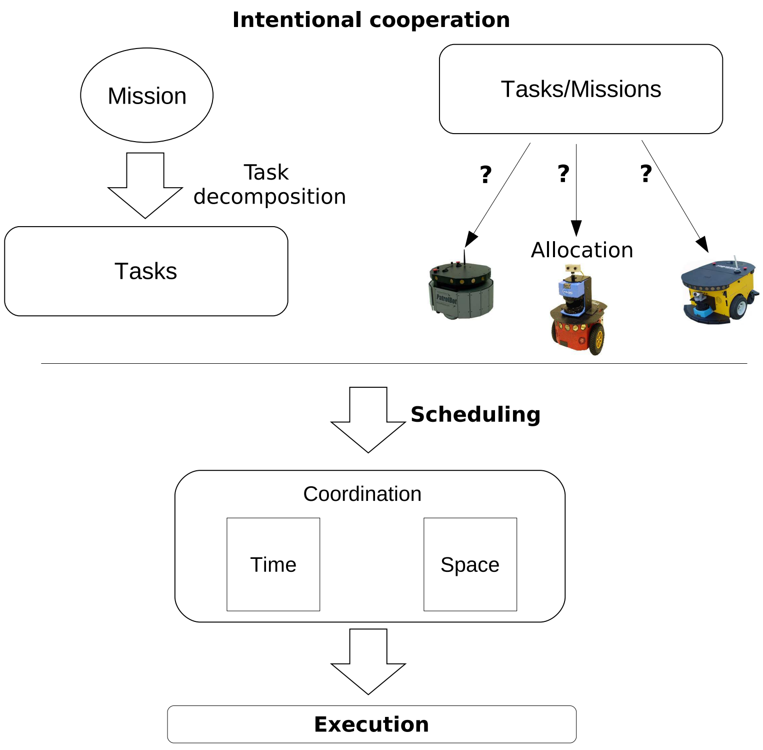


Figure 1.1: Different problems that should be solved within the multi-robot domain.

type of coordination enables robots to execute tasks before, after or during the execution of other tasks, and it can be very useful in a large number of applications, for instance, in the case of a monitoring task requiring several synchronized perceptions of the event. As can be seen in Figure 1.1, both space and time coordinations are usually taken into account in the scheduling phase. In this phase, each robot has a list of tasks to execute and should determine the most efficient total ordering in which to perform them while ensuring that any inter-task (time coordination) and inter-robot (space coordination) constraints are met.

On the other hand, cooperation is defined as a group behavior that leads to a certain objective which is of common interest. For example, in (Cao et al., 1997), the cooperative behavior is explained as an underlying mechanism that increases the total utility of the system. Moreover, two types of cooperation can be considered (Parker, 1998): swarm cooperation and intentional cooperation. The former is applied to large number of homogeneous robots, where each one has little capabilities (Tabauda et al., 2005). In this case robots, as a group, behave intelligently and efficiency metrics are not usually considered. The latter is applied to smaller number of robots that can be heterogeneous. Each robot has important individual capabilities and can accomplish meaningful tasks alone. This type of cooperation often takes into account performance metrics since energy or time constraints play an important role. Therefore, the intentional cooperation requires more directed, complex mechanisms than the swarm cooperation.

In a general scenario, a number of high-level tasks or missions have to be executed by a team of robots. There are three general phases that should be performed before starting the execution: task decomposition, task allocation, task scheduling (see Figure 1.1). A mission planner decomposes a given instance of a high-level team mission description into an efficient plan made up of simpler tasks that can be executed by the considered robots and satisfy the mission requirements. This decomposition can be performed either by a human or automatically by a computational algorithm. Usually, missions are first decomposed to tasks, and next, tasks are distributed among the different robots by means of a task allocation algorithm. However, this order can change or both problems can be solved simultaneously as in (Zlot and Stentz, 2006). Finally, before the execution of tasks, there is a scheduling phase that takes into account both space and time coordination.

Within all the problems related to the multi-robot domain, this thesis is focused on solving the task allocation problem. This type of problem is solved by algorithms that associate tasks to robots while usually trying to optimize an objective function. In this thesis, it is assumed that the task decomposition problem is solved and the list of tasks to be allocated are already calculated.

1.3 Related work

In this section a summary of the literature related with the MRTA problem is presented. Different approaches [1] have been developed in the last decade and an initial

[1]Although the task allocation problem has been studied in the field of multi-agent systems, usually those algorithms cannot be applied directly to multi-robot systems, or they need to be

classification can be made at the organization level: centralized, distributed, and hybrid approaches. Next, the main research trends found in each level are commented.

1.3.1 Centralized approach

In a centralized approach, all the information is transmitted to a central server that usually calculates the optimal allocation. This approach is usually less robust than the distributed one, mainly because of the existence of a central element (Brumitt and Stenz, 1998) and (Caloud et al., 1990). Furthermore, this element should have enough computing capacity to calculate an optimal or at least suboptimal allocation in a coherent time since the MRTA problem is $\mathcal{NP}$-hard (Lagoudakis et al., 2005). This statement can be easily proven since the different MRTA problems resemble the Multiple Traveling Salesman Problem (Lawler, 1985), the Min-Max Vehicle Routing Problem (Applegate et al., 2002), and the Traveling Repairperson Problem (Jamil et al., 1994), which are intractable even on the Euclidean plane. For this reason, the centralized approach uses results from the Operational Research field of study. Other problems related to this approach includes limitations with communication coverage, robustness and scalability. On the other hand, the primary advantage is that solutions are usually optimal or very close to the optimal, but always under the assumption that the information from the different robots is accurate enough.

1.3.2 Distributed approach

In a distributed approach, robots use local information and inter-robot communication to allocate tasks without the need of a central element. The distributed approach is more complex because robots have to solely allocate tasks without having access to all the information. First, distributed task allocation algorithms that do not require explicit communication between robots are considered. For example, ALLIANCE (Parker, 1998) is based on distributed behaviors that uses motivations mathematically modeled. Task allocation is based on the implementation of motivations, which

adapted in order to fulfill the particularities of multi-robot systems, see (Vig and Adams, 2006). The main reason is that multi-agent systems do not consider the uncertainties that are so important in robotics. For this reason, these research works are not considered in this chapter.

are impatience and consent. Impatience allows a robot to manage a situation where other robots fail to execute their own tasks, while consent allows a robot to manage a situation where it fails to execute its own task. These motivations activate the different behaviors using information about the environment and the rest of robots. For example, from the time a new task is announced, the level of impatience of each robot increases at different speeds. The more suitable the task for the robot, the faster the level of impatience increases. When impatience exceeds a limit, the robot activates the behaviors needed to execute the task. From that moment, the impatience of the rest of the robots increases much slower to avoid having more than one robot executing the same task. This algorithm was improved with learning techniques used for parameter adaptation in (Parker, 1997).

A similar idea is developed in (Agassounon and Martinoli, 2002) and (Krieger and Billeter, 2000) called threshold-based task allocation where each robot has an activation threshold for each task that needs to be performed. They define the stimulus as a value that reflects the urgency or importance of performing a task. The stimuli are perceived continuously for each of the tasks. When the stimulus for a robot exceeds a certain threshold, it starts executing the task. When the stimulus falls below the threshold, the robot stops the behaviors that execute the task. This reaction to the stimulus can be deterministic or probabilistic.

A solution completely based on behaviors can be found in (Werger and Matarić, 2000), where a system called BLE is used to solve the CMOMMT (*Cooperative Multi-robot Observation of Multiple Moving Targets*) problem. This system extends the subsumption architecture (Brooks, 1986) to multiple robots. To achieve this, BLE uses suppression and inhibition techniques between the behaviors of the different robots of the system, called cross inhibition. Basically, it is based on the fact that each robot executes the task for which it is best prepared. At the time a robot starts executing a task, it inhibits the same level behaviors from the rest of robots. In this way, the robot is demanding the task. Cross inhibition is an active process. If one robot fails, it will stop to inhibit the behaviors of the other robots, and afterwards, one of them will take care of the task.

Recently, another task allocation approach (Dahl et al., 2008) has been used with behavior-based robots. This distributed algorithm uses the concept of vacancy chains to allocate tasks. A vacancy chain is a social structure through which resources are distributed to consumers. This algorithm is completely distributed and communication-free since they use local task selection procedures. Reinforcement learning techniques are also used for estimation of task utility and rewards structures.

On the other hand, another group of fully distributed task allocation algorithms can be found. But they do not allocate specific tasks to robots; their only aim is to divide the group of robots in subgroups, where each subgroup executes a different task. For example, 30% of the robots should execute task A and the other 70% execute task B. In (Lerman et al., 2006) a dynamic task allocation algorithm for this type of problem with theoretical analysis is presented. Also, in (McLurkin and Yamins, 2005) four different algorithms are explained and tested with real robots. Although this type of problem is considered within the multi-robot task allocation domain, it is not studied in this thesis. The previous commented algorithms are high fault tolerant since they do not make use of explicit communication. However, these approaches do not have in their priorities the efficiency of the solution and their main objective is to finish the mission successfully; where a mission can be defined as a partially ordered group of tasks.

Most of the rest of the distributed task allocation algorithms use explicit communication messages which means that robots make decisions based on inter-robot communications transmitted at different time instances. This characteristic makes the algorithms more efficient than the previous algorithms, and with a higher level of fault tolerance than a centralized approach due to its distributed nature. Negotiation techniques based on market rules (market-based approach) fall within the distributed algorithms that make use of explicit communication. These techniques have received significant attention (Dias et al., 2006) since they offer a good compromise between communication requirements and the quality of the allocation. A detail summary of this approach is described in Section 1.4, since this is the approach used in the thesis.

The token-based technique, (Farinelli et al., 2006) and (Scerri et al., 2005), can also be considered within the explicit communication group. In this technique, each token represents a different task to be executed. When a robot receives a token, it decides whether to perform the task associated to it or to pass the token to another robot. This decision is usually made based only on local information. To prevent conflicts regarding token coherence, distributed algorithms have also been developed (Farinelli et al., 2005). Another work with a similar approach can be found in (Xu et al., 2005) where the robots use local decision theoretic models to determine when and where to pass the tokens.

In (Fua et al., 2004), a different task allocation approach is described. Instead of using utilities or costs to compare different tasks, they use a suitability metric. This allows them to transform the task allocation problem into a Transportation Problem (TP) (Munkres, 1957) which has $\mathcal{P}$ complexity, instead of treating the problem as a Multiple Traveling Salesman Problem or similar that are $\mathcal{NP}$-hard. However, their algorithm maximizes the robot suitability for each task which does not mean, for most of the cases, that the utility is maximized as well. In this approach, robots exchange task suitability matrices (TSM) that models the state of the system. Using the updated information, each robot solves the TP problem. Finally, they consider the fault tolerance aspect (Fua and Ge, 2005) using a backoff scheme which is based on ideas from communication protocols.

In the case where all tasks and robot states are completely known, a valid distributed approach could be used to run an optimal algorithm on each robot, and in theory, all the robots will obtain the same solution (the optimum). Each robot executes the tasks that the optimal algorithm has decided, and they will not overlap. However this method strongly depends on the assumption that the information is accurate and all the robots have the same view of the system or situational awareness information, which is not always true in a distributed robotic system. Therefore, in order to use this approach in a real system, algorithms should be robust to differences in the situational awareness information. In (Gil et al., 2003), a distributed robust approach where all the robots have a common cooperative scheduling strategy is explained. They make use of exclusion algorithms to deal with the asynchronous aspect of the decision making process and the inaccuracy of the information.

A different work, within the same approach, can be consulted in (Alighanbari, 2005) that uses an algorithm that adds a second planning step based on sharing the planning data. This approach is analogous to closing a synchronization loop on the planning process to reduce the sensitivity to incorrect data. This work has been applied to the task allocation problem for a group of Unmanned Aerial Vehicles (UAVs) (Bethke et al., 2008). They use consensus techniques to overcome the differences between the situational awareness of each UAV. A new version of the algorithm can be found in (Alighanbari, 2008) that combines robust planning with techniques developed to eliminate the churning coming from the replanning when the situational awareness is updated. This version is less conservative than robust planning and does not suffer from churning type of instability.

A similar idea is studied in (Beard et al., 2006). In this case, every robot calculates the optimal solution assuming perfect global information. Afterwards, a consensus algorithm is used to ensure that each robot has consistent information. The consensus algorithm makes use of a communication network, but only local communication is assumed. The approach is implemented on a team of fixed-wing UAVs.

Finally, in (Atay and Bayazit, 2007a) mixed integer linear programming techniques with local information is used for task allocation. In order to improve the results, information is shared between neighbors. Although only local information is used for the linear programming algorithm, they state that the system reaches an equilibrium when all the robots find the same solution (Atay and Bayazit, 2007b).

1.3.3 Hybrid approach

A hybrid approach tries to improve the efficiency of the solution with respect to the distributed systems without significantly reducing the fault tolerance aspect. There has not been much work done using this approach.

In (Dias and Stenz, 2002), they use several central elements called leaders. Leaders are able to optimize within subgroups of robots by collecting information about their tasks and status, and re-allocating the tasks within the subgroup in a more profitable manner. Also, they consider the effects of a leader optimizing a single subgroup, and some effects of multiple leaders optimizing overlapping subgroups.

On the other hand, in (Ko et al., 2003), they make use of dynamic clusters of robots called exploration clusters. Such clusters contain all robots that can communicate with each other and they know their relative locations. A decision-theoretic framework is used to coordinate robots within each exploration cluster.

1.3.4 Comparisons

Comparison between different approaches are difficult since there is no common set of test data. Also, it is not easy to implement distributed algorithms developed by others with the same characteristics; different implementation issues can affect the algorithm performance. However, there are some works that compare different approaches, for example in (Dias and Stenz, 2003) they compare different task allocation methods, both centralized and distributed (behavior and market-based approaches). They conclude that the market-based approach is the best option since it offers a good compromise between communication requirements and the quality of the solution. Also, the market-based approach is compared with the threshold-based approach in (Karla and Martinoli, 2006) and with the token-based approach in (Xu et al., 2006). They conclude that the market-based approach obtains more efficient solutions but it usually needs more communication requirements. Also in (Karla and Martinoli, 2006), it is pointed out that market-based approaches need accurate information about tasks and local states to work properly. When the information is not accurate, it seems that threshold-based approaches offer the same quality of the solution with a fraction of the requirements. Finally, a comparison between different market-based algorithms and different optimization criteria can be consulted in (Mosteo and Montano, 2007). As can be seen, the market-based approach has received a lot of attention, and it is considered a good approach to solve the task allocation problem when explicit communication is possible.

1.4 Fundamentals of market-based approaches

A task allocation algorithm can be seen as a method of distributing common resources. Humans have dealt with similar problems for thousands of years with increasingly sophisticated market economies in which the individual pursuit of profit leads to the redistribution of resources and an efficient production of output. Therefore, market-based approaches make use of the principles of the market economy and apply them to multi-robot coordination. This idea started with the *Contract Net Protocol* or CNP (Smith, 1980) that allocates tasks through negotiation of contracts. In this virtual economy, robots are traders, tasks are traded commodities, and virtual money acts as currency. Robots compete, despite being teammates in reality, to win tasks by participating in auctions that produce efficient distributions based on specified preferences. When the system is appropriately designed, each robot acts to maximize its individual profit and simultaneously improves the efficiency of the team. This is the foundation of the market-based approach success; to engineer the costs, revenues, and auction mechanisms in such a way that individual self-interest leads to globally efficient solutions.

A multi-robot cooperation approach can generally be considered a market-based approach when it satisfies the following requirements (Dias et al., 2006):

- The team is given a number of tasks that are achievable by individuals or subteams. To execute those tasks, the team has at its disposal a limited set of resources (robot capacities) that is distributed among the team members.
- A global objective quantifies the system designer's preferences for all possible solutions.
- An individual utility function specified for each robot quantifies that robot's preferences for its individual resource usage and contributions towards the team objective. Evaluating this function cannot require global or perfect information about the state of the team or team objective.
- A mapping is defined between the team objective function and individual or subteam utilities. This mapping addresses how the individual production and

consumption of resources, and individuals' advancement of the team objective affect the overall solution.

- Resources and individual or subteam objectives can be redistributed using a mechanism such as an auction.

As can be observed, the auction mechanism is the core of the market-based approach. This mechanism can be divided in two phases: bidding phase and winner determination phase. In the former, tasks are evaluated using a utility function which does not require the use of global information. In the latter, after receiving the different bids, a task awarding mechanism is applied in order to choose the most suitable robot for the task under auction. Moreover, these two phases consider the participation of two roles: auctioneer and bidders. The bidding phase starts with a robot "auctioneer" offering a task to the rest of the robot "bidders". After receiving the announcement, they should reply with their bids based on their capacity to execute that task (utility function). The bidding phase finishes when all the bids are received by the auctioneer. Next, the winner determination phase starts; the "auctioneer" applies a mechanism that awards the task to one of the "bidders". Finally, the winner executes the allocated task. It is important to point out that robots are not designed to be auctioneer or bidders, instead those roles are played dynamically. In Figure 1.2, both phases and roles are represented along with the different steps considered in a general task allocation process based on auctions.

Market-based task allocation algorithms do not limit the number of auctioneers and more than one can operate at the same time. Therefore, tasks can be announced by different means: a human operator using a monitoring center (such as a scientist that selects the areas from where he is interested in obtaining data), an autonomous computer (a high level mission planner that generates the different tasks), or the robots themselves that create tasks dynamically.

In summary, the main concepts that define a task allocation mechanism based on auctions are: global objective, utility function and task awarding mechanism.

- The global objective defines which is the team goal to be optimized by the coordination of all the robots. Different global objective functions can be considered

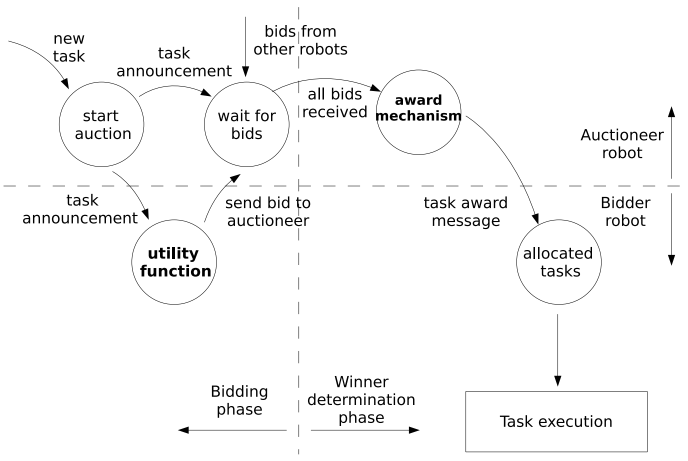


Figure 1.2: Diagram that shows the different states involve in a task allocation process by means of an market-based auction.

(Tovey et al., 2005): the sum of the utilities, the maximum of all the utilities, and the average of the utilities. The sum of utilities is used in scenarios where it is important to minimize the total energy consumed by the team of robots. The maximum of all the utilities is used in scenarios where it is fundamental to minimize the time needed to execute all tasks. Both objectives have been used in multi-robot exploration scenarios. On the other hand, the average of the utilities is used in search-and-rescue scenarios where it is important to minimize how long on average it takes until a task is executed.

- The utility function is used to evaluate tasks and calculate bids. This function is composed of the reward and cost functions, i.e., $U(T_i, R_j) = R(T_i, R_j) - C(T_i, R_j)$. The reward function indicates the benefit of executing a task, and the cost function gives an estimate of the effort to accomplish the same task.

- The most common task awarding mechanism is to allocate a task to the robot with the highest utility or lowest cost considering all the received bids. In this thesis, novel task awarding mechanisms are presented (see Chapter 4).

As was said before, there is a connection between the individual utility function, the task awarding mechanism and the global objective, and it is responsibility of the system's designer to choose a utility function and an awarding mechanism that leads to efficient global solutions. In (Tovey et al., 2005), systematic methods for deriving appropriate utility functions and awarding mechanisms for each of the commented global objectives are explained.

Finally, other properties that allow the characterization of a market-based task allocation algorithm are described:

- Multiple Robots Multiple Tasks (MRMT) and Multiple Robots Single Task (MRST) algorithms: MRST algorithms do not make use of local execution plans, and therefore, they are suited for applications where task costs may change through time. However, they usually obtain less efficient allocations than MRMT algorithms which use local plans to increase the information used in the bid calculation. It can be said that MRST algorithms have a capacity constraint (Koenig et al., 2007) equal to 1, and MRMT algorithms greater than 1.

- With and without reallocations: when reallocations are not considered, tasks are executed by the same robots to which the tasks were intially allocated. On the other hand, when a task allocation algorithm considers reallocations, it means that in order to increase the efficiency of the final allocation, a robot could reannounce its already allocated task or tasks.

- Combinatorial or single-item auctions: in most of the task allocation algorithms, each auction process only considers a single task. In combinatorial auctions, each auction can involve more than one task. Therefore, bids are calculated for bundles of tasks (Zheng et al., 2006).

- Coordinated or loosely coupled tasks: when the execution of tasks is completely independent from the rest, tasks can be defined as loosely coupled. However, if the execution of tasks depends on others, tasks are defined as coordinated. This fact should be taken into account in the task allocation algorithm in order to avoid execution deadlocks (see Chapter 2).

- Sequential and parallel auctions: when only one auction is run at a time, the task allocation algorithm is defined as sequential. On the other hand, if more than one auction can be performed together, they are executed in parallel. When parallel auctions are used, the system's designer should be careful since bids used in one auction process can be no longer valid due to the result of another parallel auction.

1.5 Previous work on market-based approaches

Market-based approaches have received significant attention in the last decade, mainly because they offer a good compromise between communication requirements and the quality of the allocation. In Section 1.3.2 only a brief summary of this approach was described. However, in this section, a detail summary of the main research works related to market-based algorithms is explained.

Likely, the first distributed market-based system was M+ (Botelho and Alami, 1999), defined within a general architecture for the cooperation among multiple robots (Botelho and Alami, 2001). In this system when a robot calculates the task cost, it considers the next one in order to increase the efficiency of the solution. A similar system is MURDOCH ((Gerkey and Matarić, 2000) and (Gerkey and Matarić, 2002)). But in this case tasks are only allocated to robots that are idle, i.e., during the execution of a task, robots do not take part in any auction. Therefore, the mechanism of task allocation is based on a purely greedy method (the best solution at a certain time only considering the state of the system in that moment). The results obtained with this algorithm are less efficient than M+, because it does not consider future tasks that robots plan to execute. However, the main advantage of this method is that it is very simple and needs few resources.

TraderBots (Dias, 2004) is a distributed system whose main contribution is the consideration of dynamic environments with partial failures of the robots and communications (Dias et al., 2004), while obtaining efficient solutions. Unlike MURDOCH, robots have a local plan, and more than one task can be allocated to each one. Moreover, its efficiency is improved (Dias and Stenz, 2002) using two different techniques: the use of leaders in clusters of robots that allows negotiations among more than two robots at the same time, and the capacity to negotiate a group of tasks. After showing various simulation results using these techniques, it is concluded that a greater efficiency is achieved when it is possible to negotiate more than one task at a time which is usually known as combinatorial auctions.

There are a number of works that have focused on combinatorial auctions. In (M. et al., 2003), they propose different combinatorial bidding strategies, and compare their performance with each other as well as to single-item auctions and an optimal centralized mechanism. Their experimental results show a substantial advantage of combinatorial auctions over single-item auctions. A blend of combinatorial and sequential single-item auctions is developed in (Koenig et al., 2007). In order to reduce the team cost of sequential single-item auction algorithms, they generalize them and assign more than one task during each round, which increases their similarity to combinatorial auction algorithms. They show that, for a given number of additional tasks to be assigned during each round, every robot needs to submit only a constant number of bids per round and the runtime of the task awarding mechanism is linear in the number of robots. Finally, in (Lin and Zheng, 2005), it is presented a combinatorial bid-based multi-robot task allocation method. The proposed method provides an explicit cooperation mechanism to the bidding robots, so that they can form a group of robots to bid for complex tasks.

On the other hand, another interesting work is the so called *Distributed and Efficient MultiRobot Cooperation Framework* (DEMIR-CF) (Sariel and Balch, 2006a), (Sariel and Balch, 2006b) and (Sariel et al., 2006a). In this case, the assignment of tasks is done incrementally, so robots do not have a local plan. However, they maintain a list of possible tasks to be executed from which the next task to be announced is selected. This list of tasks is obtained based on heuristic algorithms. They also consider tasks that need more than one robot to be executed. To solve this problem, they create dynamic coalitions with a leader, and it is the leader who is in charge of selecting which robots will form the coalition. Also, the leaders release robots from the coalition when it is necessary. Finally, precaution routines are used to prevent inconsistencies in the system due to robot or communication failures (Sariel et al., 2006b). This framework has been successfully applied to naval mine countermeasure missions (Sariel et al., 2008).

A number of market-based works have studied specifically task allocation problems that consider tight relations or restrictions between tasks. In (Kalra et al., 2005; Kalra et al., 2007) a market-based framework, called Hoplites, is explained where passive and active coordinations are taken into account. In simple situations, the team can complete tasks faster by utilizing the passive coordination mechanism which facilitates more local decision making processes and is light on both computation and communication. More complex scenarios require more complex interaction between teammates. This is provided by the active coordination mechanism which enables robots to actively influence each other's actions over the market. The coordination strategy is adapted to the changing demands of the task. In (Lemaire et al., 2004), they deal with a relaxed problem of temporal constraints. Tasks may be numerically partially ordered, i.e., the constraint between T1 and T2 should be of the type "T1 n seconds before T2". These restrictions are solved by means of master/slave relations between robots. Also, market-based algorithms that allocate tasks while preserving the communication links between robots are explained in (Mosteo et al., 2008). Their work guarantees mission completion in open spaces and is customizable by means of swappable algorithms in order to optimize preferred performance metrics. Their findings show that effective multi-robot routing can be achieved even under limited communication ranges with moderate loss compared to the case of infinite communication ranges. A different type of relation between tasks is studied in (Zlot and Stentz, 2006) where task decomposition and allocation phases are considered together instead of separated as usual. This combination of both phases has the aim to increase the execution efficiency of complex missions.

Finally, there are other research works that develop interesting ideas. In (Golfarelli et al., 1997) a market system, based on the exchange of tasks, is presented. The main advantage to this approach is that it is not necessary to have a common metric among the values of the different robots' cost functions. This allows robots to calculate the cost of each task based on their own parameters, and it is not necessary to normalize them with the rest of robots. Usually, cost and utility functions are simple enough that are easy to normalize with other robots. However, if very different tasks were considered, the normalization step would not be so easy, and the need to have a common metric could be a problem. The objection of this system is the need of large number of interactions to achieve a positive exchange of tasks. Also, in (Guerrero and Oliver, 2004) market and threshold approaches have been combined to determine the optimal number of robots needed to solve specific tasks. And in (Nanjanath and Gini, 2006), it is presented some variations of the CNP protocol in such a way that tasks can be completed even if mobile objects hinder the execution of tasks. This system is based on the reallocation concept and, due to the dynamism of the environment, every time a robot finishes one task, it announces again all the planned tasks that are not in execution.

1.6 Limiting the scope

In this section, the characteristics and assumptions considered in this thesis for the multi-robot system and the task allocation problem are explained.

1.6.1 Multi-robot system

First, the multi-robot system considered in this thesis is classified in the system dimensions, using the taxonomy presented in (Dudek et al., 2002), as:

- Collective size: the number of robots considered is limited, although the number of robots could be in the range of tens of robots (SITE-LIM).
- Communication topology: robots are linked in a general graph (TOP-GRAPH).
- Communication bandwidth: the cost of communication is negligible compared to other costs (BAND-INF).
- Collective reconfigurability: the relationship among robots can be reconfigured dynamically (ARR-DYN).

- Processing ability: each robot can be thought of as Turing Machine Equivalent (PROC-TME).
- Collective composition: the algorithms explained in this thesis have been mainly tested in systems where robots have the same behavioral level (CMP-HOM). However, thanks to the use of a layered multi-robot architecture (see Appendix A), there is no reason why these algorithms cannot be applied to a heterogeneous team of robots.
- Communication range: initially it is assumed that robots can communicate with any other robot (COM-INF). However, as it is explained later, the algorithms presented in this thesis work with limited communication ranges. Although, as a logical consequence, the quality of the solutions decreases.

On the other hand, the considered multi-robot system can be classified in the coordination dimensions, using the classification stated in (Farinelli et al., 2004), as:

- Cooperation level: the system is considered cooperative since it is assumed that robots operate together for the benefit of the team. Situations where robots aim to behave against the global goal deliberately are not considered.
- Knowledge level: the system can be considered as aware since robots have some knowledge of their teammates.
- Coordination level: the actions performed by each robot takes into account the actions executed by the other robots as a result of the coordination protocol. Therefore, it can be said that the system has a strong coordination level.
- Organization level: the system does not consider the existence of leaders and there is no central element. Hence, the system can be considered distributed since robots do not have access to all the information constantly. However, they communicate between each other occasionally to transmit the information needed in the auction process (bids, task awarded messages, etc.).

Multi-robot system characteristics, such as the communication range and the organization level, have an impact on the task allocation algorithm implementation.

When the multi-robot system is distributed, there are different possible implementations of the task allocation algorithm. A parallel market-based implementation, such as (Bertsekas, 1981), that supposes that all the information is accessible to all the robots. As a result, every robot has an infinity communication range. This implementation usually obtains the optimal solution, but they use a large number of messages. The other possibility is a fully distributed implementation, for example (Gerkey and Matarić, 2002), where only local information is used for the auction process (limited communication ranges), and therefore, the number of messages is much smaller. However, they do not usually obtain optimal solutions. In most of the cases, the efficiency of the solution (closeness to the optimal solution) depends on the information accessibility.

In this thesis, task allocation algorithms are distributed in the sense that robots do not have access to all the system information. Therefore, bids are calculated using only the information of the task under auction and the robot state. However, task allocation algorithms use inter-robot communication to transmit their bids and allocate the different tasks. These facts enable the use of the algorithms in non-static scenarios where: tasks are created dynamically, auctioneers do not need to know how many bidders exist and the number of bidders can change through time. It is important to point out that these algorithms can work with limited communication ranges. But, as a logical consequence, the quality of the solution depends on the robot communication ranges since more or less information is used in the allocation process.

1.6.2 Multi-robot task allocation problem

First, it is important to define what it is understood as an allocation. An allocation of a set of tasks $T = \{T_1, T_2, \ldots, T_t\}$, among a set of robots $R = \{R_1, R_2, \ldots, R_r\}$, is a partitioning of the set T among the robots. This can be mathematically denoted by a tuple $\langle T^1, \ldots, T^j, \ldots, T^r \rangle$ (Zlot, 2006) where:

- Each component of the tuple represents the tasks assigned to a robot, i.e., robot R_j is assigned a number of tasks represented by $T^j = T_a, T_b, \ldots, T_x$ which is a subset of the total number of tasks.

- The union of all the components of the tuple is equal to the complete set of tasks, i.e., $\bigcup_{i=1}^{r} T^i = T$.

- The intersection of any two components of the tuple is equal to the empty set, i.e., $T^i \bigcap T^j = \emptyset$ for all $i \neq j$.

A formal definition of the multi-robot task allocation (MRTA) problem can be stated as follows
Given a set of tasks, $T = \{T_1, T_2, \ldots, T_t\}$, a set of robots $R = \{R_1, R_2, \ldots, R_r\}$ and a function $U(T^j, R_j)$ that specifies the utility of executing a subset of tasks T^j by robot R_j, find the allocation that assigns tasks to robots and try to optimize a global objective.

In order to define completely the MRTA problem, it is necessary to specify the utility function and the global objective that needs to be optimized. As was explained before, the utility is composed of the reward and cost functions.

$$U(T^j, R_j) = R(T^j, R_j) - C(T^j, R_j)$$

The reward function indicates the benefit of executing a task, and the cost function gives an estimate of the effort to accomplish the same task. In this thesis, it is not considered rewards associated to tasks, so the utility functions are equal to the cost of the tasks. Moreover, this thesis is applied to exploration scenarios, where tasks are usually waypoint tasks. And costs are defined as a quantity that reflects how much it will cost a robot to go to a certain location, such as the traveled euclidean distance or the traversability index (Howard et al., 2005). This fact does not mean that the algorithms presented in this thesis could not be used with other costs functions, such as energy or time, and with other types of tasks.

On the other hand, it is important to define the global objective function. Different global objective functions (Tovey et al., 2005) can be considered such as: the sum of the costs, the maximum of all the costs, and the average of the costs. Since this thesis is focused on exploration scenarios, the proposed metric is the sum of all the costs. This metric gives an estimation of the total energy consumed by the team of robots which is an important parameter for studying the performance of task allocation algorithms in this type of scenarios. Therefore, the global objective of the task allocation algorithm is to minimize the sum of the costs. An important term that is used in the following chapters is the global cost, which is defined as the sum of the allocated task costs. Therefore, the global objective used in this thesis can be redefined as the minimization of the global cost.

Finally, the version of the MRTA problem that is used throughout the thesis can be stated as
Given a set of tasks, $\{T_1, T_2, ..., T_t\}$, a set of robots $\{R_1, R_2, ..., R_r\}$, and a function $C(T^j, R_j)$ that specifies the cost of executing the subset of tasks T^j by robot R_j, find the assignment that allocates tasks to robots and try to minimize the global cost defined as $\sum_{j=1}^{R} C(T^j, R_j)$, where the subset of tasks $T^j = T_a, T_b, \ldots, T_x$ is assigned to robot R_j.

1.7 Thesis outline

The rest of the thesis is organized as follows. The next chapter deals with tasks that need an unknown number of robots to be executed. The number of robots depends on their capacities and it is calculated during the negotiation process. For this purpose, the service concept is introduced. A service is generated from a task to which is related. Moreover, a service can generate another service creating a hierarchical relation that gives name to this algorithm. The hierarchical relation between tasks and services has an impact on the task execution which is also studied.

Chapter 3 presents a market-based algorithm that is suited for cases where the efficiency of the allocation algorithm is the primary issue. The allocation performance is increased combining the concepts of reallocation and combinatorial auctions. Tasks are grouped and allocated together instead of using single-item tasks. The efficiency of the allocation is increased since robots calculate their costs with a longer execution horizon and take advantage of the synergies between tasks. However, a high confidence in the task costs is needed. Finally, synchronization issues regarding the MRMT and MRST algorithms are studied.

In Chapter 4, different MRST task allocation algorithms are described. This type of algorithm does not make use of local execution plans, and therefore, they are suited for applications where task cost may change through time. These algorithms are also based on the market approach, but different strategies are used to increase the efficiency of the solutions. Finally, these algorithms are implemented within a robot architecture that considers path planning and obstacle avoidance.

Chapter 5 presents a probabilistic analysis that is used to compare different MRST market-based task allocation algorithms. The objective of the analysis is the calculation of the expected value of the objective function. This metric gives us an idea of the algorithm performance over time. The obtained theoretical results are validated through simulations and experiments with real robots. Also, a performance study is carried out where the algorithm solutions are compared with the optimum. Finally, the analysis is extended to calculate not only the expected value, but also, the complete probabilistic distribution that models the objective function.

The thesis is completed with Chapter 6, which discusses and concludes the results and in which the future work is summarized.

1.8 Main contributions

This thesis makes the following primary contributions to the study of the multi-robot task allocation problem using distributed market-based algorithms:

1. Regarding MRMT market-based algorithms, a new concept, called services (Viguria et al., 2008) and (Viguria et al., 2010), has been introduced in order to allocate tasks that need an undetermined number of robots to be executed. The precise number of robots is decided during the negotiation process and depends on the robot capacities. Moreover, reallocations and combinatorial auctions

have been combined in order to obtain a distributed task allocation algorithm (Viguria et al., 2007) that presents high performance and fault tolerance properties.

2. Novel ideas have been applied to MRST task allocation algorithms in order to improve their performance (Viguria and Howard, 2009b) in comparison with the standard market-based implementation. The main difference is a small increase in the shared information used to model the level of relevance for a specific task. Using this extra piece of information, robots choose tasks that are better not just for themselves, but also for the benefit of the group.

3. A complete new approach, based on a probabilistic analysis (Viguria and Howard, 2007) and (Viguria and Howard, 2009a), is used to calculate performance metrics for MRST market-based algorithms. This analysis consists of calculating the expected value of the objective function which is later used as a descriptor to compare different algorithms (Viguria and Howard, 2009c). Since the expected value is a measure of the behavior of the algorithm "in average", it becomes a more informative descriptor of the performance of an algorithm than the typical measure carried out in worst case analyses, which only provides a comparison of "bad"-performance with respect to the optimal solution.

Chapter 2

Hierarchical market-based algorithm

In this chapter, it is presented a distributed market-based algorithm called S+T, which solves the multi-robot task allocation (MRTA) problem in applications that require the cooperation among the different robots to accomplish all the tasks. This algorithm deals with tasks that need an apriori unknown number of robots to be executed. The number of robots depends on their capacities and it is calculated during the negotiation process. For this purpose the service concept is introduced. If a robot cannot execute a task by itself, it asks for help and, if possible, another robot will provide the required service. Moreover, a service can generate another service creating a hierarchical relation that gives name to this algorithm. The hierarchical relation between tasks and services has an impact on the task execution which is also studied.

First, the S+T algorithm is described and illustrated with a simple example. In the same section, the changes on the costs that allows the algorithm to prioritize between the execution time and the energy spent on the mission is also explained. Next, the deadlock problem is stated, and a distributed algorithm that solves it is commented. Finally, simulation results that illustrate the main characteristics of the S+T algorithm are shown.

2.1 Introduction

Usually market-based approaches assume that each task can be executed completely by a single robot. But this could not be the case for example in a surveillance or exploration scenario, in which a task consisting in transmitting images in real-time could require another robot to act as a communication relay. The approach to solve this problem is based on the concept of service. If a robot cannot execute a task by itself, it asks for help and, if possible, another robot will provide the required service. Required services are generated dynamically and are necessary to successfully complete their associated tasks. Other possible scenarios, where this approach is useful, could be the box-pushing problem and the cooperation among various robotic arms. In the first one, assuming that the weight of the box is known and how much weight a robot can push, one or more services could be required until the pushing capacity of the team of robots is equal or greater that the weight of the box. In the second scenario, it is assumed that we have several robotic arms with a limited set of tools and some overlapping of their workspaces. When a robot has to perform a task, it will need a group of tools. If these tools are not within its workspace, the robot will ask for a service to get the desired tool from another robot.

It is widely accepted that one of the main advantages of multi-robot systems with respect to a stand-alone robot is their capability to perform tasks that can be impossible for a single robot. The new task allocation protocol, called S+T, is designed to exploit this characteristic. The basic idea is that a robot can ask for services when it cannot execute a task by itself. The cost of the task will be the sum of the costs of the task and the service or services required.

Finally, using the concepts from the MRTA formal taxonomy explained in (Gerkey and Matarić, 2004), the S+T algorithm can be posed as an instance of the ST-MR-TA case:

- Single-task robots (ST): each robot is capable of executing at most one task at a time.
- Multiple-robot tasks (MR): each task may require more than one robot to be executed.

- Time-extended assignment (TA): tasks are not allocated instantaneously, and for some of the algorithms, more information is available to make the decision.

2.2 S+T: Services and tasks algorithm

As any other market-based algorithm, there are two roles (bidders and auctioneer) that are played dynamically by the robots. The algorithms associated to each role are detailed in Algorithms 1 and 2. In the bidding process, when a robot needs a service to execute a given task, it will bid initially with just the cost of the task (because it still does not know the cost of the required services) labelling the message to the auctioneer as "provisional". The auctioneer will evaluate all the bids, and if the best bid requiring a service is better than the best bid without the need of a service, the robot requiring the service will start another auction in order to find which robots can perform that service. When this second auction is finished, the robot will send to the auctioneer the complete cost of the task, including the cost of the associated services. Afterwards, the auctioneer will decide which robot executes the task based on the updated costs. If a task is allocated to a robot requiring a service, that service will be also allocated at the same time.

It should be pointed out that both the algorithms, used to allocate services and tasks, are based on the SIT-MASR algorithm presented in (Viguria et al., 2007). The only differences are:

- Services cannot be reallocated dynamically.
- When a robot that will execute a service changes its local plan, it has to report the new cost of the service to the robot which required it. So, it can start another auction to check if a different robot has a lower cost for that task.

A relevant feature of the algorithm is that services can be allocated recursively, i.e., a robot that executes a service could also require another service to accomplish the first one, and in this way to any number of recursive services. Therefore, the algorithm takes full advantage of the possibilities that a team of robots can offer (it is even possible to execute missions with a task involving the whole team).

Algorithm 1 S+T auctioneer algorithm

```
if there is any task to announce then
    announce task
    while timer is running do
        receive bids
    end while
    calculate best bid (lowest cost)
    if best bid is lower than the auctioneer bid then
        if best bid requires a service then
            allow robot to start a new auction in order to find a robot who can execute
            that service
        end if
        wait until the second auction is finished and the total cost of the task (including
        the service cost) is sent
        send task to best bidder taking into account the updated bids
    end if
    delete task from announcement list
    if task has an associated service then
        send a message to the robot that will execute the service in order to delete it
        from its local plan
    end if
end if
```

Algorithm 2 S+T bidder algorithm

```
a new message is received
if new message is a task announcement then
    compute the optimal insertion point for the task in the local plan
    calculate bid (marginal cost)
    if the task requires a service then
        send initial bid to the auctioneer and indicate that a service is needed
    else
        send bid to the auctioneer
    end if
else if new message allows to ask for a service then
    start a new auction in order to find a robot that can execute the service
    receive all the bids for the service
    calculate the complete cost for the task including the cost for the service
    send the new cost to the auctioneer
else if new message is a task award then
    insert task in the local plan in the position calculated before
    add task in the announcement list
    if the task needs a service, allocate the service to the robot that won the auction
    if the cost of any allocated service (in case it exists) has changed because of the
    insertion of the new task in the local plan then
        send the new cost of the service to the robot with the task
    end if
end if
```

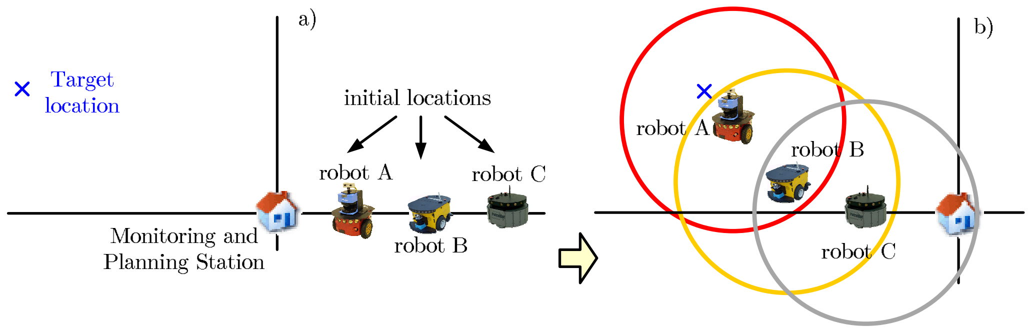


Figure 2.1: Example of multiple recursive services required to accomplish one task. Figure a) shows the initial positions of the robots and the base station and b) shows the final assignment of tasks and services that allows robot A to transmit images to the base station using robots B and C as communication relays.

In order to illustrate this characteristic, a surveillance mission will be considered. The mission consists in transmitting information from a certain area to a base station in real-time. The robot has to be within the communication range of the base or in the range of another robot acting as a communication relay. As it can be seen in Figure 2.1, the transmission to the base requires two robots acting as communication relays. The most relevant messages involved in the negotiation process are represented in the diagram depicted in Figure 2.2.

It should be pointed out that when a robot announces a service required for a certain task, the robot that will execute that task cannot take part in the auction process for the service.

The use of services increments the cooperation among robots and allows to achieve missions that could be impossible using a regular task allocation algorithm. For example, transmitting images in a surveillance mission from a position that does not have direct coverage with the base of operations. However, services can also increment the total time of the mission since more than one robot could be used to execute one task and, therefore, less tasks can be executed "in parallel". In this context, if a robot can execute a task by itself with a higher cost than another robot using services, it should be decided which option is better. The answer to this question depends on the specific application and two different approaches have been developed to tackle with different scenarios:

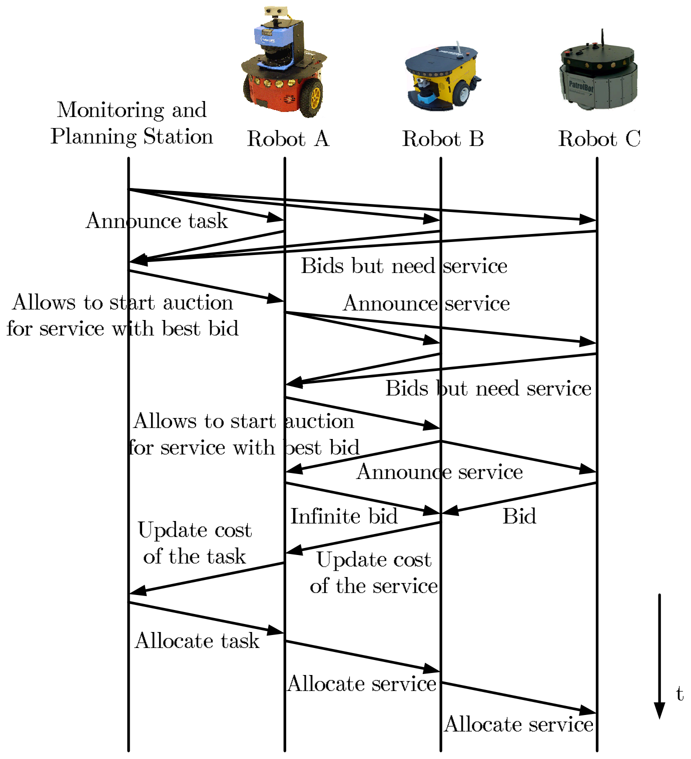


Figure 2.2: Messages interchanged in the negotiation process using the S+T algorithm for the example illustrated in Figure 2.1 (one task requiring two services to be executed). When several robots ask for a service, only the robot with the lowest bid is allowed to start an auction for the service. For example, robot C asks for a service twice but it is never allowed to start an auction, because the negotiation is over once robot A can execute the task using the communication relay services from robots B and C.

- In the first approach, tasks have a higher priority than services, and therefore, it should be applied to scenarios where the goal is to minimize the total execution time of the mission. Basically, when an auctioneer receives bids from robots and, at least one of them does not require a service, the task will be directly allocated to it. This approach also needs less communication messages since services will be only considered when they are totally necessary for the success of the mission.

- In the second approach, the priority between the total time of the mission and the energy consumed by the team can be adjusted with a parameter α defined as follows:
$$\alpha = \frac{P}{1-P}$$
where $P \in [0,1]$ is the priority to minimize the total time of the mission. This parameter is used in the computation of the cost for the service:
$$C_s = C_o \cdot (1 + \alpha \cdot L)$$
where C_o is the original cost of the service, C_s is the new cost of the service and L is the level of the service, i.e., if it is the first service that depends on a task, L is equal to 1. If it is a service that depends on the first service, then L is equal to 2 and so on. This second parameter is used to penalize the use of more than one robot to execute one task. Moreover, when the use of services is unavoidable, L allows to increase the priority of services that need less robots.

  The value of the parameter P should be selected depending on the type of mission. If it is more important to minimize the energy spent on the mission and the total time is not so important, $P = 0$ should be selected, which means $\alpha = 0$. On the other hand, if it is important to minimize the total time of the mission without considering a complete execution of all the tasks, $P = 1$ should be selected which means $\alpha \to \infty$. In this case, services will not be considered and the algorithm will behave as the SIT-MASR market-based algorithm with local plans and reallocations (Viguria et al., 2007).

2.2.1 Deadlock situations

In the previous section, the allocation process of tasks and services has been presented, but not the synchronization issues related with the relation between tasks and services during the execution. From a general point of view, when the execution of tasks depends on others, the generation of deadlocks should be considered, and even more when the process is distributed. It has been noticed in simulation that this problem appeared frequently since each robot only has local information and there is no direct way to know if its particular local plan will generate a deadlock in all the tasks and services executed by the team of robots. For example, in Figure 2.3, it is shown how an execution loop can be generated using the S+T algorithm for a particular example with data transmission tasks and communication relay services.

This problem has not an easy solution since robots only have knowledge of their own plans. It is also important to find an algorithm to solve this problem in a distributed way since the key idea is to have a whole functional robotic system that works without the presence of a centralized entity. The presented solution is based on the use of “check loop” messages, i.e., every time a robot wins a task, it will broadcast a message indicating the service associated to the new task (in case it exists). The robot which has won that service will process the message and will send a message for every task or service that appears in its local plan before the mentioned service and has also a service associated to it. As it is shown in Figure 2.4, when a robot receives back a “check loop” message with its id, it will sell the task that provokes the loop and it will introduce it in a black list in order to avoid biding again for it. The use of a black list has the purpose to prevent the generation of allocation loops when the best two robots for a task are involved in an execution loop when they integrate the task in their local plans (i.e., they start to reallocate the task to each other and in both cases an execution loop is formed). Finally, the complete algorithm is shown in Algorithm 3.

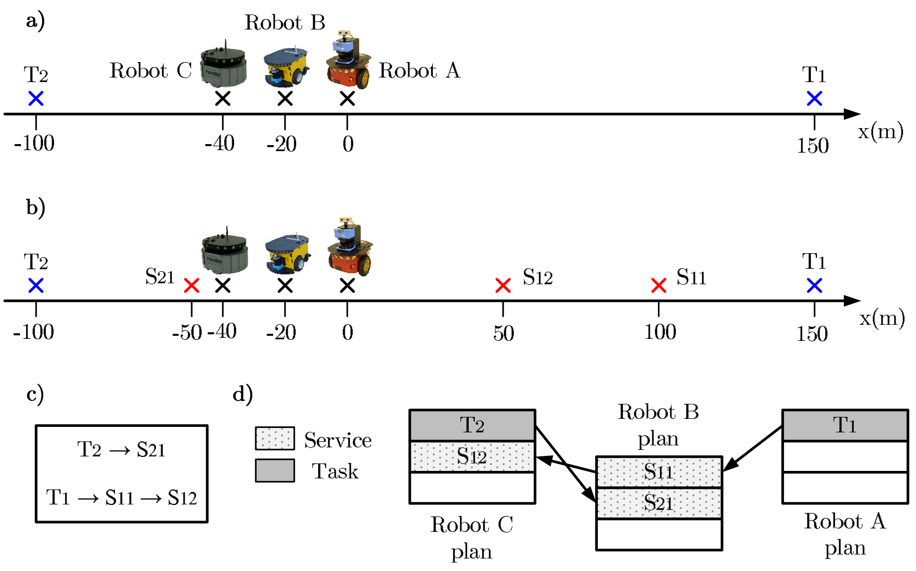


Figure 2.3: Example where a deadlock is generated since the execution of the tasks depends on the execution of services. Figure a) shows the initial position of the robots and the tasks to be allocated. Those tasks are represented by cross marks and consist in visiting target locations to transmit images to a monitoring station located in $x = 0$. Assuming a radius of communication of 50 meters, Figure b) shows the services (also represented by cross marks) needed to execute the tasks. Figure c) shows the relation of execution between the tasks and the services, and Figure d) presents the relation in terms of execution in the final allocation using the S+T algorithm along with the plan of each robot (with the order of execution of its tasks and services). T_i represents the task with identifier i and S_{jk} means a service associated to the task j with level k.

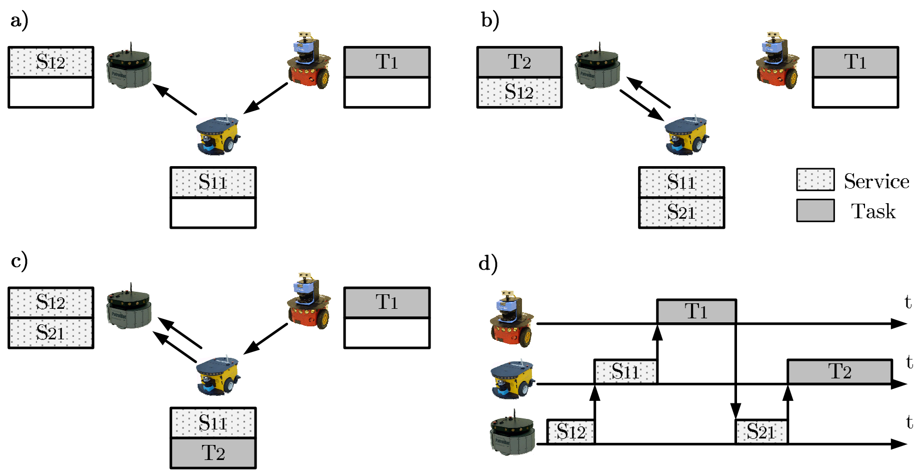


Figure 2.4: Considering the initial configuration presented in Figure 2.3, Figure a) shows the allocation after the announcement of the first task and the path followed by the "check loop" messages. Figure b) presents the allocation of the second task and the path of the "check loop" messages that detects the execution loop. Figure c) shows the allocation after the reallocation of the task and how the execution loop has been removed. Figure d) presents the final execution sequence of the different tasks and services with one timeline per robot (the arrows represent the required synchronization during the distributed execution).

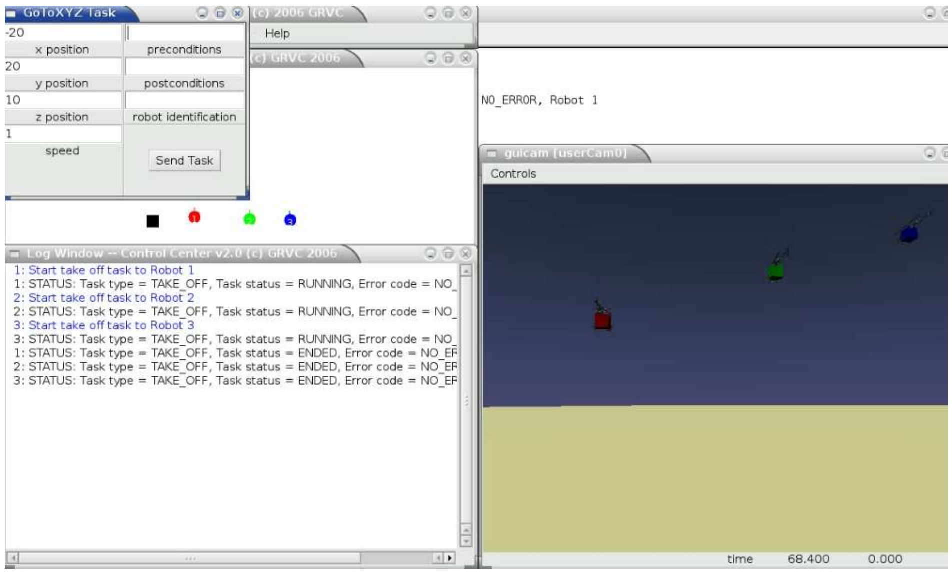
Algorithm 3 Distributed loop detection algorithm

```
wait until receive a “check loop” message with a task or service that the robot has
in the local plan
if id message == robot id then
  if task has an associated service then
    send a cancel service message
  end if
  delete task from won-tasks list (loop detected)
  insert task in black-tasks list
  insert task in announcement-tasks list
else
  move to the initial position of the local plan
  repeat
    if task or service has a service associated to it then
      send “check loop” message
    end if
    next task or service in the local plan
  until task or service != task received in the “check loop” message
end if
```

2.3 Simulation results

In these simulations, surveillance tasks where robots have to send back images in real-time to a base station from a certain point were considered. Therefore, a robot transmitting images have to be within the communication range of the base station using its own communication device or using one or more robots as communication relays (services). For this particular scenario, the execution synchronization between tasks and services has been implemented using preconditions, i.e., a task cannot start until all the services associated to it have been executed. Moreover, the robot or robots that execute a service cannot start the next task or service in their local plan until the associated services have been completed.

Numerous simulations with different number of robots were performed for the surveillance missions mentioned above with several communication range values in a scenario of 1000x1000 meters. In Figure 2.6, it can be observed that the total distance traveled by all the robots decreases when the communication range increases as far as the probability to require a service decreases. The total distance traveled



-20
x position
20
y position
10
z position
1
speed
preconditions
postconditions
robot identification

NO_ERROR, Robot 1

```
1: Start take off task to Robot 1
1: STATUS: Task type = TAKE_OFF, Task status = RUNNING, Error code = NO_
2: Start take off task to Robot 2
2: STATUS: Task type = TAKE_OFF, Task status = RUNNING, Error code = NO_
3: Start take off task to Robot 3
3: STATUS: Task type = TAKE_OFF, Task status = RUNNING, Error code = NO_
1: STATUS: Task type = TAKE_OFF, Task status = ENDED, Error code = NO_ER
2: STATUS: Task type = TAKE_OFF, Task status = ENDED, Error code = NO_ER
3: STATUS: Task type = TAKE_OFF, Task status = ENDED, Error code = NO_ER
```

time 68.400 0.000

Figure 2.5: Simulation environment and interface used to simulate the S+T algorithm with surveillance tasks where robots have to send back images from an area.

by all the robots is considered as a good measurement of the energy spent during the mission. Moreover, the mean of the total distance traveled decreases when the number of robots increases due to the fact that a constant number of tasks is used in all the missions.

Table 2.1 shows the resulting mean values of some parameters in missions with five tasks, different number of robots and values for the communication range. The number of services executed increases when the communication range of the robots decreases and, as a logical consequence, the number of messages received by one robot and the total distance traveled by all of them also increases, as it was mentioned above. This means that the communication requirements and the energy needed to execute the mission will be higher when the number of services increases.

On the other hand, simulations have been run with different values of the α parameter that depends on $P \in [0, 1]$ (see Section 2.2). As it can be seen in Figure 2.7, one hundred random simulations have been executed for different values of P. $P = 0$ is an extreme value applied when the user wants to minimize the total distance traveled by all the robots in the mission in terms of energy, and therefore, the cost of

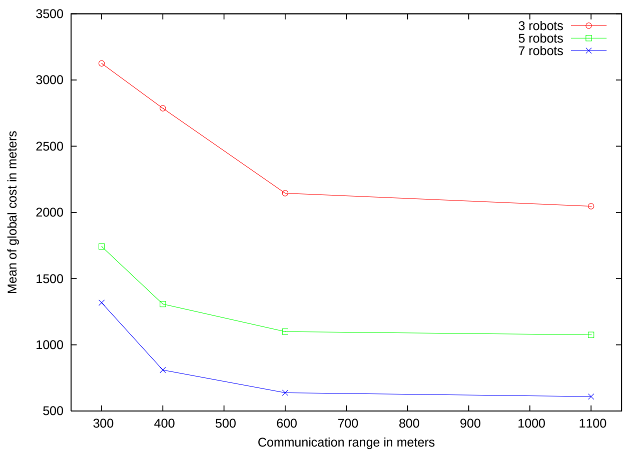


Figure 2.6: Mean of the total distance traveled by all the robots over one hundred missions with different communication ranges, number of robots and five tasks.

Robots	**Comm. range (m)**	**Total distance (m)**	**Messages received**	**Services**
	600	2145.15	47.96	0.56
3	400	2786.52	80.32	2.44
	300	3125.23	150.45	4.36
	1100	1075.23	48.06	0.0
5	600	1099.43	52.3	0.30
	400	1307.97	85.66	1.36
	300	1742.34	164.87	3.45
	1100	609.14	45.06	0.0
7	600	638.42	45.8	0.24
	400	810.23	79.76	1.24
	300	1318.31	142.96	2.76

Table 2.1: Results with five tasks, different number of robots and values for the communication range. The mean of the values from one hundred random missions are shown where total distance means the distance traveled by all the robots, messages received is the number of messages received by one robot in the S+T algorithm and number of services is related to the ones executed by one robot.

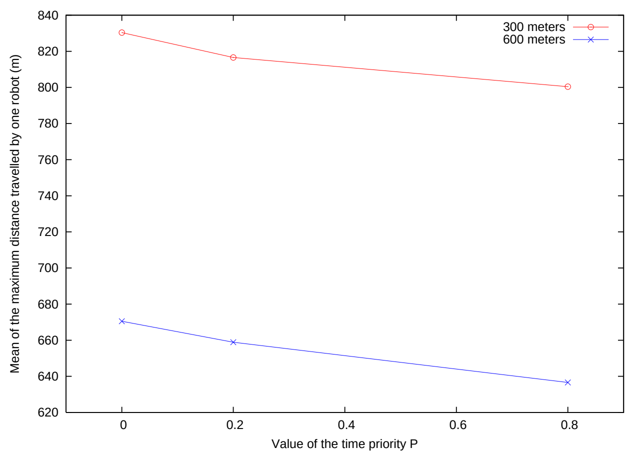


Figure 2.7: Mean of the maximum distance traveled by one robot over one hundred missions with $300m$ and $600m$ as the communication range and five number of robots and tasks.

the services is not modified. Also in Figure 2.7, it can be observed how the maximum distance traveled by one robot decreases when P increases, and therefore, the time of the mission will be smaller (assuming that all the robots move at the same speed) because of the penalization of the costs associated to the services. However, if the execution time is critical, with $P = 1.0$ the S+T algorithm services are not considered and some tasks could be undone (mission partially accomplished). In Figure 2.8, it is shown the mean of the number of tasks executed over 100 missions with different values for the communication range and with $P = 1.0$. Up to six hundreds meters, it can be seen that a significant number of tasks cannot be accomplished for the group of robots if the use of services is not considered. Therefore, it is important to be careful when the parameter P is equal to 1.0 and a given mission needs services to execute most of the tasks. In that case, the time of the mission will be minimized but many tasks will not be executed. Then, it is advisable to only use $P = 1.0$ when most of the tasks can be executed without services and the execution time of the mission is very critical.

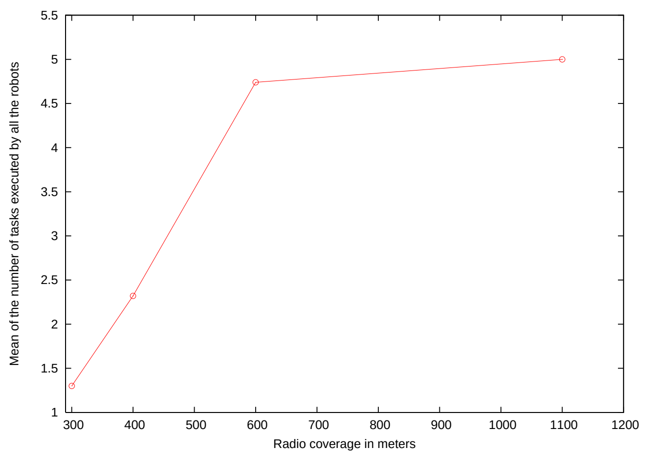


Figure 2.8: Mean of the number of tasks executed by all the robots over one hundred missions with different values of the communication range and five number of robots and tasks. The use of services are not considered in this simulations, i.e., $P = 1.0$ or $\alpha \to \infty$.

2.4 Experimental demonstration

A demonstration was conducted in the "Alamillo" park in the city of Seville, in cooperation with the University of Malaga [1]. Three different robots were involved: the autonomous ground vehicle ROMEO-4R developed by the GRVC at the University of Seville, provided with a trailer for helicopter take-off and landing, the helicopter HERO2 also developed by GRVC, and a mobile fire extinguisher unit. This unit was the all terrain tracked robot AURIGA developed by the University of Malaga which was provided with a conventional fire extinguisher. The area considered for the demonstration was around one square kilometer.

The demonstration was performed to be significant for many disaster management activities, with a team of robots performing the following activities: detection, confirmation, localization, monitoring and actuation. In this demonstration, the S+T algorithm was used with $\alpha = 0$ since the energy of the robots is important in disaster scenarios where robots should be operative the maximum possible time.

[1] I would like to thank University of Malaga for their support and help in this demonstration, specially to Alfonso J. Garcia-Cerezo.

2.4.1 Tasks and services considered in the experiments

The following subset of tasks was selected for the field experiments:

- `Go-to(P)` tasks: To visit a point `P` given by its GPS coordinates.
- `Survey-area(A,object)` tasks: To cover an area `A` given by a convex polygon searching for objects of interest. The local planner of the robot computes a sequence of way-points to cover the area of interest easily and efficiently by back and forth motion along rows perpendicular to the sweep direction sending images to the Alarm Monitoring Station and performing autonomous detection. The task finishes when the object is detected.
- `Extinguish(P)` tasks: To locate, point and activate a fire extinguisher attached to the robot in order to extinguish a fire around GPS coordinates `P`.
- `Monitor(object,final_state)` tasks: To monitor an `object` until its state changes to `final_state`.

On the other hand, the services considered were:

- `Transport(P)` services: Some robots (for example Romeo-4R in the experiment described in Section 2.4) are equipped with platforms allowing aerial robots to be transported from an initial location to a point `P`.
- `Communication-relay (CRP)` services: The Alarm Monitoring Station should receive images from the area during the execution of a `survey-area` task. So if the communication range does not allow this link, another robot (or a chain of robots) should provide the `communication-relay` service moving to a certain point `CRP`.

2.4.2 Description of the demonstration in the "Alamillo" park

The goal of the mission was to detect a fire and extinguish it with the collaboration between the three robots mentioned above (ROMEO-4R, HERO2 and AURIGA).

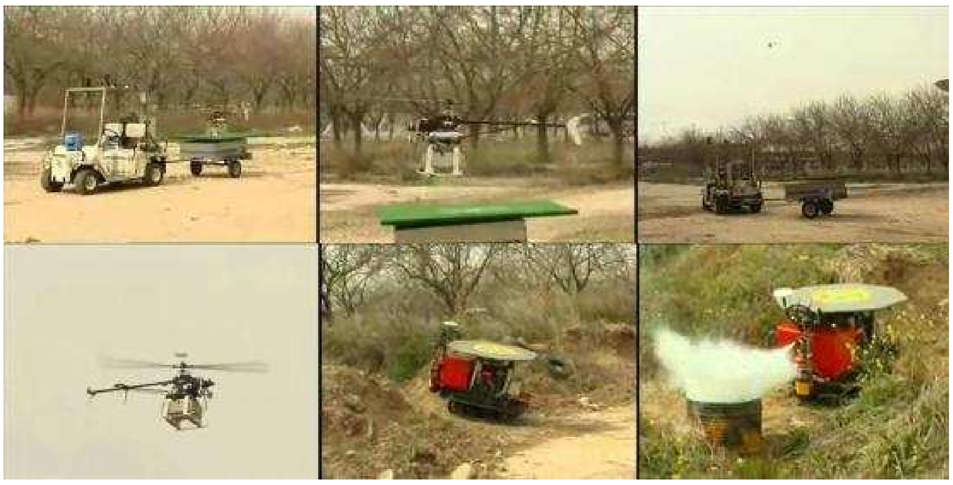


Figure 2.9: Some pictures of the demonstration in the "Alamillo" Park. The goal of the mission is to detect a fire and extinguish it with the collaboration of three robots: ROMEO-4R, HERO2 and AURIGA.

Some photos of the demonstration are shown in Figure 2.9. The mission execution was as follows (see Figure 2.11):

1. At the beginning, the two ground vehicles were in their initial positions (marked as `H` in Figure 2.11) and HERO2 (*blue line*) was on the take-off/landing platform on the ROMEO-4R (*red line*) trailer. In the MPS, the human operator inserted a waypoint `WP1` to be visited as starting exploration point and an area `A` (given by a polygon) to be surveyed.

2. In Figure 2.10, it can be observed the messages exchanged between the robots and the Monitor and Planning Station. After the distributed negotiation process using the S+T algorithm (see Figure 2.12), HERO2 won the `go-to(WP1)` task and the `survey-area(A,fire)` task. Each of these tasks has an associated service won by ROMEO-4R (the `transport(WP1)` service with the `go-to(WP1)` task and the `communication-relay(CR1)` service with the `survey-area(A,fire)` task). Due to the limited flight autonomy of HERO2 (twenty minutes), the transport service was expected to arise. The other service (communication-relay) was created since HERO2 has a limited communication range (virtually

reduced for this demonstration), and it needs to send images back to the MPS during the survey-area task.

3. `Go-to(WP1)` was the first task to be executed since the cost of the HERO2 plan is minimized in this case. Then, ROMEO-4R moved to the `WP1` coordinates (see Figure 2.11) with HERO2 on the take off/landing platform. After reaching `WP1`, the first task and its associated service were completed. HERO2 started the `survey-area(A,fire)` task, which implied to take-off (`TO`), and fly towards the `A` zone, `GT(A)`. At the same time, ROMEO-4R executed the service associated with this task and moved to the `CR1` point (Figure 2.11) to act as a communication relay between HERO2 and the MPS.

4. When ROMEO-4R arrived to CR1, HERO2 started to survey `A` following a list of waypoints generated by its local planner (marked as `A1`, `A2`, `A3`, `A4` and `A5` in Figure 2.11).

5. HERO2 detected the fire (Arrue et al., 2000) and geolocalized it (Merino et al., 2006a), (Merino et al., 2006b) computing the GPS coordinates of the fire. Then, HERO2 generated a new task (`Extinguish(E1)`) and inserted it in the multi-robot negotiation process. This task was allocated dynamically during the mission execution to the AURIGA (*green line*) robot since it was the only robot with the required systems (the other two robots bid an infinite cost for this task). AURIGA started its execution, went close to the fire and activated the extinguisher using a teleoperation interface. Even though the execution of the AURIGA robot was teleoperated, it is important to clarify that the whole task allocation process was done autonomously using the S+T algorithm. After the allocation, the task was transmitted to the teleoperation interface where the human operator executed the task. This fact also demonstrates that this architecture is flexible and enables to combine robots with different degrees of autonomy.

6. At the same time, the fire detection alarm was sent by HERO2 to the Alarm Monitoring Station who automatically created a `Monitor(fire,extinguished)`

task and started an allocation process. This task was clearly allocated to HERO2 since it was the closer robot to the fire. HERO2 was executing this task that finishes when the fire is extinguished (mission completed).

This experiment has demonstrated the coordination among ground and aerial robots using a distributed task allocation system based on a new market approach. Moreover, it has been show that the system can handle tasks created dynamically during the mission execution. The use of tasks and services allows to take advantage of the different characteristics available in an heterogeneous team of robots. The extension of this system to a larger number of robots can be accomplished easily due to the distributed nature of the system as it was demonstrated in the simulations.

2.5 Chapter summary

A distributed task allocation algorithm called S+T and based on a market-based approach has been presented.This protocol allows a team of robots to achieve tasks that could not be executed by a single robot. It does not just coordinate the robots but rather introduce cooperation among them in order to increase the capabilities of the robot team. Two approaches of the algorithm have been developed. In the first one, services are only considered on the allocation process when none of the robots can execute a particular task by itself. The second approach can be adapted to the type of application with a parameter α prioritizing between the execution time and the energy consumption in the mission. On the other hand, the problem of execution loops that appears from the relation between tasks and services has been stated and a distributed algorithm that solves this problem presented.

The S+T protocol has been implemented, simulated and tested in a demonstration with three robots. Simulations have shown that when the number of services increases, the number of messages and the global cost increases, which means that the communication requirements and the energy to execute the mission will be higher. Also, the use of a larger number of services provokes that the maximum cost per robot increases, and therefore, the total time of the mission is higher. Moreover, it has been

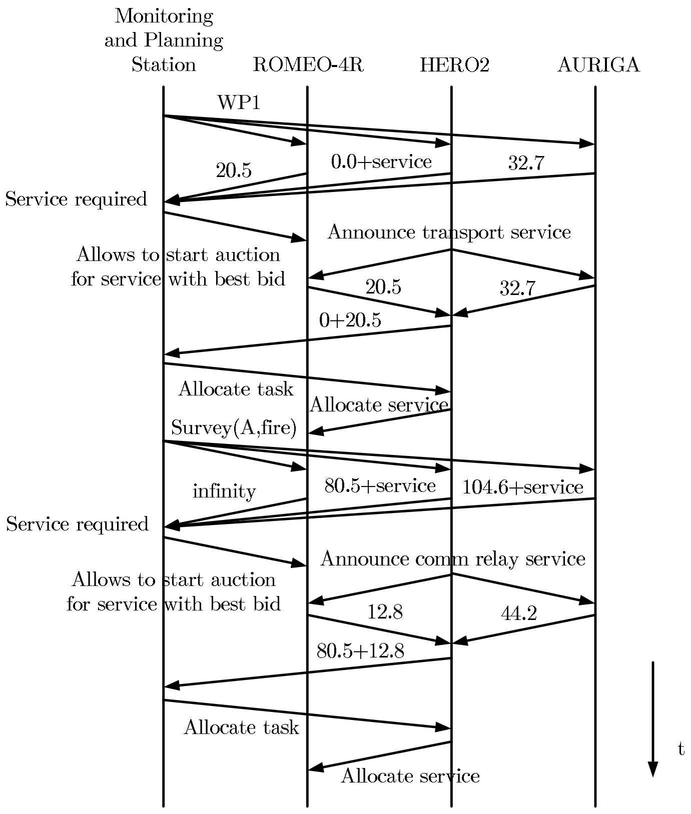


Figure 2.10: Messages interchanged in the initial negotiation process using the S+T algorithm for the demonstration in the “Alamillo” park. HERO2 wins the Go-to(WP1) task since it does not have any cost when a robot executes the transport service. This service is won by ROMEO-4R (it is closer than AURIGA to WP1. HERO2 also wins survey-area(A,fire) task since after executing Go-to(WP1) it needs to travel less distance to survey the area than AURIGA (due to the traversability of the terrain ROMEO-4R cannot execute this task and bids with infinity cost). Finally, ROMEO-4R wins the communication relay service since its cost is again smaller than the cost for AURIGA.

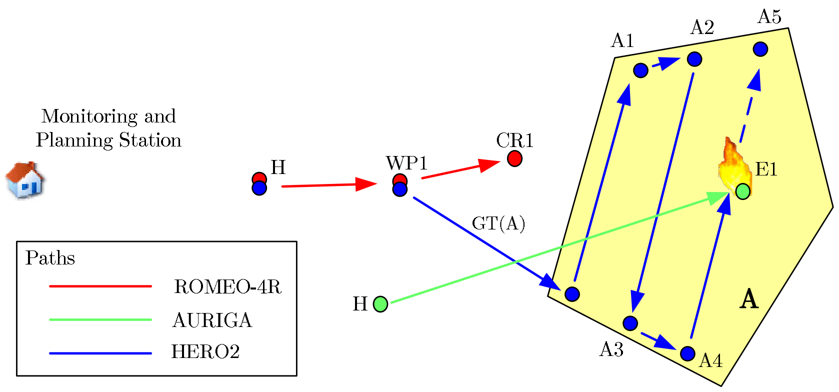


Figure 2.11: Diagram with the different elements involved in the mission execution. It represents the different tasks executed and an illustration of the paths traveled by each robot.

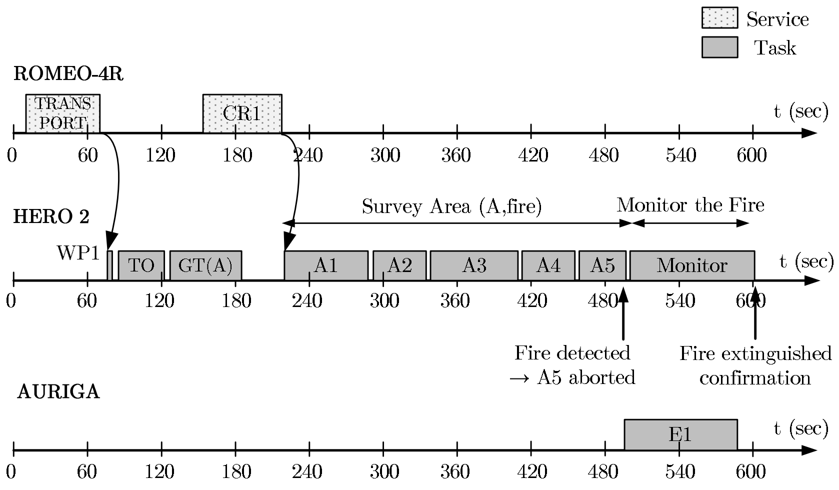


Figure 2.12: Tasks and services executed by each robot during the mission. This diagram shows the relation (preconditions) of the tasks and also a timeline with the duration of each task. All the acronyms are explained in the description of the demonstration, see Section 2.4.2.

shown the effects of some values of the parameter α used to adapt the S+T algorithm to different types of objectives.

Finally, the use of an algorithm such as S+T could increase the probability of completing a mission when tasks need more than one robot to be executed. But, this advantage entails an overhead in the allocation process. It is important to determine whether the application will require a task allocation algorithm that considers services, since not all the missions might need services to be completed successfully.

Chapter 3

Combinatorial market-based algorithm

In this chapter, it is considered scenarios where costs remain constant for a long period of time, and therefore, the use of long-term execution plans makes sense in order to increase the efficiency of task allocation algorithms. These task allocation algorithms are called MRMT (Multiple Robots Multiple Tasks) algorithms.

First, an introduction on MRMT task allocation algorithms is presented. Then, a task allocation algorithm, called SET-MASR, that combines the concepts of reallocation and combinatorial auctions is explained. Tasks are grouped and allocated together instead of using single-item tasks. The efficiency of the allocation is increased since robots calculate their costs with a longer execution horizon and take advantages of the synergies between tasks. However, a high confidence in the task costs is needed. Finally, the synchronization issues for MRMT and MRST (Multiple Robots Single Task) algorithms are also studied.

3.1 Introduction to MRMT algorithms. The SIT-MASR algorithm

Using the concepts from the MRTA formal taxonomy explained in (Gerkey and Matarić, 2004), the MRMT algorithms can be posed as an instance of the ST-SR-TA case:

- Single-task robots (ST): each robot is capable of executing at most one task at a time.
- Single-robot tasks (SR): each task requires exactly one robot.
- Time-extended assignment (TA): tasks are not allocated instantaneously, and for some of the algorithms, more information is available to make the decision.

The work presented in (Dias and Stenz, 2002) was taken as a starting point to study MRMT task allocation algorithms. In the same manner, robots with local plans and multiple tasks allocated to a single robot during the negotiation were considered. Following these ideas, a first algorithm (Viguria et al., 2007) called SIT-MASR (SIngle Tasks negotiation with Multiple Allocations to a Single Robot) was developed as a baseline for further improvements.

The SIT-MASR algorithm uses local plans to increase the local execution horizon, and therefore, improve the efficiency of the allocation. Moreover, one of the key factors of this algorithm is the use of marginal costs (Sandholm, 1993) as bids. During the negotiation process, each robot bids for a task with the cost of inserting this task in its local plan (marginal cost). Taking into account that the objective is the minimization of the sum of the costs, it has been shown in (Tovey et al., 2005) that the appropriate marginal cost for this problem should be defined as the difference between the local plan including the new task (T_i) and the current local plan:

$$MC_{ij} = C(P_j, T_i) - C(P_j), \tag{3.1}$$

where $C(P_j)$ is the total cost of the current local plan for robot j and $C(P_j, T_i)$ is the cost of a new local plan including task T_i.

An obvious question that arises now is where the new allocated task should be inserted within the local execution plan. Ideally, each robot should compute the optimal insertion point of a new task in its current plan. However, this is a complex problem that is equivalent to the TSP (Traveling Salesman Problem) (Lawler, 1985) for exploration scenarios. The TSP problem is a well known $\mathcal{NP}$-hard problem that cannot be solved optimally in polynomial time. However, as is stated in (Lagoudakis et al., 2005), a greedy insertion algorithm also obtains good results and it is much faster than calculating the optimal solution. Therefore, this type of algorithm has been applied to solve the task insertion problem, locating the new task in the position of the local plan which minimizes its insertion cost.

The SIT-MASR algorithm, as well as the rest of market-based algorithms, is based on two different roles played dynamically by the robots: auctioneer and bidders. First, the auction is opened for a period of time, and all the received bids are considered. When the task is allocated, the auctioneer considers to pass its role to another robot. If that happens, the auctioneer changes its role to a bidder, and a new robot becomes the auctioneer starting a task negotiation cycle.

Finally, the SIT-MASR algorithm is dynamic in the sense that new task can be introduced to be allocated at any moment, even during task executions. When the initial negotiation is over, all robots start executing their allocated tasks, but new tasks can be generated at any moment. All the robots take part in the negotiation of those new tasks with the only restriction that the current tasks in execution are not re-negotiated.

3.2 SET-MASR: dynamic task subSETs negotiations with Multiple Allocations to a Single Robot

In this section, a new MRMT algorithm, called SET-MASR, that combines the concepts of reallocation and combinatorial auctions is presented. In this algorithm, tasks are grouped and allocated together instead of using single-item tasks. The efficiency of the allocation is increased since robots calculate their costs with a longer execution horizon and take advantages of the synergies between tasks.

3.2.1 Motivation

Although the SIT-MASR algorithm leads to good solutions in general, simple missions can be found where it does not find the optimal allocation. For example, let consider the mission in Figure 3.1, consisting in visiting waypoints `wp1` and `wp2`. If task `wp2` is announced before task `wp1`, the SIT-MASR algorithm will not find the global optimal solution (represented in Figure 3.1,a):

1. Task `wp2` will be allocated to robot 2 (Figure 3.1,b) which is the nearest one.
2. As the marginal cost of task `wp1` is lower for robot 2 than for robot 1, this task is also allocated to robot 2 (Figure 3.1,c).
3. Robot 2 announces both tasks, but it has the lower marginal costs for them and keep both tasks (Figure 3.1,d).
4. After a given timeout expires, robot 2 starts executing their tasks.

In this particular example, the optimal solution would have been found if robot 2 had announced a subset of tasks composed by `wp1` and `wp2`. This idea has been used to develop the algorithm described in the next subsection.

3.2.2 Algorithm description

In order to find better solutions for the example explained above, the negotiation of subset of tasks was considered in the design of a new algorithm called SET-MASR (dynamic task subSETs negotiation with Multiple Allocations to a Single Robot). The basic idea behind this algorithm is that negotiating subsets of tasks provides more information to the robots than negotiating tasks one by one. It should be also noted that the SET-MASR can be considered as a generalization of the SIT-MASR algorithm, which tries to improve the quality of the solutions.

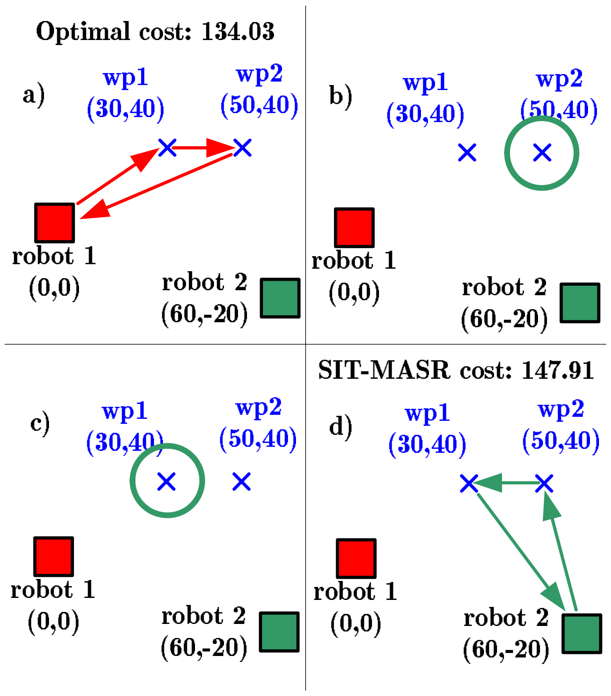


Figure 3.1: A particular mission that shows some limitations of the SIT-MASR algorithm.

Given a subset of tasks $\Gamma_i = \{T_1, T_2, \ldots, T_N\}$ with cardinality $\mid \Gamma_i \mid= N$, the marginal cost associated with this subset for the robot j is given by:

$$MC_{ij} = C(P_j, \Gamma_i) - C(P_j), \tag{3.2}$$

where $C(P_j)$ is the total cost of the current local plan for robot j and $C(P_j, \Gamma_i)$ is the cost of a new local plan including the subset of tasks Γ_i. As was commented before, a greedy approach has been applied to find where to insert the tasks of the subset in the current local plan in order to minimize the insertion cost. In (Lagoudakis et al., 2005), it is stated that the greedy insertion algorithm also obtains good results, and it is much faster than calculating the optimal solution.

On the other hand, a policy for building the subsets of tasks to be auctioned during the negotiation process is required. As it is not feasible to use a brute force algorithm that tries all the possible combinations, in this algorithm, each robot computes the subset of tasks with the highest cost for its local plan. The computational cost to find this subset is not significant for the number of tasks usually managed by a single robot (less than 50). This policy has been selected instead of others, such as the ones explained in (M. et al., 2003) and (Koenig et al., 2007), since it is not assumed that each robot knows all the tasks from the beginning. The goal is to develop a combinatorial market-based algorithm that can deal with scenarios where tasks are created dynamically.

As in a regular market-based algorithm, there are two roles: auctioneer and bidders. The basic steps of the algorithm for the auctioneer is given in Algorithm 4 whereas Algorithm 5 is used by the bidders. At the beginning, the cardinality of the subset of tasks to be announced is one. In this case, the algorithm behaves exactly as the *SIT-MASR* algorithm. Once all the tasks have been allocated and there are no changes in the local plans of the robots during a given period, the subset cardinality is increased by a particular robot. This robot will start the next phase of auctions with subsets of two tasks. Finally, the algorithm will stop when there is no interchange of tasks during a given phase or when the subset cardinality is greater than the number of tasks to be announced in every robot. Once the algorithm has finished, the robots will start executing their local plans.

Algorithm 4 SET-MASR auctioneer algorithm

```
if subset cardinality is $\leq$ dim of announcement list then
  calculate the subset of tasks to be announced
  announce the subset of tasks
  while timer is running do
    receive bids
  end while
  calculate the best bid
  if best bid is smaller than the auctioneer bid then
    send subset of tasks to the winner of the auction
  end if
  delete subset of tasks from announcement list
end if
```

Algorithm 5 SET-MASR bidder algorithm

```
new message received
if new message is task announcement then
  set the cardinality of the subset equal to the number of tasks announced;
  calculate the optimal position of the subset of tasks in the local plan
  calculate bid (marginal cost)
  send bid
else if new message is a task award then
  insert subset of tasks in the local plan in the positions calculated before
  introduce the subset of tasks in the announcement list
end if
```

3.3 Simulation results

Multi-robot missions consisting in visiting waypoints and returning to home positions have been used to test the algorithms in the simulator. Hundreds of simulations with different number of robots and tasks have been performed to compare the SIT-MASR and the SET-MASR algorithms. Moreover, a MRST algorithm, called BS-WR (see Chapter 4), has also been simulated in order to evaluate the relevance of the local plans in the quality of the solutions. Finally, a brute force algorithm has been used to compute the global optimal solutions when the sum of robots and tasks is below a certain value.

In particular, for each given number of robots and waypoints, one hundred missions have been run in a virtual world of 1000x1000 meters using random positions for the robots and the waypoints. Each mission has been simulated with the three algorithms implemented. Table 3.1 shows the different solutions compared with the global optimum. In each cell, the first number is the arithmetic mean of the global cost for 100 random missions, the value between brackets is its standard deviation, and the third number is the difference in percentage with the optimal solution.

From the results, it should be noted that using a local plan during the auction process improves the solutions significantly. On the other hand, the algorithms presented in this paper achieve very good results, with a maximum difference of 4.7% with respect to the optimal solution. Moreover, the SET-MASR algorithm computes better solutions in mean than the SIT-MASR. Also, it is important to point out that the standard deviation values are high because missions are calculated at random, i.e., the global cost of the different random missions can differ very much among them.

Tables 3.2, 3.3 and 3.4 show the results from simulating the algorithms with a high number of tasks for three, five and seven robots. In those cases, it was not possible to compute the optimal solution with the brute force algorithm due to the $\mathcal{NP}$-hard nature of the problem. Therefore, in these tables the percentage value corresponds to the difference with the solutions found with the SET-MASR algorithm. The rest of values have the same meaning and units explained for Table 3.1.

The results obtained with three and five robots are quite similar; a significant difference between the BS-WR algorithm, a MRST algorithm, and the MRMT algorithms (at least $118,9\%$), and slightly better results with the SET-MASR algorithm than with the SIT-MASR algorithm. But with seven robots the solutions of the BS-WR algorithm are better as expected; with a few robots, a single robot has a higher probability to execute a task with a high cost, if the other robots are not idle.

Figure 3.2 compares the mean of messages transmitted by each robot using the different algorithms in one hundred missions with five robots. As expected, the number

Robots	***Tasks***	***BS-WR***	***SIT-MASR***	***SET-MASR***	***Optimum***
3	3	$2371,22$	$1453,44$	$1444,9$	$1435,4$
		$(742,4)$	$(369,63)$	$(364,23)$	$(362,36)$
		$65,18\%$	$1,25\%$	$0,66\%$	
3	5	$4144,7$	$2097,83$	$2088,1$	$2061,8$
		$(923,42)$	$(414,52)$	$(405,38)$	$(396,3)$
		$101,23\%$	$1,74\%$	$1,27\%$	
3	7	$5073,2$	$2473,65$	$2457,5$	$2362,6$
		$(788,75)$	$(385,46)$	$(381,27)$	$(335,81)$
		$114,73\%$	$4,7\%$	$4,01\%$	
3	9	$6070,8$	$2816,59$	$2773,3$	$2649,5$
		$(850,46)$	$(398,16)$	$(392,7)$	$(332,58)$
		$129,13\%$	$6,3\%$	$4,6\%$	
5	3	$1764,36$	$1274,88$	$1268,21$	$1264,78$
		$(627,87)$	$(365,74)$	$(359,53)$	$(356,04)$
		$39,49\%$	$0,79\%$	$0,27\%$	
5	5	$3808,55$	$1842,96$	$1816,77$	$1793,35$
		$(921,45)$	$(363,62)$	$(348,94)$	$(337,56)$
		$112,37\%$	$2,76\%$	$1,3\%$	
5	7	$5407,59$	$2225,67$	$2209,71$	$2161,68$
		$(1238,38)$	$(384,74)$	$(381,82)$	$(365,82)$
		$150,15\%$	$2,96\%$	$2,22\%$	

Table 3.1: Solutions computed with three different distributed task allocation algorithms and the optimal result. The first number is the arithmetic mean of the global cost, the value between brackets is its standard deviation (both values in meters), and the third number is the difference in % with the optimal global cost.

Tasks	***BS-WR***	***SIT-MASR***	***SET-MASR***
9	$6070, 8$ $(850, 46)$ 118.9%	$2816, 59$ $(398, 16)$ 1.5%	$2773, 3$ $(392, 7)$
15	$9146, 9$ $(1104, 5)$ 152, 90%	$3655, 61$ $(401, 85)$ 1, 07%	$3616, 78$ $(372, 64)$
20	$12324, 99$ $(1085, 64)$ 198, 9%	$4157, 12$ $(379, 07)$ 0, 83%	$4122, 79$ $(364, 75)$
30	$16780, 74$ $(1490, 19)$ 236, 97%	$5035, 43$ $(414, 09)$ 1, 11%	$4979, 8$ $(476, 56)$
40	$22045, 37$ $(1909, 91)$ 294, 9%	$5634, 98$ $(332, 9)$ 0, 94%	$5582, 42$ $(353, 1)$

Table 3.2: Solutions computed for missions with three robots and different number of waypoints. The first number is the arithmetic mean of the global cost, the value between brackets is its standard deviation (both values in meters), and the third number is the difference in % with the SET-MASR global cost.

Tasks	***BS-WR***	***SIT-MASR***	***SET-MASR***
9	$6558,51$	$2744,81$	$2706,59$
	$(1282,5)$	$(382,15)$	$(378,55)$
	$142,31\%$	$1,41\%$	
15	$9779,13$	$3488,06$	$3459,86$
	$(11434,31)$	$(381,93)$	$(383,12)$
	$182,64\%$	$0,81\%$	
20	$12277,43$	$4058,52$	$4016,01$
	$(1389,83)$	$(398,18)$	$(389,47)$
	$205,70\%$	$1,05\%$	
30	$17312,18$	$4969,31$	$4894,02$
	$(1561,04)$	$(361,43)$	$(391,54)$
	$253,74\%$	$1,53\%$	
40	$22353,78$	$5727,55$	$5559,75$
	$(1721,68)$	$(342,5)$	$(756,3)$
	$306,06\%$	$3,01\%$	

Table 3.3: Solutions computed for missions with five robots and different number of waypoints. The first number is the arithmetic mean of the global cost, the value between brackets is its standard deviation (both values in meters), and the third number is the difference in % with the SET-MASR global cost.

Tasks	***BS-WR***	***SIT-MASR***	***SET-MASR***
9	$3214,16$	$2472,31$	$2465,29$
	$(653,28)$	$(424,77)$	$(422,53)$
	$30,37\%$	$0,28\%$	
15	$4788,04$	$3360,78$	$3337,36$
	$(716,12)$	$(414,97)$	$(400,78)$
	$43,46\%$	$0,7\%$	
20	$6093,22$	$3975,39$	$3954,57$
	$(806,31)$	$(381,4)$	$(364,07)$
	$54,08\%$	$0,52\%$	
30	$8592,23$	$4818,46$	$4794,23$
	$(959,65)$	$(508,73)$	$(430,94)$
	$79,1\%$	$0,50\%$	
40	$10932,07$	$5619,95$	$5598,34$
	$(1152,79)$	$(521,65)$	$(470,45)$
	$95,37\%$	$0,44\%$	

Table 3.4: Solutions computed for missions with seven robots and different number of waypoints. The first number is the arithmetic mean of the global cost, the value between brackets is its standard deviation (both values in meters), and the third number is the difference in % with the SET-MASR global cost.

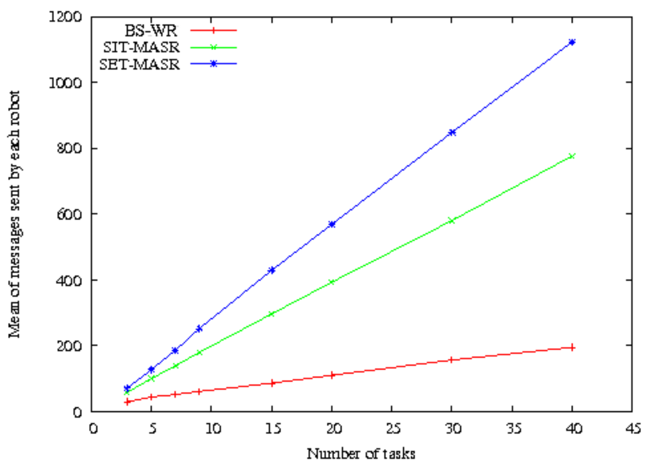


Figure 3.2: Mean of the messages sent per robot in one hundred missions with five robots and different number of waypoints.

of messages increases with the number of tasks. The SET-MASR algorithm needs more messages than others due to its more complex negotiation protocol, but the number of messages also scales linearly with the number of tasks.

Finally, the best ratio between the improvement of the solutions and the number of messages required is achieved with the SIT-MASR algorithm. However, the SET-MASR algorithm should be used if the minimization of the traveled distance (energy constraints) is a major issue. Also, the BS-WR algorithm can be used if the communication among robots should be minimized, or task costs are not expected to remain constant for a long period of time.

3.4 Synchronization aspects in market-based task allocation algorithms

The need of a synchronization mechanism in task allocation algorithms is a major issue. Task allocation algorithms that use a synchronization mechanism are more complex and less parallel, and therefore, they take more time to complete the task allocation scheme and communicate a higher number of messages.

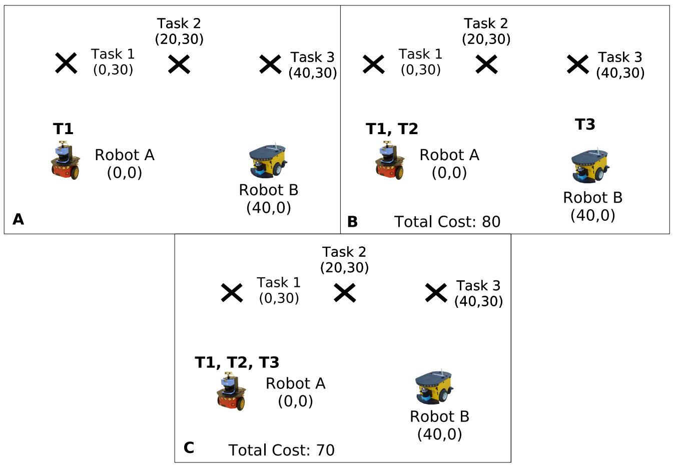


Figure 3.3: Figure A shows the initial situation where robot A has already won task 1. If tasks 2 and 3 are published at the same time and the task negotiation cycles run concurrently, the marginal cost for task 2 will be 20 for robot A and 36.05 for robot B. While the marginal costs for task 3 will be 40 for robot A, and 30 for robot B. Therefore, in Figure B it is shown the final allocation where tasks 1 and 2 are allocated to robot A and task 3 to robot B. In Figure C it is shown the final allocation when the task negotiation cycles are run sequentially, obtaining a lower global cost than previously.

As was explained before, MRMT algorithms can allocate more than one task per robot and make use of local plans (Dias and Stenz, 2002; Viguria et al., 2007). Every robot has a plan that describes the order of execution of the different tasks allocated to the robot. In these cases, better results are obtained if the task negotiation cycles are not run concurrently (Dias et al., 2004), since it is crucial to know the current state of the local plan in order to calculate an updated marginal cost. For example, in Figure 3.3, a situation is shown where robots are participating in two task negotiation cycles at the same time. Due to the fact that both task negotiation cycles are run at the same time, the marginal costs are not calculated correctly. Robots do not know yet if they will win any of these tasks, and therefore, these tasks are not considered in the calculation of the marginal cost. However, when the task negotiation cycles are run sequentially, the marginal costs are always calculated correctly and better results are obtained, as can be seen in Figure 3.3. Therefore, the synchronization during the negotiation process has a relevant impact on the efficiency of the solutions.

Tasks & Robots	***BS***	***BS without synchronization***
3	909.35	909.35
4	1473.52	1473.29
6	2020.13	2055.72
8	2443.57	2450.10
10	2865.81	2901.01
12	3233.25	3245.45

Table 3.5: Global cost mean obtained with the BS algorithm with and without synchronization over 100 simulations per each case.

In order to force the task negotiation cycles to be run sequentially, a synchronization mechanism is needed. Usually a token-based algorithm (Lynch, 1996) is used where only the robot with the token can start a task negotiation cycle. However, the need for serialized task negotiation cycles slows down the allocation algorithm and also increments the number of messages needed.

On the other hand, when the task allocation algorithm does not make use of local execution plans, as happens in MRST algorithms, the marginal costs are not used. These characteristics allow the same quality of results to be obtained with and without synchronization. In order to illustrate this result, the BS algorithm (see Chapter 4) has been implemented with and without synchronization. The synchronization has been implemented by means of a token protocol that guarantees only one task negotiation cycle running at a given time. As it can be seen in Table 3.5, the results obtained for the BS algorithm are almost the same with and without the token protocol. The maximum error for all the cases and both algorithms is 1.9%, which is negligible.

The possibility of using task allocation algorithms without any kind of synchronization mechanism makes these MRST algorithms run much faster than MRMT algorithms, since task negotiation cycles can be run in parallel. Also, this fact reduces the number of messages used in the task allocation algorithm, since all the messages used in the token algorithm are not needed any more. In Figure 3.4, the difference in execution time is shown between the BS experiments with and without the token protocol. It can be observed how the difference increases with the number of robots and tasks. When the number of robots is 12, the BS algorithm without a synchronization mechanism is 32.34% faster than the algorithm with the synchronization mechanism (token). The two main sources of delays in a market-based algorithm with a token mechanism are:

- **Wait time for bids** (T_b): due to the fact that the number of robots is not known a priori, the auctioneer should wait a certain amount for the bids to be considered in the task negotiation cycle. When a synchronization method is used, the task negotiation cycles are serialized, so the time to finish N auctions is at least $T_b \cdot N$. While in algorithms without synchronization mechanisms, the task negotiation cycles are run in parallel. Due to this fact the difference between an algorithm with and without a synchronization mechanism increases with the number of robots. With more robots, the number of task negotiation cycles increases, and also, more number of auctions can be run in parallel.
- **Time to request the token again** (T_k): in this implementation, robots that need to start a task negotiation cycle should ask for the token first. It could happen that the token cannot be passed because another robot is in the middle of a task negotiation cycle. Therefore, the robot waits for a certain amount of time, T_k, before it asks for the token again. It is important to point out that the main purpose of this timer is the reduction of the communication traffic. However, if the task negotiation cycle is finished just after someone asked for the token, the next task negotiation cycle will not start for at least T_k seconds later. This delay can be reduced if the robot with the token remembers at least one of the robots that has asked for the token during the task negotiation cycle. When the negotiation is finished, it can send the token immediately to that robot.

Finally, it is important to point out that algorithms without any synchronization method are more robust and fault tolerant since they are simpler. For example, if a token-based synchronization method is used, there should be mechanisms to recover the token when it is lost.

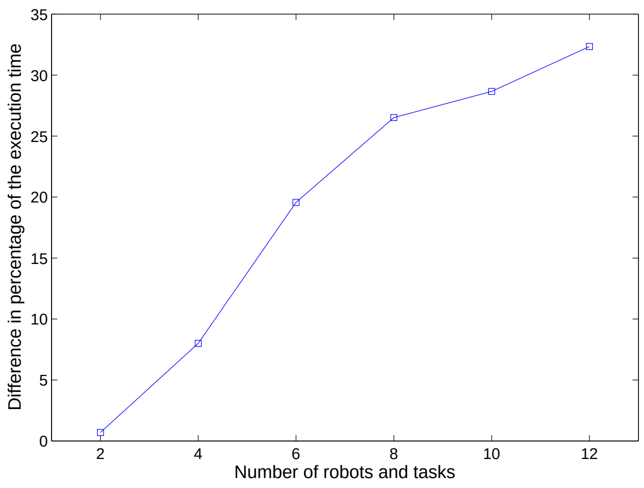


Figure 3.4: Difference in percentage of the execution time between algorithms with and without the synchronization mechanism ($T_b = 500ms$ and $T_k = 300ms$). The results are shown for different number of robots and tasks over 100 simulations per case.

3.5 Chapter summary

In this chapter a distributed market-based algorithm (SET-MASR), that solves the MRTA problem, has been presented. This algorithm is a MRMT task allocation algorithm since robots have a local plan and multiple tasks can be allocated to a single robot during the negotiation. Moreover, the SET-MASR algorithm is based on the negotiation of subsets of tasks, and it can be considered as a generalization of the SIT-MASR algorithm (which only negotiates single tasks) designed to improve its solutions.

From the simulation results, it is derived that using a local plan during the auction process improves the solutions significantly. Furthermore, the algorithms presented in this section lead to good results when comparing with the global optimal solutions in simulations consisting in visiting waypoints. Moreover, the SET-MASR algorithm computes better solutions in mean than the SIT-MASR. On the other hand, both algorithms performance scales well when the number of robots and tasks increases.

Finally, the need of synchronization mechanisms, in task allocation algorithms, has been studied. It has been explained that the use of local plans forces the need of a synchronization mechanism in order to maintain the efficiency of the solution. Also, it has been shown that MRST task allocation algorithms obtain very similar results with and without the use of a synchronization mechanism.

Chapter 4

MRST market-based algorithms

In this chapter, different MRST (Multiple Robots Single Task) market-based algorithms that solve the multi-robot task allocation (MRTA) problem are explained. These algorithms are used when task costs may change dynamically, and therefore, it is not worth to create long-term execution plans. In these situations, after the execution of the first task, costs may change enough that the initial allocation is less efficient than a hypothetical allocation obtained with the updated costs.

First, two basic algorithms are explained to introduce the concepts related to MRST algorithms and show the importance of reallocations in the quality of the solution. Next, different modifications of the basic market-based algorithm that improve its results are addressed. All the algorithms are tested in simulation and their results are commented. Finally, two of the most representative algorithms are implemented in real robots and the conclusions from a significant number of experiments are presented.

4.1 Introduction

Using the concepts from the MRTA formal taxonomy explained in (Gerkey and Matarić, 2004), the MRST algorithms have the following characteristics:

- Single-task robots (ST): each robot is capable of executing at most one task at a time.

- Single-robot tasks (SR): each task requires exactly one robot.
- Time-extended assignment (TA): tasks are not allocated instantaneously, and for some of the algorithms, more information is available to make the decision. The only exception is the BS-WR algorithm that belongs to the instantaneous assignment category. This algorithm is explained in detail in Section 4.2.1.

Thus, these algorithms can be designated as ST-SR-TA, except the BS-WR algorithm that can be designated as ST-SR-IA.

This work has taken the MURDOCH (Gerkey and Matarić, 2000) task allocation algorithm as baseline. MURDOCH is a MRST task allocation algorithm, where robots do not take part in auctions while they are executing a task. If a new task is announced dynamically, it will be allocated to idle robots. If all robots are executing a task, then the task is discarded or has to be reannounced after a period of time. The same behavior is applied to the algorithms that will be presented next. However, this behavior makes difficult the comparison between algorithms for the MRTA problem, since the subsequent allocations depend on the execution time of the first allocated tasks, which at the same time depends on the initial allocation obtained with each of the different algorithms. Since one of the thesis objectives is the study of the efficiency of task allocation algorithms, and in order to obtain a fair comparison between them, the MRST algorithms will be validated using the Initial Formation (IF) problem instead of the MRTA problem.

The IF problem (Howard and Viguria, 2007), is a rendition of the MRTA problem, in which each robot can only be allocated to one task. In order to illustrate the differences between both problems, it can be thought about the MRTA problem as a Multiple Traveling Salesman Problem (Lawler, 1985; Laporte and Nobert, 1980) and the IF problem can be viewed as a classical job assignment problem (Kuhn, 1955) where robots are the workers and tasks are the jobs to be executed by those workers. Since in the IF problem only one task is allocated per robot, it can be considered as a fair comparison framework to be used with MRST algorithms.

The IF Problem has received less attention in the task allocation domain than the MRTA problem. However, this type of problem becomes important within the field of formation control ((Hu and Egerstedt, 2001; Lawton et al., 2003)) where using local information and control laws, the distributed algorithm is able to drive a given formation error to zero. As it is stated in (Ji and Egerstedt, 2006), these algorithms require a first step that assigns the robots to the formation positions while taking into account their initial locations, i.e., answer the question, "who goes where?"

An advantage of using the IF problem for comparison purposes is that this problem can be solved optimally in a small period of time using centralized algorithms, such as the Hungarian method (Kuhn, 1955). Alike the MRTA problem, the computational complexity of the optimal algorithm is not an issue, since the Hungarian method has complexity $\mathcal{O}(n^3)$ (where n is the number of robots or tasks), and there are algorithms with complexity $\mathcal{O}(n^2)$, such as (Toroslu and Üçoluk, 2007). This characteristic allows the comparison between market-based and optimal allocations for any number of robots and tasks.

In summary, novel MRST and distributed market-based algorithms are explained in this chapter. These algorithms are compared with the basic market-based algorithms for the IF problem. However, it is important to point out that although these algorithms are validated using the IF problem, there is no restriction to solve the MRTA problem with MRST algorithms, as was done with MURDOCH (Gerkey and Matarić, 2000).

4.2 Basic market-based algorithms

Two basic algorithms are presented next. The main difference between both of them is the use of reallocation mechanisms in the BS algorithm. The rest of characteristics are very similar: the cost function is equal to the task costs (euclidean distance), the lowest cost is used as the task awarding mechanism and the global objective function is the minimization of the sum of the costs or the so-called global cost.

4.2.1 BS-WR: BaSic market-based algorithm Without Reallocations

In this algorithm, bidders broadcast their bids only if they do not already have a task assignment, i.e., when a task is allocated to a robot, it no longer bids on other tasks in the following auctions. Therefore, robots execute their first allocated task. In the IF problem, all the initial tasks are usually announced in a small period of time. After all the robots have completed their tasks, new tasks may be announced. For this reason, once a robot has finished executing its task, it resumes to accept task announcements and broadcast the corresponding bids again. If a new task is announced dynamically, it will be allocated to idle robots. This algorithm is based on MURDOCH (Gerkey and Matarić, 2000) which was applied to the MRTA problem.

On the other hand, this algorithm is easy to implement and uses a small number of messages. However, the solution depends on the order that tasks are announced and, as such, may not result in an efficient solution. The bidder and auctioneer algorithms are explained in Algorithms 6 and 7.

Algorithm 6 Bidder algorithm

```
a new message is received
if new message is a task announcement then
   if won-task list is empty then
      calculate bid (distance to the task)
      send bid to the auctioneer
   end if
else if new message is a task award then
   introduce awarded task in the won-task list
end if
```

4.2.2 BS: BaSic market-based algorithm

This algorithm uses reallocation of the tasks to increase the efficiency of the solution, but the number of messages also increases. This algorithm is independent of the order that tasks are announced, i.e., it gets the same solution regardless of the order in which tasks are introduced in the system. The basic idea is that each robot should

Algorithm 7 Auctioneer algorithm

```
if announcement task is not empty then
  announce task
  while timer is running do
    receive bids
  end while
  calculate best bid
  send task to best bidder
  delete task from announcement list
end if
```

have only one awarded task, so it will keep the task with the lowest cost. If it wins a new task that has a lower cost than the one already won, it will reallocate the old task to the robot with the best bid worse than its own bid. The best bid worse than the robot's bid is selected in order to avoid infinite loops in the negotiation. This scenario could happen when two robots have the best bids for at least three tasks as shown in Figure 4.1.

This reallocation process lasts until no further tasks are announced or reallocated. This means that all the tasks have been assigned and all the robots have at most one task allocated. Therefore, every robot waits a period of time without receiving any task announcement before executing its task. The auctioneer and bidder algorithms are explained in Algorithm 8 and 9 respectively.

Algorithm 8 Auctioneer algorithm

```
if announcement-task list is not empty then
  announce task
  while timer is running do
    receive bids
  end while
  calculate best bid worse than the robot's bid
  send task to best bidder
  delete task from announcement-task list
end if
```

There are situations when this algorithm does not obtain good results which usually happens when a robot has to execute a task that is the worst one for its own

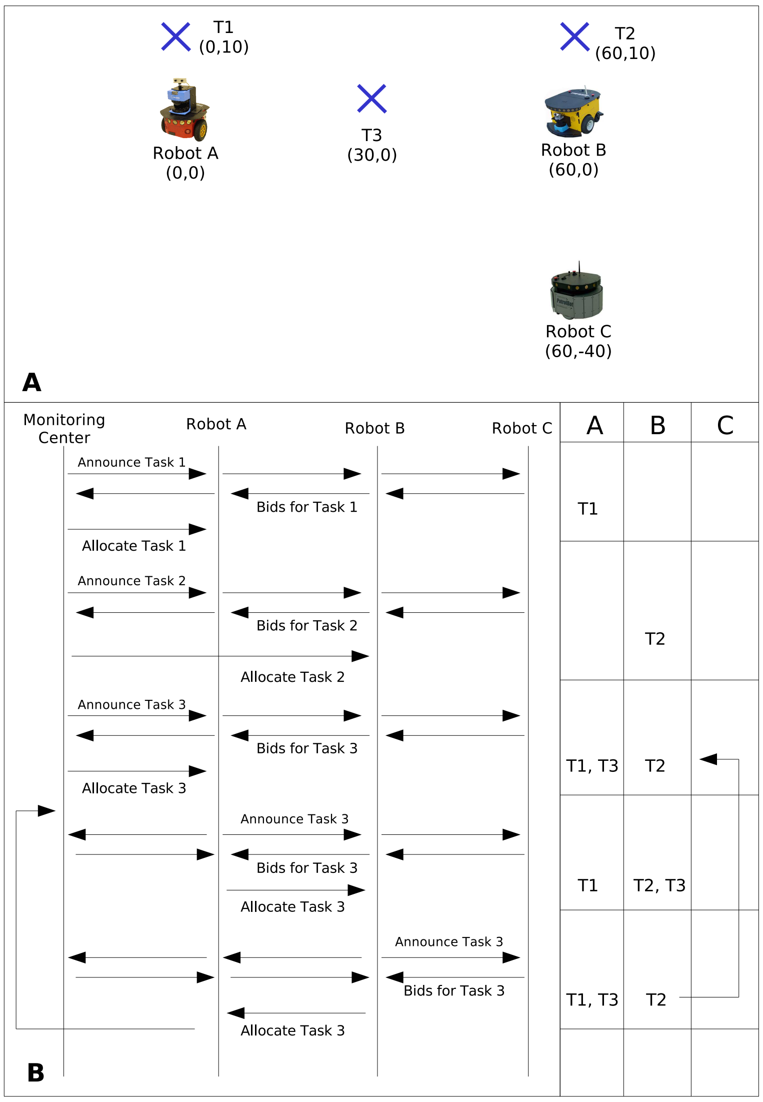


Figure 4.1: Figure A presents the initial position of the robots and the tasks. Figure B presents the messages exchanged among the different agents and shows how an infinite loop appears in the negotiation protocol.

Algorithm 9 Bidder algorithm

```
a new message is received
if new message is a task announcement then
  calculate bid (distance to the task)
  send bid to the auctioneer
else if new message is a task award then
  if the robot has already won a task then
    if cost of the new task < cost of the won one then
      introduce old task in announcement-task list and delete it from won-tasks
      list
      introduce awarded task in the won-tasks list
    else
      introduce awarded task in the announcement-task list
    end if
  else
    introduce awarded task in the won-tasks list
  end if
end if
```

interest, as can be seen in Figure 4.2. In this example, the global cost obtained with the BS algorithm is 66.67% greater than the optimal allocation.

4.3 Improved market-based algorithms

Two different aspects of market-based algorithms are considered in order to improve the BS algorithm: modification of the cost function and the task selection mechanism. Moreover, the improved algorithm should keep the advantages of the market-based approach: fault tolerance, independence from the number of robots and high adaptation to changes in the environment using reallocations.

4.3.1 RMA: Robot Mean Allocation algorithm

The first improved algorithm (RMA) is focused on trying to choose in a more clever way the task that should be kept when a bidder wins more than one task. This is accomplished using additional knowledge available to the system. Instead of keeping

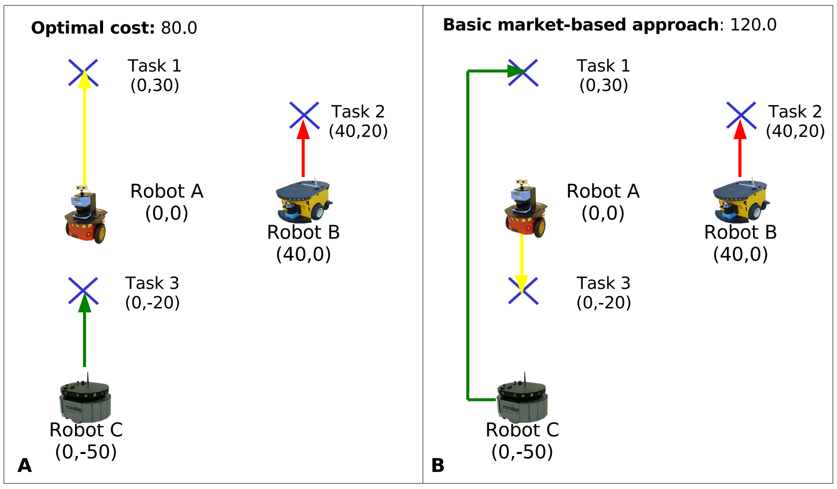


Figure 4.2: Difference in cost between the optimal allocation and the one obtained with the basic market-based algorithm.

the task with the smallest distance to the robot, the task with the highest difference between the distance to the robot and the mean of its distance to all the robots will be selected. In other words, suppose that there are a finite number of robots N_R and robot R_k has won tasks T_i and T_j. In this case, robot R_k keeps task T_i if and only if

$$\sum_{r=1}^{N_R} \frac{D(R_r, T_i)}{N_R} - D(R_k, T_i) > \sum_{r=1}^{N_R} \frac{D(R_r, T_j)}{N_R} - D(R_k, T_j)$$

where $D(R_k, T_i)$ is the distance between robot R_k and task T_i. The reason behind this idea is to make the robot willing to choose the task that is best for the team, not just for itself. So, robots will more likely win tasks that have a high cost for the rest of the robots. For example, let us suppose that a robot has won tasks 1 and 2 which original costs are 5 and 25 respectively. Also, the mean of the costs for task 1 is 15 and for task 2 is 50. Therefore, the RMA robot cost used to decide which task to keep will be 10 and 25 for task 1 and 2 respectively. Since the algorithm chooses the task with the highest RMA robot cost, task 2 will be the task kept for the robot. Although it is the task with the highest cost for itself. However, it is the best option taken into consideration the performance of the team of robots.

The question that arises now is how to calculate the mean of the distances for a certain task. During the normal operation of the algorithm, the auctioneer receives bids from all functional robots in order to allocate the task to the best robot. At this moment, the auctioneer knows all the distances between every robot and the current task. Thus, the mean is calculated by the auctioneer and transmitted to the robot within the message that informs the robot that has won the task. The major difference with the BS algorithm is that the robot should remember the mean associated with the won task. Furthermore, the robot is able to compare their means to different tasks because it remembers the mean of the task already won and the mean of the new allocated task is sent by the auctioneer, as was explained previously.

4.3.2 TMA: Task Mean Allocation algorithm

In the TMA algorithm, instead of changing the way that task selection mechanism, the cost function will be changed. In the BS algorithm the cost function used to calculate the bid for a certain task is the distance between the robot and the task. However, in this improved algorithm the cost function will be the difference between the distance of the robot and the task, minus the mean of the distances between that robot and all the tasks, i.e.,

$$C(R_i, T_j) = D(R_i, T_j) - \sum_{t=1}^{N_T} \frac{D(R_i, T_t)}{N_T} \tag{4.1}$$

where $C(R_i, T_j)$ is the cost function for robot R_i and task T_j and the total number of tasks is N_T. The idea is to decrease substantially the cost of a task when is close to a robot and the rest of the tasks are far away from the same robot. But if all the tasks have similar costs, the cost reduction will be smaller. Therefore when bids are received by the auctioneer, the tasks from robots that are in the first situation will be favor with respect to the tasks from robots that are in the second situation. Also, if a task is far away from a robot and the rest are close to the robot, the cost of the task will be kept almost the same. For example, let us suppose that an auctioneer has started an auction for task 1 with robots A and B. The original costs for task 1 are 5 units for both robots. However, the mean of the task costs is 20 for robot A and 7 for robot B. Therefore, the TMA task costs for task 1 are -15 and -2 for robots A and B respectively. Finally, robot A will win the auction since it has the lowest TMA task cost.

The rest of the algorithm works the same as the BS algorithm but using the new cost function instead of the distance. At the bidder side, when one robot wins two tasks, it will compare the costs using the new cost function, and it will select the task with the lowest TMA cost for itself. Thus, a robot R_k that have won two tasks (T_i and T_j) keeps T_i if and only if

$$C(R_k, T_i) < C(R_k, T_j),$$

or

$$D(R_k, T_i) - \sum_{t=1}^{N_T} \frac{D(R_k, T_t)}{N_T} < D(R_k, T_j) - \sum_{t=1}^{N_T} \frac{D(R_k, T_t)}{N_T}. \qquad (4.2)$$

As it can be seen in (4.2), the sum factor is equal in both parts of the inequality. So, the tasks selected by the bidder are the same by using either the distance or the new cost function.

The differences between the BS and TMA algorithm appears also in the auctioneer. When the auctioneer receives the costs from all the bidders, they no longer contain just the distances between robots and tasks. However, the task awarding mechanism remains the same but using the new cost function, i.e., the task is allocated to the robot that has sent the lowest bid. In summary, task T_i is allocated to robot R_j if an only if

$$D(R_j, T_i) - \sum_{t=1}^{N_T} \frac{D(R_j, T_t)}{N_T} < D(R_k, T_i) - \sum_{t=1}^{N_T} \frac{D(R_k, T_t)}{N_T}, \forall k \neq j.$$

The only drawback to this improvement is that robots should know the different tasks at the beginning in order to calculate the mean of the distances. However, the resources needed for the algorithm are almost the same as that for the BS algorithm since robots only have to memorize the mean calculated at the beginning and implement one basic cost function operation.

4.3.3 RTMA: Robot and Task Mean Allocation algorithm

The last algorithm is a combination between the RMA and the TMA algorithms. Therefore, the cost function is the one used in the TMA algorithm, while the task selection mechanism is the one used in the RMA algorithm. In the RMA algorithm, this selection mechanism uses the euclidean distance as the cost function. This could lead to an implementation problem since the task selection mechanism uses a different cost function than the one used to calculate bids (TMA cost function). However, it is shown next that the same results are obtained whether the euclidean distance or the TMA cost function is used in the task selection mechanism.

First, if the distances are used to calculate costs, task T_i will be the one selected if and only if

$$\sum_{r=1}^{N_R} \frac{D(R_r, T_i)}{N_R} - D(R_k, T_i) > \sum_{r=1}^{N_R} \frac{D(R_r, T_j)}{N_R} - D(R_k, T_j). \tag{4.3}$$

On the other hand, if the TMA cost function is used, task T_i will be the one selected if and only if

$$\sum_{r=1}^{N_R} \frac{C(R_r, T_i)}{N_R} - C(R_k, T_i) > \sum_{r=1}^{N_R} \frac{C(R_r, T_j)}{N_R} - C(R_k, T_j)$$

where $C(R_r, T_i) = D(R_r, T_i) - \sum_{t=1}^{N_T} \dfrac{D(R_r, T_t)}{N_T}$. Thus,

$$\begin{aligned}\sum_{r=1}^{N_R} \frac{D(R_r, T_i)}{N_R} - \sum_{r=1}^{N_R}\sum_{t=1}^{N_T} \frac{D(R_r, T_t)}{N_T \cdot N_R} - D(R_k, T_i) + \sum_{t=1}^{N_T} \frac{D(R_k, T_t)}{N_T} > \\ \sum_{r=1}^{N_R} \frac{D(R_r, T_j)}{N_R} - \sum_{r=1}^{N_T}\sum_{t=1}^{N_T} \frac{D(R_r, T_t)}{N_T \cdot N_R} - D(R_k, T_j) + \sum_{t=1}^{N_T} \frac{D(R_k, T_t)}{N_T}.\end{aligned}$$

Simplifying the previous equation,

$$\sum_{r=1}^{N_R} \frac{D(R_r, T_i)}{N_R} - D(R_k, T_i) > \sum_{r=1}^{N_R} \frac{D(R_r, T_j)}{N_R} - D(R_k, T_j). \tag{4.4}$$

As can be seen, (4.3) and (4.4) are exactly the same. However, due to practical implementation issues, it is easier to compare tasks using the TMA cost function since bids are calculated with it.

The auctioneer algorithm is practically the same as shown in Algorithm 8. The task awarding mechanism is still the one with the lowest cost, but the bid is calculated with the TMA cost function. On the other hand, the new bidder algorithm is explained in Algorithm 10.

Algorithm 10 Bidder algorithm

```
a new message is received
if new message is a task notification then
    mean = mean + $D(R_k, T_i)$/N
else if new message is a task announcement then
    calculate bid (distance to the task minus mean)
    send bid to the auctioneer
else if new message is a task award then
    if the robot has already won a task then
        if cost of the new task - mean of the costs > cost of the won one - mean of
        the costs then
            introduce old task in announcement-task list and delete it from won-tasks
            list
            introduce awarded task in the won-tasks list
        else
            introduce awarded task in the announcement-task list
        end if
    else
        introduce awarded task in the won-tasks list
    end if
end if
```

4.4 Simulation results

A multi-robot simulator has been used to test distributed task allocation algorithms. The same simulator is used for all the algorithms presented in this thesis. This simulator is based on an architecture (see Appendix A) designed for heterogeneous robots (Maza et al., 2006; Viguria et al., 2010) and divided into three layers. The highest layer is independent from the type of robot and is the one aware of the existence of other robots. Thus, the task allocation algorithm is implemented in this layer. On the other hand, the last two layers are used to execute the different tasks allocated to the robot, and make the creation of new algorithms easier by using a modular and component-based architecture.

The communication among robots has been implemented by using the BBCS (BlackBoard Communication System) developed by the Technical University of Berlin (Remußet al., 2004), and created upon the UDP protocol. The BBCS is a robust communication system implemented via a distributed shared memory, the blackboard (BB), in which each network node has a local copy of the BB portion it is accessing. This communication system allows to run a multi-robot simulation in a single or multiple machines, and it can be also used as the communication system for the real robots.

4.4.1 Ideal simulations

The different algorithms have been tested using initial positions of the robots and tasks calculated at random in a virtual world of 1000x1000 meters without obstacles where costs are calculated as euclidean distances between robots and tasks. Although robot positions and task locations are calculated at random, they are calculated once for each number of robots and tasks, and afterwards used for all the different algorithms. Also, it is important to point out that in all the results shown in this section, in order to obtain a fair comparison between algorithms, the same number of robot positions and task locations is used for the different algorithms.

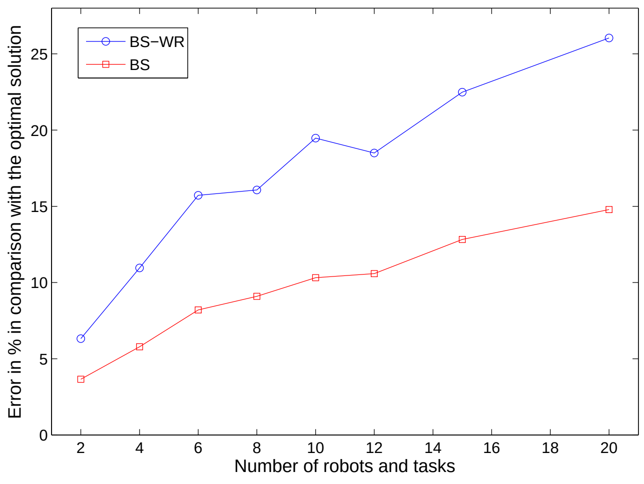


Figure 4.3: Mean error in percentage in comparison with the optimal solution for the BS-WR and BS algorithms where the initial positions of the robots and locations of the tasks are calculated at random over 100 simulations.

First, the BS-WR algorithm is compared against the BS algorithm to state the advantage of the use of reallocations since it increases the performance of task allocation algorithms. As can be seen in Figure 4.3, the BS algorithm obtains better results than the BS-WR algorithm for all the cases. Also, when reallocations are used, the final allocation does not depend on the order of the task announcements.

Next, all the algorithms except the BS-WR algorithm have been simulated using a variety of scenarios in which the number of robots and tasks ranged from 2 up to 20, and for every case one hundred simulations were run. These results are shown in Table 4.1 where, in each cell, the mean of the global cost and the error in percentage in comparison with the optimal solution are presented. The optimal solution has been calculated using the Hungarian method (Kuhn, 1955). In order to show the results clearly, only the error in percentage is shown in Figure 4.4. It can be observed that the best algorithm is the RTMA algorithm and the worst one is the BS algorithm. It is important to point out that all the algorithms obtain efficient results up to 8 robots and tasks where the largest error is less than 10%. For more than 8 robots, only the RTMA algorithm presents good results, with a maximum error of 5.98% in the case of 20 robots. As can be seen in Figure 4.4, the error with the optimal

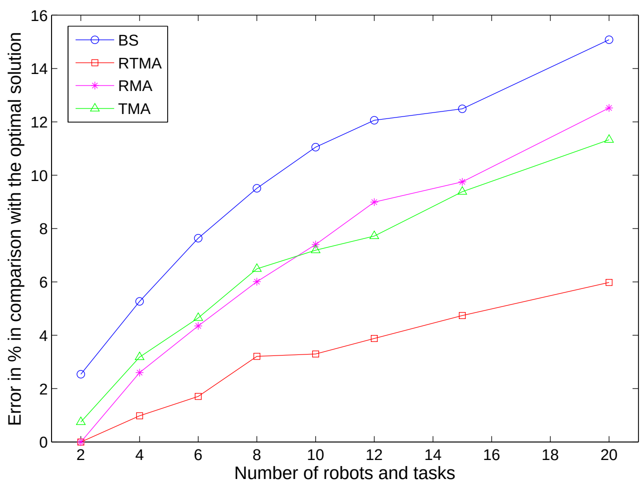


Figure 4.4: Mean error in percentage in comparison with the optimal solution for the different types of algorithms and calculating the initial positions of the robots and the locations of the tasks at random over 100 simulations.

solution increases linearly for all the algorithms with respect to the number of robots and tasks. However, the RTMA algorithm is the one with lowest slope. Furthermore, it is interesting to mention that with less than 10 robots the RMA algorithm obtains results slightly better than the TMA algorithm, but with over 10 robots the TMA algorithm obtains better results than the RMA. It is also important to point out that for 2 robots and tasks the RMA and RTMA algorithms always obtain the optimal solution.

The results of Table 4.1 only show statistically how good the algorithm is based on the mean. However, it could be the case that an algorithm could have good results on average but there are some situations where its results have large errors. Therefore, another important parameter to consider is the maximum error with respect to the optimal solution over all the simulations. In Figure 4.5, the maximal errors in percentage are shown. First of all, it can be observed that the RMA algorithm obtains worse maximal errors than the TMA algorithm and, in some cases, even worse than the BS algorithm, but the mean of the global cost is lower for the RMA algorithm as can be seen in Table 4.1. Therefore, the RMA algorithm has a better behavior on average but in certain circumstances the results can be worse than the TMA and BS

Tasks & Robots	***BS***	***TMA***	***RMA***	***RTMA***	***Optimum***
2	909.35	893.48	886.84	886.84	886.84
	(2.54%)	(0.75%)	(0.0%)	(0.0%)	
4	1473.52	1444.22	1436, 16	1413.45	1399.73
	(5.27%)	(3.18%)	(2.6%)	(0.98%)	
6	2020.13	1964.07	1958.49	1908.85	1876.77
	(7.64%)	(4.65%)	(4.35%)	(1.71%)	
8	2443.57	2376.03	2365.30	2302.90	2231.27
	(9.51%)	(6.49%)	(6.01%)	(3.21%)	
10	2865.81	2766.18	2771.63	2666.06	2580.65
	(11.05%)	(7.19%)	(7.4%)	(3.30%)	
12	3233.25	3108.09	3144.98	2997.34	2885.35
	(12.06%)	(7.72%)	(8.99%)	(3.88%)	
15	3749.97	3646.21	3658.48	3491.44	3333.55
	(12.49%)	(9.38%)	(9.75%)	(4.74%)	
20	4639.68	4488.53	4536.47	4272.99	4031.69
	(15.08%)	(11.33%)	(12.52%)	(5.98%)	

Table 4.1: Results computed over 100 simulations per each case. In each cell the mean of the global cost and the mean error in percentage with the optimal solution are presented.

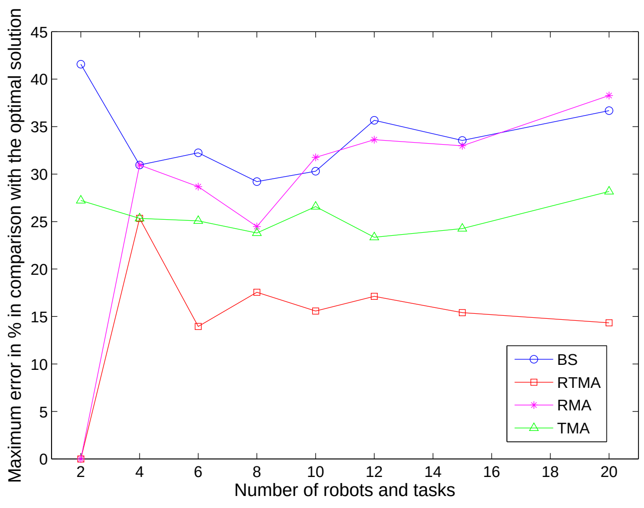


Figure 4.5: Mean of the maximum errors in percentage in comparison with the optimal solution in 100 simulations for the different types of algorithms. The initial positions of the robots and the locations of the tasks are calculated at random.

algorithms. On the other hand, the BS algorithm is still the worst one for most of the cases, while the RTMA algorithm presents the best results. As can be seen in Figure 4.5, the mean of the maximum errors considering all the cases is 14.91% for the RTMA algorithm and 33.77% for the BS algorithm (which is greater than the mean error commented in Table 4.1). That means these algorithms do not have a constant behavior and for a specific situation, results could be worse than the average.

All the results presented so far have been calculated using random position of the robots and random locations of the tasks uniformly distributed. However, the quality of the solution for some of the algorithms depends on how tasks and robots are randomly distributed. In Figure 4.6, there are two types of distributions: the one on the left is calculated totally at random and is the one used so far, the other formation on the right has a structure formed by two boxes. Most of the tasks and robots are in the small box, and the others outside the big box. As can be seen in Figure 4.7, the BS algorithm obtains worse results than the ones obtained with the other type of formation, specially for low number of robots and tasks. Another important characteristic of this type of distribution is that the error in percentage in comparison with the optimal solution remains almost constant for different number of robots and

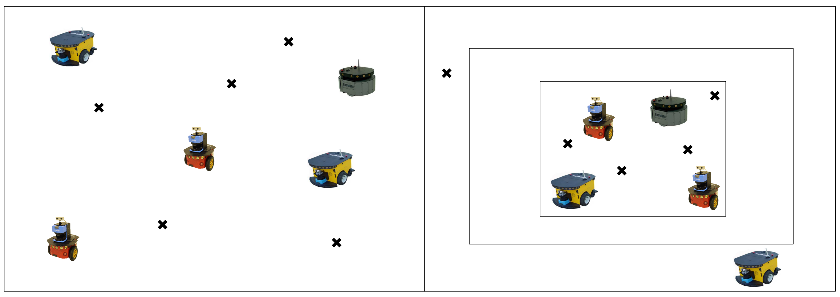


Figure 4.6: Types of distributions used in the simulations. Left: initial positions of the robots and task locations calculated at random. Right: most of the task locations and the initial positions of the robots calculated at random in the small box and the others calculated outside the big box and calculated also at random.

tasks. Therefore, for this type of formations, the behavior of the algorithms for a specific situation is more predictable than with the uniformly distributed random formations. Finally, the RTMA algorithm obtains also the best results while the BS algorithm the worst ones and, unlike the first type of formations, the TMA algorithm always obtains worse results than the RMA algorithm for all the simulated cases.

From these simulation results, it can be said that the algorithm efficiency depends on the random distributions that are used to calculate both robot and task locations. Also, the mean of the global cost is a representative value of the algorithm performance but more information is needed in order to completely understand the algorithm behaviors. Finally, it is not clear whether the RMA algorithm obtains better results than the TMA algorithm or viceversa. All these statements have a clear explanation in the next chapter where a theoretical study of MRST algorithms is presented.

4.4.2 Realistic simulations

In the last section, distributed task allocation algorithms have been simulated in scenarios without obstacles where task costs are calculated as the euclidean distance. Next, the task allocation algorithms are integrated within a robot architecture that couples the task allocation algorithm with navigation modules. The effect that the

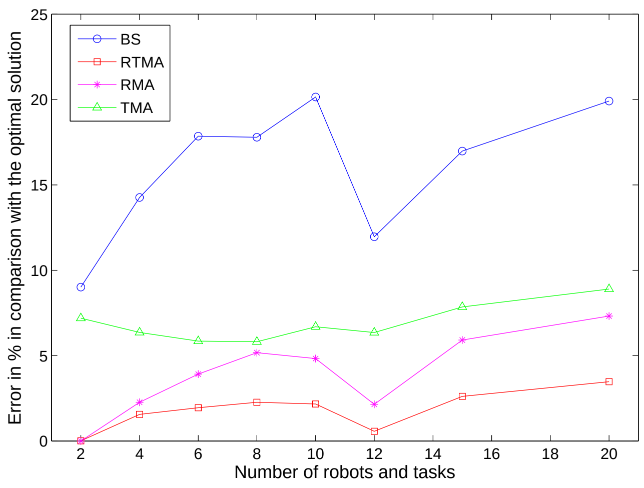


Figure 4.7: Mean error in percentage in comparison with the optimal solution for the different types of algorithms and calculating the initial positions of the robots and the locations of the tasks as is described in the right part of the Figure 4.6 over 100 simulations.

different modules, specially the path planner module, causes to the task allocation efficiency is studied.

Integration within a multi-robot architecture

The presented task allocation algorithms are integrated within a complete robotic system ready to be used in real world applications, using the multi-robot architecture based on modules (Maza et al., 2006) that was commented before. As can be seen in Figure 4.8, in each robot the task allocation algorithm has been integrated with a path planner algorithm and the execution of the tasks are within a behavior architecture that combines the path following algorithm with obstacle avoidance.

When the task allocation algorithm has to calculate the cost for a specific task, it sends the location data to the path planning module. Next, the path planner calculates the path using the information from an internal wold model and sends it back to the task allocation module as a list of points. A 2D grid has been used as the world model where each grid is considered as navigable or non-navigable.

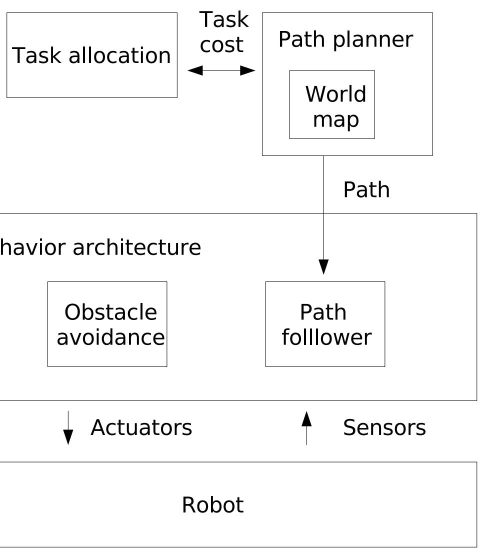


Figure 4.8: Scheme that shows the integration of a task allocation algorithm in a complete system ready to be used in a real world application. The path planning algorithm is used to calculate the cost of the tasks and as an input for the path follower algorithm which is combined with obstacle avoidance using the DAMN architecture.

Two of the most popular path planning algorithms have been implemented: A^* algorithm (Nilson, 1971) and RRTs (Rapidly-exploring Random Trees) (LaValle and Kuffner, 2001). These allow the system to integrate map-based information in the task allocation scenarios. The first algorithm is based on a heuristic estimator to find the optimal solution faster than general search algorithms such as breadth-first or depth-first search. Even so, for robotic applications, the A^* algorithm still requires a significant amount of processing power, specially for large state spaces with constraints. RRTs is also a search algorithm that has a random nature and the quality of the solution cannot be determined a priori, but it is faster than A^*. This algorithm works like a search tree that starts from an initial state and is expanded by performing incremental motions towards the direction of random points. The main difference between this algorithm and a random walk is that the latter suffers from a bias towards places already visited, while RRTs works in the opposite manner by being biased towards places not yet visited. Specifically, the bias version of RRTs with a probability equal to 0.05 has been used.

During the negotiation process, it is possible that the task allocation algorithm has to calculate several times the cost for the same task. For this reason, everytime the path planning module calculates the path for a task, it will save the path and its cost. In this way, the computation power and the calculation time for future requests have been reduced. Each task is identified by a unique sequence number.

After all tasks have been allocated, each robot starts the execution of its own. The path planning module sends the path to the path follower module which is combined with an obstacle avoidance behavior. Both behaviors are combined using a DAMN architecture (Rosenblatt, 1997). This architecture was designed to combine different behaviors, specially, for mobile robots in unknown and dynamic environments which fits the considered demonstration scenario. Each of the behaviors votes for a set of possible actuators values satisfying its objectives. Then, an arbitrator combines those votes and generates actions which reflects the behaviors objectives and priorities. Regarding the behaviors, a laser scanner was used as the sensor for the obstacle avoidance and the Pure Pursuit algorithm (Ollero and Amidi, 1991) has been used as the path follower. The Pure Pursuit algorithm geometrically determines the curvature that will drive the vehicle to a chosen path point defined as one lookahead distance from the current position of the robot.

Simulation results

In order to prove that the presented algorithms still obtain the same kind of efficiency in real world applications, the task allocation algorithms have been integrated in a complete system within the multi-robot simulator. Since these simulations are much more time consuming, only results for the BS and RTMA algorithms are presented. These two algorithms have been chosen since the RTMA algorithm obtains the best results and the BS algorithm is a simple algorithm that can be used as a benchmark.

Player/Gazebo (Gerkey et al., 2003) has been used to simulate the environment and the robots. An exploration application is considered where robots have to navigate towards some specific locations and take environmental measurements. As will be seen in the next section, the iRobot Create platform is used to test the MRST

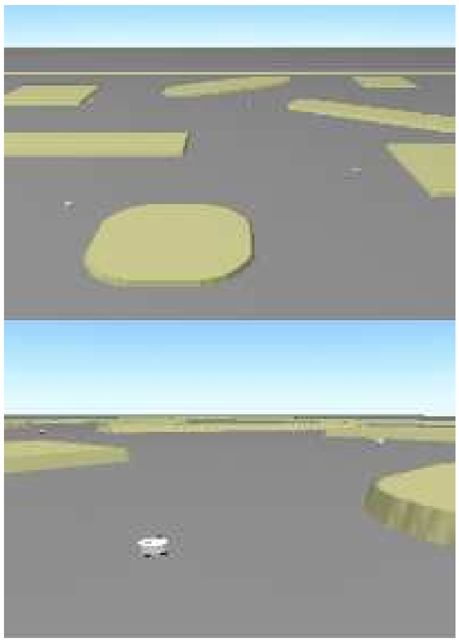


Figure 4.9: Snapshots of the simulator Player/Gazebo. At the top, an aerial view of the environment with obstacles. At the bottom, a close view of the 3D model of the test platform.

algorithms. For that reason, a 3D model of these robots has been created to be used in the simulator (see Figure 4.9).

After the integration of the task allocation algorithm within a robot architecture, the effect that individual robot errors causes to the task allocation efficiency is studied. There are different sources of errors: localization, path planning, etc. In this thesis, the effort has been concentrated to study the effect of the path planner module on the bidding process. Other sources of errors and their influences on the task allocation will be considered for future work.

First, global costs obtained with the two path planners (A^* and RRTs) are compared. Second it is studied, for each of the task allocation algorithms (BS and RTMA), whether the difference between global costs using both path planners are equivalent. Finally, it is interesting to study the effect that obstacle density has on the performance of the task allocation algorithms and whether differences depend on the path planner algorithm.

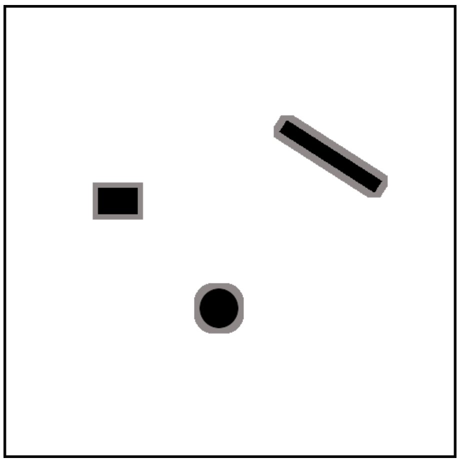


Figure 4.10: Scenario with 5% of non-navigable terrain. The obstacles are increased virtually the size of the robot, so they do not navigate too close to them. This reduce the probability of a collision due to noises and inaccuracies in the sensors and the map.

A classification based on the percentage of non-navigable terrain (in this case obstacles) has been made: high navigable terrains (less than 15% of non-navigable terrain), medium navigable terrain (between 15% and 30% of non-navigable terrain) and low navigable terrain (more than 30% of non-navigable terrain). For the simulations, three different scenarios have been used, all of them with a 75x75 m^2 area, to test the complete robotic system for this type of application. The first scenario (see Figure 4.10) has 5% of non-navigable terrain. The second scenario has 20% of non-navigable terrain (see Figure 4.11), and the last scenario has 40% of non-navigable terrain (see Figure 4.12). Also, Figures 4.11 and 4.12 show the solution obtained using the task allocation algorithms and the path followed by the robots using the RRTs and A^* algorithms respectively. It can be observed directly how the A^* obtains the optimal path while the RRTs has a lower rate of finding a path close to the optimal one. This fact has a large impact on the performance of the task allocation algorithms, as will be commented next.

Due to the complexity of these simulations, only 20 simulations have been run per case where the position of the robots and tasks have been calculated at random (avoiding the areas considered obstacles in the world). First, simulations were run

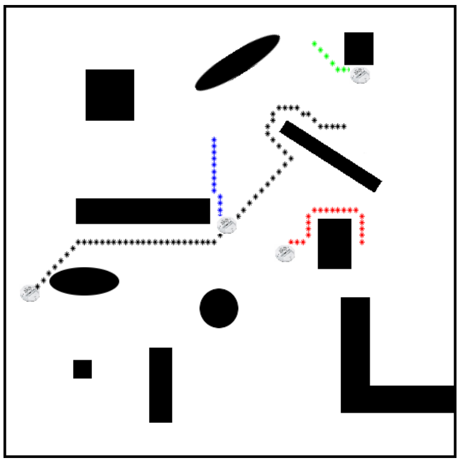


Figure 4.11: Scenario with 20% of non-navigable terrain. The paths show the solution of one of the random missions obtained using the BS task allocation algorithm with the A^* path planner.

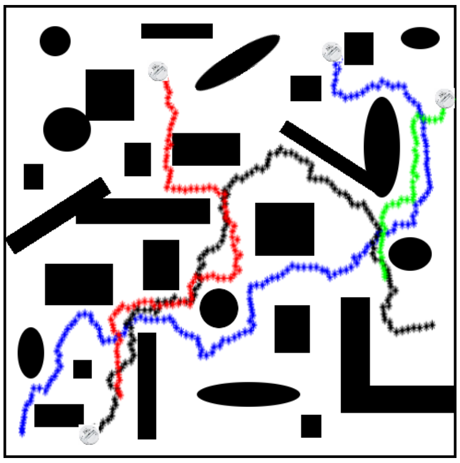


Figure 4.12: Scenario with 40% of non-navigable terrain. The paths show the solution of one of the random missions obtained using the RTMA task allocation algorithm with the RRTs path planner.

Tasks & Robots	***Scenario***	***BS***		***RTMA***		***Optimum***
		Mean	Error	Mean	Error	
4	5%	124.74	4.58%	119.99	0.60%	119.27
4	20%	127.38	8.70%	119.62	2.07%	117.19
4	40%	161.94	6.85%	153.61	1.35%	151.55
6	5%	144.02	6.54%	139.88	3.47%	135.18
6	20%	187.76	7.53%	177.25	1.51%	174.60
6	40%	216.88	7.08%	204.92	1.17%	202.54
8	5%	197.52	8.94%	186.72	2.98%	181.31
8	20%	208.08	9.16%	195.86	2.75%	190.61
8	40%	261.62	12.63%	235.74	1.48%	232.29

Table 4.2: Results computed over 20 simulations per each case using the A^* algorithm. Each cell represents the mean of the global cost and the mean percentage error in comparison with the optimal solution. The obstacles are distributed as Figure 4.10 for the 5% scenario, Figure 4.11 for the 20% scenario and Figure 4.12 for the 40% scenario.

using the A^* algorithm for path planning. The results obtained from these simulations are showed in Table 4.2 where each cell represents the mean of the global cost over 20 missions, i.e., the sum of the distance traveled by all the robots, and the error in percentage with the optimal solution. It can be seen that the RTMA algorithm still obtains better results than the BS algorithm when it is integrated in a complete robotic system. Also, the results obtained with the complete system are equivalent, in comparison with the optimal solution, to the results obtained in Section 4.4. The improvements obtained with the RTMA algorithm, in comparison with the BS algorithm, are of the same order of magnitude and both algorithms obtain similar results in all the scenarios. Therefore, the integration of the task allocation algorithms in a complete robotic system, with the A^* planner, does not affect the task allocation algorithms performance.

The optimal solution has been calculated using the A^* algorithm with the Hungarian method (Kuhn, 1955), i.e., all the different optimal paths between every robot and task have been calculated using the A^* algorithm, then the distance of all these

Tasks & Robots	***Scenario***	***BS***		***RTMA***		***Optimum***
		Mean	Error	Mean	Error	
4	5%	159.68	33.88%	155.91	30.72%	119.27
4	20%	155.49	32.68%	149.92	27.92%	117.19
4	40%	190.78	31.98%	186.34	29.00%	144.55
6	5%	172.02	27.25%	171.33	26.74%	135.18
6	20%	235.70	34.99%	223.96	28.27%	174.60
6	40%	290.44	43.39%	291.62	43.98%	202.54
8	5%	242.09	33.52%	229.44	26.54%	181.31
8	20%	258.40	35.56%	252.77	32.61%	190.61
8	40%	354.56	52.66%	345.69	48.82%	232.29

Table 4.3: Results computed over 20 simulations per each case using the RRTs algorithm. Each cell represents the mean of the global cost and the mean percentage error in comparison with the optimal solution.

paths have been used as the values of the cost matrix that represents the task allocation problem as a job assignment problem. Finally, the Hungarian method has been applied to that cost matrix to calculate the optimal assignment.

Next, the task allocation algorithms are tested with the RRTs instead of the A^* algorithm. The results are shown in Table 4.3. First, it can be observed that these results are worse than using the A^* algorithm. This makes sense since the RRTs algorithm does not ensure any kind of efficiency of the solution. Also, when RRTs are used, the differences between both algorithms decreases and there is even one case where the BS algorithm performs better than the RTMA algorithm.

In summary, it has been shown that the performance of the task allocation algorithms is better with the A^* algorithm rather than RRTs. Also, the use of RRTs reduce the advantages obtained with a more complex algorithm, such as the RTMA algorithm, and make the results of both algorithms very similar. The percentage of non-navigable terrain in the scenario seems to not affect the performance of the system and similar results have been obtained for the three different scenarios.

Finally, for this kind of application, where robots use an occupancy grid to navigate in 2D, the computational complexity of A^* is not high, and therefore, it is the best option. However, the RRTs algorithm should not be completely discarded since

A^* might be too slow to be applied in some scenarios with high dimensional state spaces with constraints.

4.5 Experiments with real robots

In this section two distributed task allocation algorithms (BS and RTMA), integrated within the explained robot architecture, are implemented in real robots. A significant number of experiments have been run using the team of robots (see Appendix B for more information about the multi-robot testbed). Results from experiments with and without obstacles are commented.

4.5.1 Results from experiments

Since experimentation is tedious and slow, only experiments with the BS and RTMA algorithms have been performed, but different numbers of robots have been considered. Specifically, four experiments have been run with two robots, six with four robots and eight with six robots. In total, 18 experiments have been run with each of the two algorithms. Different number of robots has been considered in order to illustrate the evolution of the presented algorithms when the number of robots increases.

The first set of experiments has been performed in an 10x10m^2 arena where the positions of the robots and tasks have been calculated at random. Although this experimental validation does not consider a large number of cases, it is important to verify that these algorithms can be implemented and used with real robots. Also, these experiments have the aim to show that the presented simulation results are still valid even when real robots are considered.

The results from the experiments are shown in Figure 4.13. It can be seen that these results follow the same trend as the simulation results where the RTMA algorithm obtains better results than the BS algorithm and the difference between both of them increases with the number of robots and tasks.

On the other hand, experiments with obstacles have been performed. The same testbed was used but with an area of 15x23m^2. Since the best results are obtained

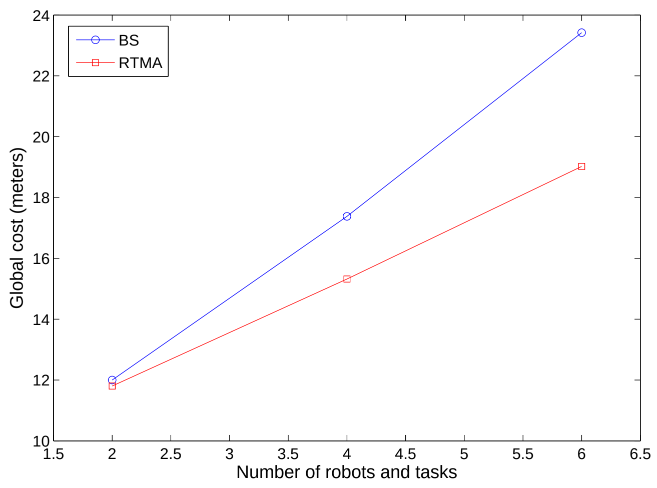


Figure 4.13: Results from the experiments in an arena of 10x10m^2 (mean of the global cost). The BS and RTMA algorithms have been tested with 2, 4 and 6 real robots, obtaining results similar to the simulated ones.

with the A^* algorithm, only experiments with this path planner were performed. Different numbers of robots were considered. Again, four experiments have been run with two robots, six with four robots and eight with six robots. In total, 18 experiments have been run with each of the two algorithms. All these experiments have been performed in the described arena where the positions of the robots and tasks have been calculated at random avoiding the areas with obstacles.

The results from the experiments are shown in Figure 4.15. It can be seen that these results obtain similar characteristics to the ones obtained in simulation where obstacles were considered. It can be observed how the difference between both algorithms increases with the number of robots, and again, the RTMA algorithm obtains better results than the BS algorithm.

4.6 Chapter summary

This chapter presented five different MRST task allocation algorithms. The first two are a basic implementation of a market approach and it has been demonstrated the

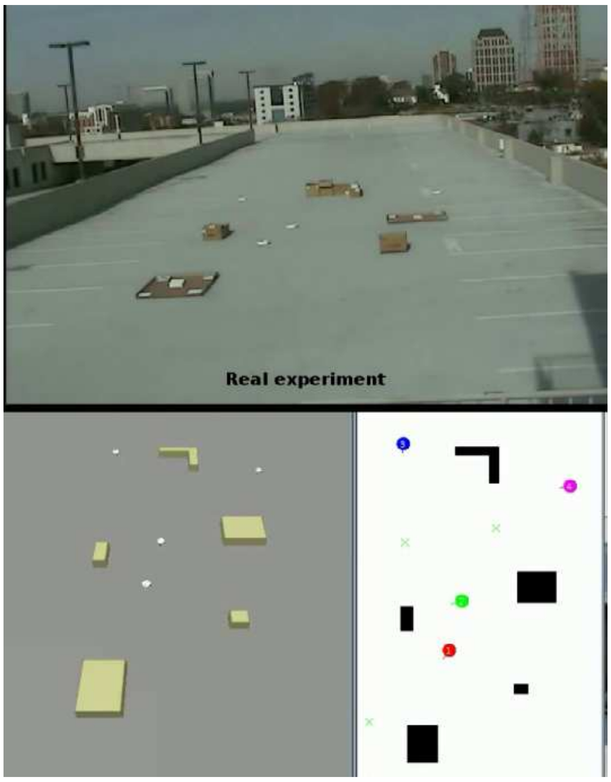


Figure 4.14: Team of robots running one of the experiments in an arena of 15x23m^2 with obstacles. Visual interface used to follow the experiments.

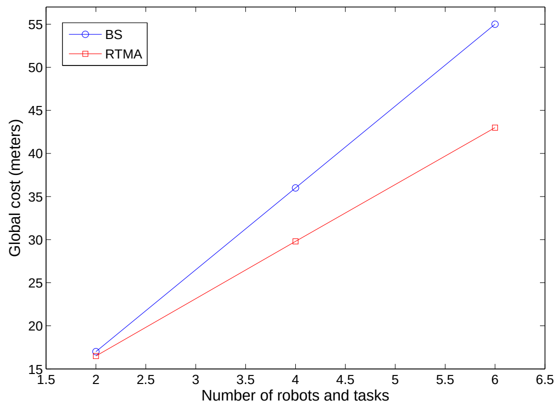


Figure 4.15: Results from the experiments in an arena of 15x23m^2 with obstacles (mean of the global cost). The BS and RTMA algorithms have been tested with 2, 4 and 6 real robots integrated in a complete robotic system including the A^* algorithm as path planner.

importance of reallocations since it improves the quality of the solution and makes the final allocation independent from the task announcements order.

Next, the BS algorithm has been compared with three improved algorithms. The BS algorithm is the simplest algorithm but obtains the worst results in most of the cases. Also, it is the algorithm that is most affected by the structure of the formation due to the fact that its results get worse with the second type of formations. The RMA and TMA algorithms use the mean of the costs (considering all the tasks associated to a robot or all the robots for a specific task) in order to increase the information about the whole system and improve the results, but always keeping the distributed computation of the algorithm. These two algorithms obtain similar results for all the cases and better than the ones obtained with the first algorithm. Finally, the RTMA, which is a combination of the RMA and TMA algorithms, obtains the best results in all the cases for both types of formations since combines the good characteristics of the RMA and TMA algorithms.

Two different types of formations have been used. In the first one, the error in comparison with the optimal solution increases in a linear way with respect to the number of robots and tasks. In the second one, the error is kept slightly constant. Therefore, the behavior of the algorithms is more predictable for the second type of formations. Finally, the BS and RTMA algorithms have been validated in realistic simulations and experiments with real robots obtaining results similar to the simulations.

Chapter 5

Performance evaluation of MRST market-based algorithms

This chapter proposes a probabilistic analysis approach for MRST (Multiple Robots Single Task) market-based algorithms that can be used to compare different algorithms in different scenarios without the need of simulations or experiments. The probabilistic analysis is used to calculate the expected value of the global cost, which at the same time is used as a metric to compare different algorithms. The probabilistic analysis is general and does not require any supposition, but the probabilistic distribution of the costs should be known.

The chapter is structured as follows. After a description of previous works, each task allocation algorithm is related with an equivalent centralized greedy algorithm that is applied to a matrix representation of the problem. Next, a probabilistic analysis of the algorithms is presented where a general formula for the expected value of the global cost for any cost distribution is calculated. Moreover, the results of the analysis are applied to three different scenarios: random, dispersion and extension scenarios. These results are validated with simulations and real experiments.

On the other hand, the performance of the task allocation algorithms is studied and compared with the optimal solution. Finally, an extension of the probabilistic approach is developed to calculate the variance of the global cost. This result is used to model the distribution of the global cost using a normal distribution.

5.1 Introduction

Although the efficiency of market-based algorithms has been evaluated in both simulation and some real implementations (Dias et al., 2004; Gerkey and Matarić, 2002; Zlot and Stentz, 2006), none of these works has obtained a theoretical estimation on the real efficiency of these algorithms. As far as I know, the only works that obtain a performance bound for a market-based algorithm is (Lagoudakis et al., 2004). But, their implementation differs from the classical market-based approach (used in this thesis) and computes the costs using the information of all the tasks plus the local information of the robot. Also, a generic framework for market-based algorithms that studies theoretical guarantees for different bidding rules and different team objectives can be found in (Lagoudakis et al., 2005). There are other recent works on theoretical bounds in (Smith and Bullo, 2007; Yun and Rus, 2007; Zavlanos and Pappas, 2007) but their algorithms are not based on auctions. (Yun and Rus, 2007) explains a distributed heuristic with local communication, while in (Zavlanos and Pappas, 2007) the agents are controlled by hybrid models using distributed potential fields. In (Smith and Bullo, 2007), a solver of the Traveling Salesman Problem (TSP) is used to decide which robot should execute which task. An important point to note here is that the above-mentioned analyses derive a worst-case bound, a "pessimistic" bound since it is computed assuming that the worst scenario occurs, which may be an unlikely event; actually, it may not ever happen in a real experiment.

The work presented in this thesis is novel in the sense that it calculates the expected value of the global cost, which is proposed as an estimate of the algorithm efficiency. This descriptor is more informative than the worst-case value since it is based on the algorithm average behavior (the most likely event). Moreover, as will be shown, the expected global cost provides a calculation of the performance over time.

5.2 Relation between market-based and greedy algorithms

In this section, the equivalence between a distributed market-based algorithm and a centralized greedy algorithm is described. This equivalence allows to apply the probabilistic analysis to the greedy algorithm and extend the results to the distributed task allocation algorithm.

The MRTA problem is modeled as a cost matrix where each element is the cost associated with the respective robot and task. Each of the distributed MRST algorithms is related with a greedy algorithm applied to the cost matrix representation. Next, it will be shown how both the greedy and distributed market-based algorithms obtain the same allocation, and therefore, the same global cost. This is true when it is assumed that all robots are in communication range and no robot has a failure. These assumptions were also considered in previous works that studied the performance of market-based algorithms such as (Lagoudakis et al., 2004).

5.2.1 Algorithm without reallocations

BS-WR: BaSic market-based algorithm Without Reallocations

This algorithm is equivalent to the column-scan method (Kurtzberg, 1962) for the assignment problem expressed as a matrix where each element is the cost associated with the respective robot and task. It is considered that tasks are the columns of the cost matrix and robots the rows. In this algorithm, each column of the matrix is examined and the row with the lowest cost is selected. The selected row is marked and no longer examined for the rest of the algorithm. Through this process, the algorithm functions as follows:

1. Each column is scanned.
2. The smallest element of the column is selected.
3. The column and the row associated to this element are deleted and not considered for the rest of the algorithm.

4. The same procedure is repeated for the next column until all the columns have been scanned.

5. The selected elements represent the costs of the final allocation and the global cost is the sum of these elements.

A simple example will be used to show how both algorithms obtain the same solution:

- The initial positions of the robots and the desired locations of the tasks are the ones show in Figure 5.1.

- The matrix that models this specific problem is:

$$\begin{pmatrix} 30.0 & 41.23 & 20.0 \\ 50.0 & 10.0 & 44.72 \\ 80.0 & 72.11 & 30.0 \end{pmatrix}$$

- Following the algorithm steps, the smallest element of the first column is selected. This element is 30.0 which assigns robot A with task number 1. The row and column of the selected element is deleted and the following matrix is obtained:

$$\begin{pmatrix} 10.00 & 44.72 \\ 72.11 & 30.0 \end{pmatrix}$$

- Next the smallest element of the second column is selected. This element is 10.0 and therefore, robot B is assigned to task number 2.

- Finally, the last assignment is made such that robot C is assigned to task number 3. The global cost for this problem is $GC(3) = 70.0$.

As can be observed in Figures 5.1 and 5.2, the solution obtained with the BS-WR algorithm is exactly the same as the one obtained by the column-scan method. Thus, both algorithms obtain the same global cost, and therefore, its same expected value. As was commented in the last chapter, it can be seen clearly how a change in the order of the tasks will produce a different allocation. For example, if Task 3 is announced

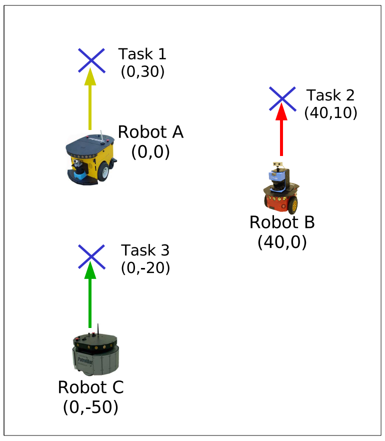


Figure 5.1: Initial position of the robots and the desired locations of the tasks, and also, the final assignment obtained with the BS-WR algorithm.

before Task 1, the final allocation will be the same as Figure 5.4. This is not a desirable characteristic since it is difficult to predict the performance of the final allocation.

Validation using simulations

It has been illustrated how the BS-WR algorithm is equivalent to a centralized greedy algorithm (column-scan algorithm). In order to strengthen this fact, the distributed and greedy algorithms have been simulated with the same random missions. In Figure 5.3 can be observed how the BS-WR and the greedy algorithm always obtain the same global cost for each of the simulations. This means that both algorithms have allocated the same tasks to the same robots for all simulations.

5.2.2 Algorithms using reallocation mechanisms

The rest of distributed MRST algorithms are equivalent to the matrix-scan algorithm (Kurtzberg, 1962). These algorithms make use of reallocations which in the greedy algorithm is translated in the need to consider all the costs before a task allocation is executed. Moreover, as it will be seen next, the only differences between the equivalent greedy algorithms are how costs are calculated.

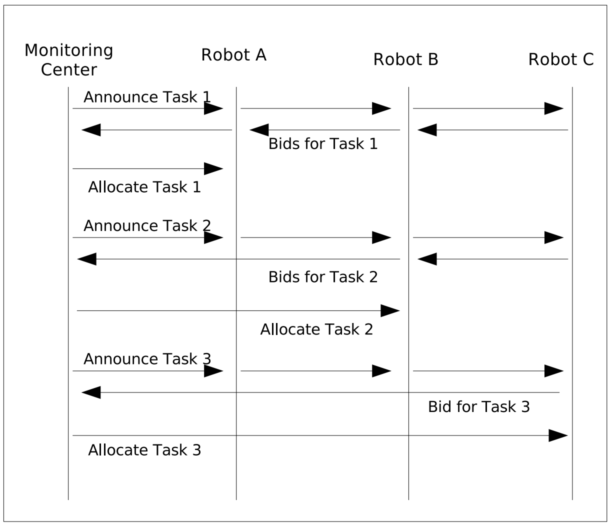


Figure 5.2: Messages exchanged in the auction process among the different robots using the BS-WR algorithm. The initial positions of the robots and the locations of the tasks are the same as Figure 5.1.

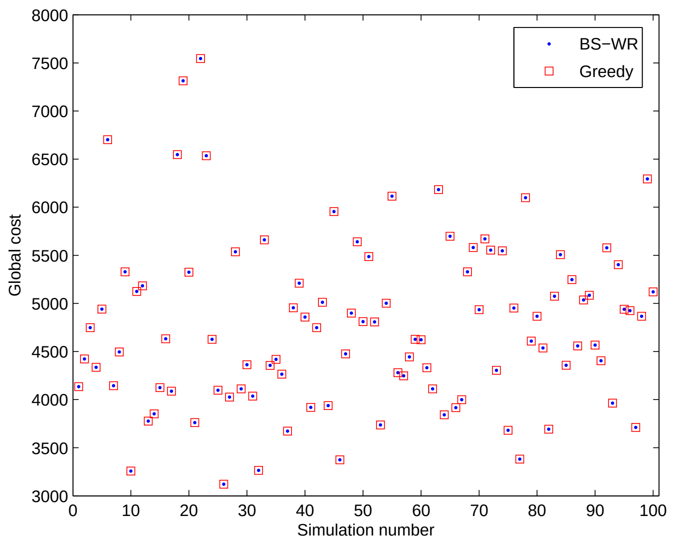


Figure 5.3: Solution for 100 simulations where the position of the robots and tasks have been calculated at random in a 1000mx1000m area. The BS-WR and the column-scan (greedy) algorithms obtain the exactly same solutions.

BS: BaSic market-based algorithm

As was stated in the previous chapter, the BS algorithm uses reallocations to obtain the same solution no matter the order in which tasks are announced. This algorithm is equivalent to the matrix-scan algorithm that solves the assignment problem when it is expressed in a matrix form, and it works as follows:

1. The smallest element of the entire matrix is selected.
2. The row and column associated with this element are deleted and therefore, the order of the matrix is reduced by one.
3. The matrix is searched again for the smallest element and the process is repeated until a matrix of order one is reached.
4. The selected elements represent the costs of the final allocation and the global cost is the sum of them.

The use of reallocations in the BS algorithm ensures that it will obtain the same solutions as the matrix-scan algorithm. This fact is illustrated with the following example:

- The initial positions of the robots and the desired locations of the tasks are the ones show in Figure 5.4.
- Supposing that the columns represent tasks and the rows robots, the matrix that models this specific problem is:

$$\begin{pmatrix} 30.0 & 41.23 & 20.0 \\ 50.0 & 10.0 & 44.72 \\ 80.0 & 72.11 & 30.0 \end{pmatrix}$$

- Following the algorithm steps, the smallest element of the matrix is selected. This element is 10.0 which assigns robot B with task number 2. The row and

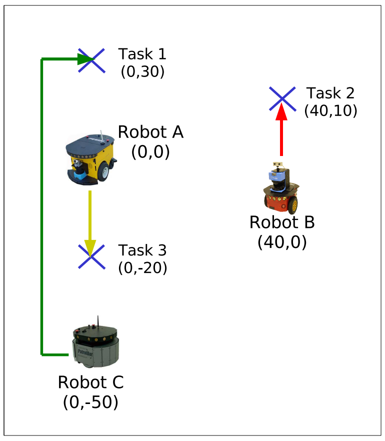


Figure 5.4: Initial position of the robots and the desired locations of the tasks, and also, the final assignment obtained with the BS algorithm.

column of the selected element is deleted and the following matrix is obtained:

$$\begin{pmatrix} 30.0 & 20.0 \\ 80.0 & 30.0 \end{pmatrix}$$

- Again the smallest element of the new matrix is selected. This element is 20.0 and, therefore, robot A is assigned to task number 3.
- Finally, the last assignment is made such that robot C is assigned to task number 1. The global cost is $GC(3) = 110.0$.

Figure 5.5 illustrates the increase in messages exchanged among the robots running the BS algorithm for the problem stated in Figure 5.4. As can be observed, the solution obtained with the BS algorithm is exactly the same as the one obtained by the matrix-scan method. Thus, both algorithms obtain the same expected value of the global cost.

RMA: Robot Mean Allocation algorithm

The RMA algorithm relates also to the matrix-scan method, but each cell has to represent the negative of the RMA cost. This value is the original cost minus the

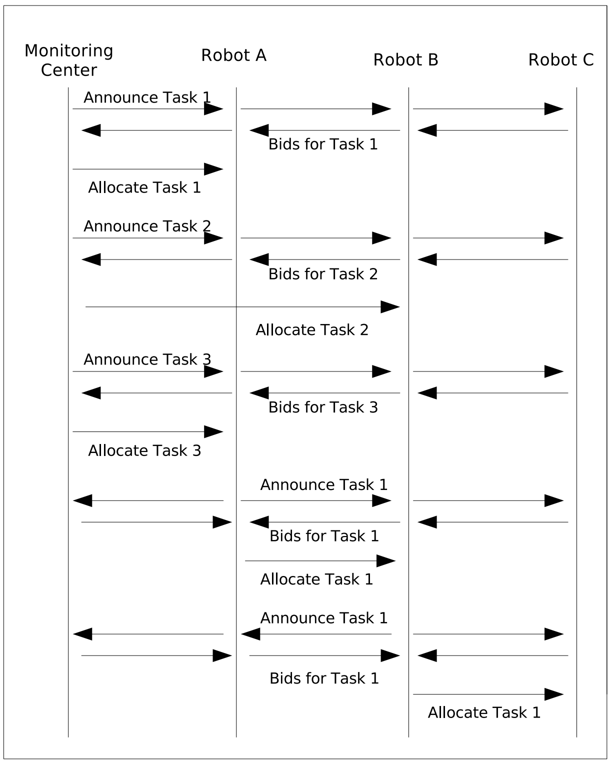


Figure 5.5: Messages exchanged in the auction process among the different robots using the BS algorithm. The initial positions of the robots and the locations of the tasks are the same as Figure 5.4.

mean of the costs considering one task and all the robots, i.e., the mean of all the costs of one column. Therefore, the representation of the cost matrix is

$$\begin{pmatrix} x_{11}-\bar{x}_{\cdot 1} & x_{12}-\bar{x}_{\cdot 2} & \cdots & x_{1t}-\bar{x}_{\cdot t} \\ x_{21}-\bar{x}_{\cdot 1} & x_{22}-\bar{x}_{\cdot 2} & \cdots & x_{2t}-\bar{x}_{\cdot t} \\ \cdot & & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & & \cdot & \cdot \\ x_{r1}-\bar{x}_{\cdot 1} & x_{r2}-\bar{x}_{\cdot 2} & \cdots & x_{rt}-\bar{x}_{\cdot t} \end{pmatrix}$$

where r is the number of robots, t the number of tasks, and $\bar{x}_{\cdot j}$ is the mean of the costs for task j considering all the robots, i.e., $\bar{x}_{\cdot j} = \sum_{i=1}^{r} x_{ij}/r$.

An example is used to illustrate the relation between the RMA algorithm and the matrix-scan algorithm applied to the RMA cost matrix.

- The initial positions of the robots and the desired locations of the tasks are the ones show in Figure 5.6.

- The cost matrix that models this specific problem following the definition described above is:
$$\begin{pmatrix} -23.33 & -0.12 & -11.57 \\ -3.33 & -31.11 & -13.15 \\ 26.67 & 30.99 & -1.57 \end{pmatrix}$$

- Following the matrix-scan algorithm steps, robot B is assigned to task number 2, robot A to task 1, and robot C to task 3. The global cost for this problem is $GC(3) = 70.0$.

As can be observed in Figures 5.6 and 5.7, the solution obtained with the RMA algorithm is exactly the same as the one obtained by the matrix-scan algorithm using the RMA matrix definition.

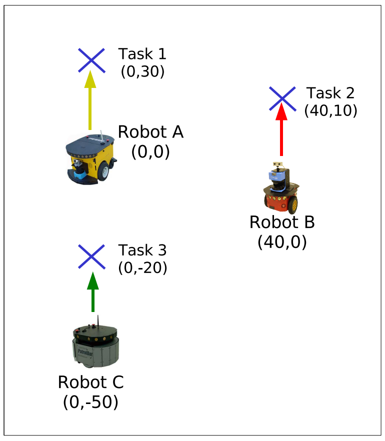


Figure 5.6: Initial position of the robots and the desired locations of the tasks, and also, the final assignment obtained with the RMA, TMA and RTMA algorithms.

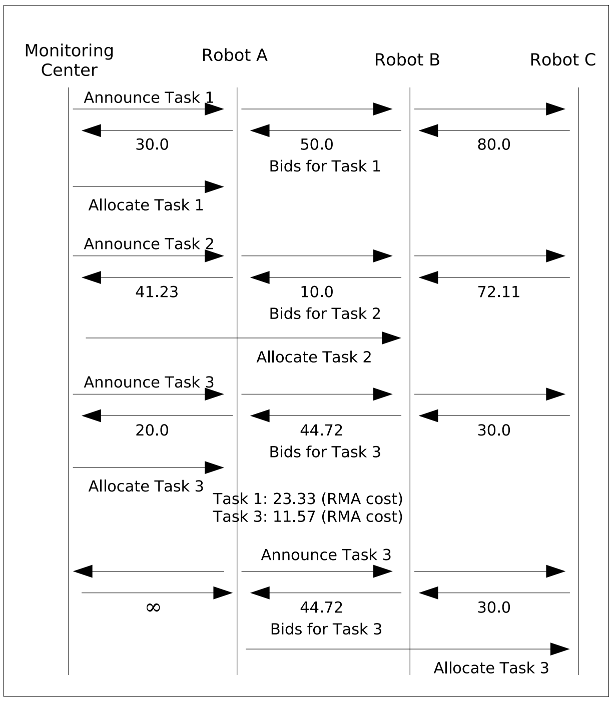


Figure 5.7: The auction process using the RMA algorithm for the initial positions of robots and tasks as shown in Figure 5.6.

TMA: Task Mean Allocation algorithm

Like in the RMA algorithm, the cost matrix for the TMA algorithm needs to be changed and becomes:

$$\begin{pmatrix} x_{11}-\bar{x}_{1\cdot} & x_{12}-\bar{x}_{1\cdot} & \cdots & x_{1t}-\bar{x}_{1\cdot} \\ x_{21}-\bar{x}_{2\cdot} & x_{22}-\bar{x}_{2\cdot} & \cdots & x_{2t}-\bar{x}_{2\cdot} \\ \cdot & & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & & \cdot & \cdot \\ x_{r1}-\bar{x}_{r\cdot} & x_{r2}-\bar{x}_{r\cdot} & \cdots & x_{rt}-\bar{x}_{r\cdot} \end{pmatrix}$$

where $\bar{x}_{i\cdot}$ is the mean of costs for robot i considering all the tasks, i.e., $\bar{x}_{i\cdot} = \sum_{j=1}^{t} x_{ij}/t$. In this case, each element of the matrix represents the TMA cost explained in Chapter 4.

In order to obtain the same results as the TMA algorithm, the original matrix-scan method has to be applied to the new definition of the cost matrix, which is illustrated in the following example:

- The initial positions of the robots and the desired locations of the tasks are the ones show in Figure 5.6.
- The matrix that models this specific problem following the definition for the TMA algorithm is:

$$\begin{pmatrix} -0.41 & 10.82 & -10.41 \\ 15.09 & -24.91 & 9.81 \\ 19.30 & 11.41 & -30.70 \end{pmatrix}$$

- Following the matrix-scan algorithm steps, robot C is allocated to task 3, robot B to task 2, and robot A to task 1.

As can be observed in Figure 5.6 and 5.8, the solution obtained with the TMA is exactly the same as the one obtained by the matrix-scan method using the TMA cost matrix definition.

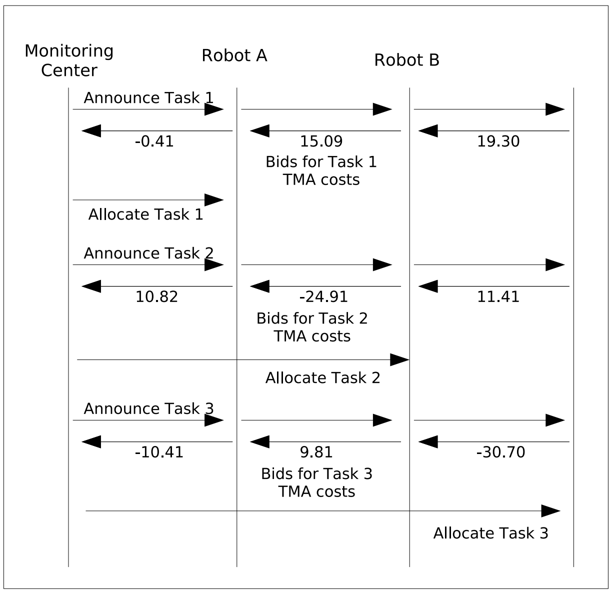


Figure 5.8: Messages exchanged in the auction process among the different robots using the TMA algorithm. The initial positions of the robots and the locations of the tasks are the same as depicted in Figure 5.6.

It is interesting to point out that the new cost matrix definitions, for the RMA and TMA algorithms, are very similar. From the greedy algorithm point of view, both algorithms are almost the same with the only difference of how the mean of the costs is defined. Supposing that costs are equally distributed, it is obvious to observe that both algorithms obtain the same results for a large number of robots and tasks, since both mean definitions will obtain close values. This fact answer the question stated in the previous chapter, where it was not obvious which algorithm performs best. Actually, both algorithms have the same behavior, and they only differ when the number of robots and tasks is low since the sample mean of a small number of observations has a high level of variation.

RTMA: Robot Task Mean Allocation algorithm

A similar equivalence can be obtained with the RTMA algorithm. In this case the problem is represented in the following matrix form:

$$\begin{pmatrix} w_{11} - \bar{w}_{\cdot 1} & w_{12} - \bar{w}_{\cdot 2} & \cdots & w_{1t} - \bar{w}_{\cdot t} \\ w_{21} - \bar{w}_{\cdot 1} & w_{22} - \bar{w}_{\cdot 2} & \cdots & w_{2t} - \bar{w}_{\cdot t} \\ . & & & . \\ . & . & & . \\ . & & . & . \\ w_{r1} - \bar{w}_{\cdot 1} & w_{r2} - \bar{w}_{\cdot 2} & \cdots & w_{rt} - \bar{w}_{\cdot n} \end{pmatrix}$$

where $w_{ij} = x_{ij} - \bar{x}_{i\cdot}$ and $\bar{w}_{\cdot j}$ is defined as

$$\bar{w}_{\cdot j} = \sum_{i=1}^{r} \frac{w_{ij}}{r} = \sum_{i=1}^{r} \frac{x_{ij} - \bar{x}_{i\cdot}}{r}.$$

Applying the new cost matrix definition to the previous example (see Figure 5.6), it is obtained

$$\begin{pmatrix} -11.74 & 3.76 & 7.97 \\ 11.71 & -24.02 & 12.30 \\ 0.02 & 20.24 & -20.27 \end{pmatrix} \tag{5.1}$$

Following the same steps used for the other two algorithms, it can be easily seen that the RTMA algorithm obtains the same results as the the matrix-scan algorithm applied to matrix (5.1).

Validation using simulations

It has been illustrated how the distributed task allocation algorithms are equivalent to centralized greedy algorithms. In order to strengthen this fact, the distributed MRST and greedy algorithms have been simulated with the same random missions. In Figure 5.9 can be observed how the RTMA and the greedy algorithm always obtain the same global costs for each of the simulations. This means that both algorithms have allocated the same tasks to the same robots for all simulations.

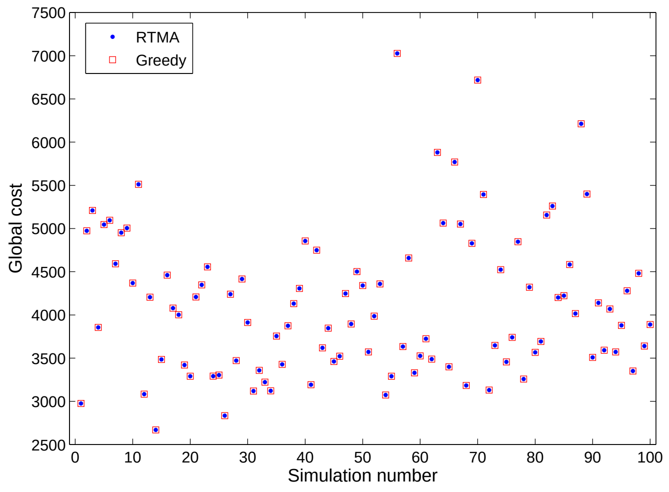


Figure 5.9: Solution for 100 simulations where the position of the robots and tasks have been calculated at random in a 1000mx1000m arena. The RTMA and its equivalent greedy algorithms obtain the exactly same solutions.

Similar simulations have been performed with the BS, RMA and TMA algorithms to validate the equivalence with their respective greedy algorithms.

5.3 Probabilistic analysis of MRST market-based algorithms

A probabilistic approach is used to analyze the different MRST algorithms. The key idea is to calculate the expected value of the global cost that provides an estimate of the algorithm performance over time. The probabilistic analysis is applied to the greedy algorithms but is valid also for the distributed market-based algorithms based on solution equivalence (as discussed in Section 5.2). This probabilistic approach is applied to the BS-WR and BS algorithms. These two algorithms cover both categories of algorithms, with and without reallocations. For the rest of MRST algorithms (RMA, TMA and RTMA), a similar procedure can be used. The only difference in comparison with the BS algorithm is that the cost distribution becomes more complex.

In the probabilistic analysis approach, the MRTA problem is formulated as a cost matrix where the different costs are modeled as random variables independent and equally distributed. The independence between costs is a reasonable assumption since the assignment of a robot and task position do not have an effect on the rest of the costs. In the first two sections, the costs follow the same distribution (equally distributed random variables), while in the last section this assumption is relaxed by considering a mixture of random variables.

Finally, it is important to clarify the behavior of the task allocation algorithms considering in this chapter when the number of robots (r) and tasks (t) are different. When the number of robots is higher than the number of tasks, the allocation process is developed as usual, but $r-t$ robots will not be allocated with any task and they will wait for more task announcements (idle state). On the other hand, when the number of robots is lower than the number of tasks, all the tasks will be announced but only one task is allocated per robot, and therefore, $t-r$ tasks will not be allocated. In order to simplify the analysis, it is assumed that the non-allocated tasks are deleted and not considered in future allocations.

5.3.1 BS-WR: Basic Market-Based Algorithm Without Reallocations

As was explained before, the global costs for the BS-WR algorithm and the column-scan method are the same. For the column-scan method the global cost is defined as $\sum_{k=1}^{Q} m_k$ where m_k is the minimum element of the k^{th} column. The k^{th} column has $r-k+1$ elements where $k \in [1, Q]$ and $Q = \min\{r, t\}$. A general case has been considered where the cost matrix has r (robots) rows and t (tasks) columns.

In the probabilistic analysis approach, the cost matrix is formed by equally distributed and independent random variables X_{ij}. The variable M_k is defined as the minimum of $r-k+1$ independent and equally distributed random variables ($X_{i,k}$) of the k^{th} column, i.e.,

$$M_k \equiv \min\{X_{1,k}, X_{2,k}, X_{3,k}, \dots, X_{(r-k+1),k}\}.$$

It is considered that m_k is an observation of the random variable M_k, and therefore, the random variable that represents the global cost is defined as

$$GC = \sum_{k=1}^{Q} M_k. \tag{5.2}$$

Next, the expected value of GC is computed.

The cumulative distribution function of M_k is given by

$$\begin{aligned} F_{M_k}(x) = P(M_k \leq x) &= \\ P(\min\{X_{1,k}, X_{2,k}, \dots, X_{(r-k+1),k}\} \leq x) &= \\ 1 - P(X_{1,k} > x, X_{2,k} > x, \dots, X_{(r-k+1),k} > x) &= \\ 1 - [1 - F_X(x)]^{r-k+1}, & \end{aligned} \tag{5.3}$$

since all the random variables are equally distributed and independent. $F_X(x)$ is the cumulative distribution function of each of the random variables, $X_{i,k}$, $i \in [1, r-k+1]$. The resulting probability density function of M_k is

$$f_{M_k}(x) = (r-k+1)\,[1 - F_X(x)]^{r-k}\, f_X(x) \tag{5.4}$$

where $f_X(x)$ is the probability density function for each of the random variables, $X_{i,k}$.

Once the probability density function is known, the expected value of M_k can be calculated as

$$E(M_k) = \int_{-\infty}^{\infty} x f_{M_k}(x)\, dx. \tag{5.5}$$

Finally, the expected value of the global cost is defined as the expected value of the sum of the n random variables $\{M_1, M_2 \dots, M_n\}$. Since the expected value is a linear operator

$$E_{GC}(Q) = E(M_1 + M_2 + \dots M_Q) = \sum_{k=1}^{Q} E(M_k). \tag{5.6}$$

Thus,

$$E_{GC}(Q) = \sum_{k=1}^{Q} \int_{-\infty}^{\infty} x \cdot (r-k+1) \left[1 - F_X(x)\right]^{r-k} f_X(x)\, dx \tag{5.7}$$

where, as was commented before, $F_X(x)$ is the cumulative distribution function and $f_X(x)$ is the probability density function of any of the $r \cdot t$ random variables that form the cost matrix that models the MRTA problem. Therefore, in order to calculate the expected value of the global cost for the BS-WR algorithm it is just needed to know which distribution the costs follow.

5.3.2 BS: Basic Market-Based Algorithm

As was shown before, the global costs for the BS algorithm and the matrix-scan method are the same. From Section 5.2.2, it is known that in order to compute the global cost related to the matrix-scan method, the minimum value of the matrix of order $r \times t$ has to be calculated. Next, the row and the column of the minimum is removed and a matrix of order $(r-1) \times (t-1)$ is obtained. This process is continued until the matrix does not have any more rows, columns or both. The global cost is the sum of the calculated minimums. As was stated before, the cost matrix is formed initially with $r \cdot t$ independent and equally distributed random variables, $\{X_1, X_2, \ldots, X_{rt}\}$. In order to simplify the explanation, the notation of the random variables has been changed such that they are specified as a vector of size $r \cdot t$. The variable M_1 is defined as $M_1 \equiv \min\{X_1, X_2, \ldots, X_{rt}\}$. Then, the cumulative distribution function is given by

$$F_{M_1}(x) = 1 - \left[1 - F_X(x)\right]^{rt}. \tag{5.8}$$

The expected value, μ_1, is then determined as

$$\mu_1 = E(M_1) = \int_{-\infty}^{\infty} x \cdot \frac{d}{dx} F_{M_1}(x)\, dx = \int_{-\infty}^{\infty} x \cdot rt \left[1 - F_X(x)\right]^{rt-1} f_X(x)\, dx \tag{5.9}$$

where $F_X(x)$ and $f_X(x)$ are the cumulative and density functions of each of the random variables X_i, where $i \in [1, rt]$. Next, $(rt-1)$ variables (those which are located at the same row and column as the minimum) are removed, and afterwards, the minimum value of the new matrix with order $(r-1) \times (t-1)$ is calculated again. This will lead to a new variable M_2 defined as the minimum of the left variables, $M_2 \equiv \min\{X_1, X_2, \ldots, X_{(r-1)(t-1)}\}$. Notice that M_2 depends on the first minimum selected M_1, and therefore, the expected value of M_2 can be computed as

$$\mu_2 = E(M_2) = E(E(M_2|M_1)). \tag{5.10}$$

The variable $h_{M_2}(M_1)$ is defined as $h_{M_2}(M_1) = E(M_2|M_1)$. It is important to take into account that $h_{M_2}(M_1)$ is a random variable depending on M_1 and not a real number, and thus, $E(M_2)$ will be a function depending on $E(M_1)$. The expected value of the global cost is defined as the expected value of the sum of $\{M_1, M_2 \ldots, M_Q\}$ where Q is equal to $\min\{r, t\}$. Next, the previous procedure is repeated to obtain

$$\begin{aligned} E_{GC}(Q) = E(M_1 + M_2 + \ldots M_Q) &= \\ \sum_{k=1}^{Q} E(M_k) = \mu_1 + \sum_{k=2}^{Q} E(h_{M_k}(M_{k-1})). \end{aligned} \tag{5.11}$$

(5.11) simplifies notably if $h_{M_k}(\cdot)$ is a linear function of the form $h_{M_k}(x) = a_k x + b_k$, for $k \in [2, Q]$. Then,

$$\begin{aligned} \mu_k = E(M_k) = E(h_{M_k}(M_{k-1})) &= \\ h_{M_k}(E(M_{k-1})) = h_{M_k}(\mu_{k-1}). \end{aligned}$$

And (5.11) becomes

$$E_{GC}(Q) = \mu_1 + \sum_{k=2}^{Q} h_{M_k}(\mu_{k-1}).$$

Notice that μ_k can be expressed in a recursive way as a function of μ_1

$$\begin{aligned} \mu_k = h_{M_k}(\mu_{k-1}) = a_k\mu_{k-1} + b_k & = \\ a_k\left(a_{k-1}\mu_{k-2} + b_{k-1}\right) + b_k = \dots & = \\ \mu_1 \prod_{i=2}^{k} a_i + \sum_{i=2}^{k-1} \left(b_i \cdot \prod_{j=i+1}^{k} a_j \right) + b_k. & \end{aligned}$$

Finally, the expected value of the global cost can be expressed as

$$E_{GC}(Q) = \mu_1 + \sum_{k=2}^{Q} \left[\mu_1 \prod_{i=2}^{k} a_i + \sum_{i=2}^{k-1} \left(b_i \cdot \prod_{j=i+1}^{k} a_j \right) + b_k \right]. \tag{5.12}$$

5.3.3 Extension

In this section, a generalization of the previous probabilistic analysis is proposed by considering that costs follow a mixture of distributions. This implies that costs follow different distributions with different probabilities:

$$X_{ij} = \begin{cases} Y_1 & p_1 \\ Y_2 & p_2 \\ \vdots & \\ Y_l & p_l \end{cases}$$

where Y_i is a random variable with density function $f_{Y_i}(\cdot)$ for $i = 1, \dots, l$, and the set of probabilities $\{p_1, \dots, p_l\}$ verifies that $\sum_{i=1}^{l} = 1$. For example, if $l = 2$, $p_1 = 0.3$, $p_2 = 0.7$, $Y_1 \sim U(1,2)$ and $Y_2 \sim N(0,1)$, the value of each cost will be that of a $U(1,2)$ with probability 0.3 and that of a $N(0,1)$ with probability 0.7.

The advantage of using mixtures lies in the fact that enables the modelling of situations where costs do not necessarily behave in the same way. Thus, given a cost matrix, its values might have been generated by random variables with different means, variances, or even different distributions. It is worth pointing out that, although this new approach gains in generality, the cumulative and density functions of the variables $\{M_k\}_{k\geq 1}$, now depending on l different distributions, become more complex. The procedure previously showed in sections 5.3.1 and 5.3.2 to calculate the expected value of the global cost for the BS-WR and BS algorithms is generalized next.

For the BS-WR algorithm, the cumulative distribution and density functions of M_k are

$$F_{M_k}(x) = 1 - \left[1 - \sum_{i=1}^{l} p_i F_{Y_i}(x)\right]^{r-k+1}, \tag{5.13}$$

$$f_{M_k}(x) = (r-k+1)\sum_{i=1}^{l} p_i f_{Y_i}(x)\left[1 - \sum_{i=1}^{l} p_i F_{Y_i}(x)\right]^{r-k}. \tag{5.14}$$

Applying these expressions, the expected value of the global cost (5.7) becomes

$$\sum_{k=1}^{Q} \int_{-\infty}^{\infty} x f_{M_k}(x)\, dx.$$

with $f_{M_k}(\cdot)$ given in (5.14).

In the case of the BS algorithm, the expected value of the global cost follows (5.11), but the new μ_1 (5.9) is given by

$$\mu_1 = \int_{-\infty}^{\infty} x\ rt \sum_{i=1}^{l} p_i f_{Y_i}(x)\left[1 - \sum_{i=1}^{l} p_i F_{Y_i}(x)\right]^{rt-1} dx. \tag{5.15}$$

The calculation of $h_{M_k}(M_{k-1})$ will be explained in Section 5.4.3, where this methodology is applied to a specific scenario.

5.4 Application of the analysis to different scenarios

The following section focused on an exploration application in different types of scenarios. It is considered that tasks are waypoint tasks and the costs are defined as the euclidean distance in 2D between the robot and the point to which the robot is

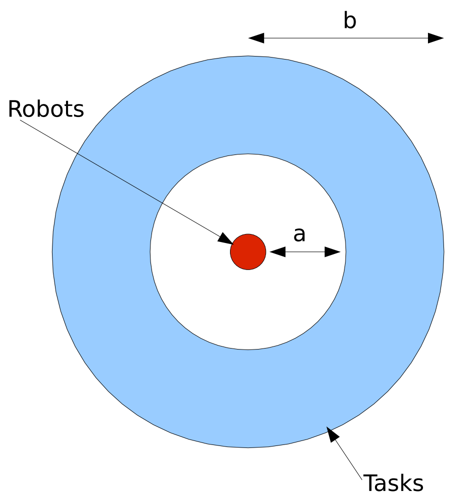


Figure 5.10: Dispersion scenario with costs uniformly distributed between $[a, b]$. Robots are within the red circle and the tasks are distributed at random within the blue doughnut.

expected to navigate. The results obtained in the previous section are completely general and can be used with other definitions of costs such as the consumed energy or the euclidean distance considered in three dimensions.

5.4.1 Dispersion scenario

The dispersion scenario describes a situation when a team of robots are deployed together, and afterwards have to be dispersed around an area. For example, imagine that a team of robots is sent to Mars, and after the landing, they have to get dispersed and explore the area. In this scenario, costs follow a uniform distribution between $[a, b]$. In this case the positions of the new formation are at least a distance a from the original position of the robots (see Figure 5.10).

Expected value for the BS-WR algorithm

Supposing that costs are uniformly distributed between $[a, b]$, i.e., $X_{i,j} \sim U(a, b)$. Then,

$$F_X(x) = \frac{x-a}{b-a}, \quad \text{for } a \leq x < b.$$

Applying (5.7), the expected value of the global cost is

$$E_{GC}(Q) = a \cdot Q + \sum_{k=1}^{Q} \frac{b-a}{(r-k+1)+1}. \tag{5.16}$$

Applying the change of variable $k' = r - k + 1$, the second term of the equation can be expressed as

$$\begin{aligned}\sum_{k'=r-Q+1}^{r} \frac{b-a}{k'+1} = (b-a) \cdot (H_{r+1} - H_{r-Q+1}) \quad &\simeq \\ (b-a) \cdot \left[\ln \frac{r+1}{r-Q+1} + A - B\right] & \end{aligned} \tag{5.17}$$

where $H_a = \sum_{k=1}^{a} 1/k$ is the harmonic number of the first a natural numbers and

$$A = \frac{1}{2 \cdot (r+1)},$$
$$B = \frac{1}{2 \cdot (r-Q+1)}.$$

Since $Q = \min\{r, t\} > 0$, A has a higher value than B and $\ln(r+1) > \ln(r-Q+1)$. When r increases, A tends to zero, and therefore, B also tends to zero. Then, (5.17) increases with the logarithm of the number of robots. Therefore, it can be seen that the term $a \cdot Q$ is dominant in the expected value of the global cost. This means that when the number of robots and tasks increases, $E_{GC}(Q)$ increases linearly with Q. Furthermore, the larger the number a, the bigger the slope will be of the linear dependency. However, when $a = 0$, $E_{GC}(Q)$ does not increase in a linear way but logarithmic with the number of robots when $r \leq t$.

Expected value for the BS algorithm

It is assumed that costs are uniformly distributed between $[a, b]$, i.e., $X_i \sim U(a, b)$. Then, using (5.8), it is obtained

$$F_{M_1}(x) = 1 - \left[1 - \frac{x-a}{b-a}\right]^{rt(0)}, \quad \text{for } a \le x < b$$

where $rt(a) = (r - a) \cdot (t - a)$.

First, μ_1 is computed using (5.9), obtaining

$$\mu_1 = \frac{rt(0) \cdot a + b}{rt(0) + 1}.$$

Next, the cumulative distribution function is needed to calculate $E(M_2|M_1)$ which has the following expression

$$P(M_2 < x|m_1) = 1 - \left[\prod_{i=1}^{rt(1)} P(X_i > x|m_1)\right]$$

where $P(X_i > x|m_1) = 1 - F_{X_i|M_1}(x)$. It can be easily seen that the random variable

$$X_i|M_1 \sim U(M_1, b), \quad \text{for } i = 1, \ldots, rt$$

and thus,

$$F_{M_2|M_1}(x) = 1 - \left[1 - \frac{x - m_1}{b - m_1}\right]^{rt(1)} \\ \text{for } m_1 \le x < b. \tag{5.18}$$

Therefore, the density function of the random variable M_2 is

$$f_{M_2|M_1}(x) = \frac{rt(1)}{b - m_1} \cdot \left(\frac{b - x}{b - m_1}\right)^{rt(1)-1} \quad \text{for } m_1 \le x < b. \tag{5.19}$$

By (5.18) and the computed density function (5.19), it follows that

$$h_{M_2}(M_1) = E(M_2|M_1) = \frac{rt(1)}{rt(1)+1}M_1 + \frac{b}{rt(1)+1}.$$

Using (5.10), the expected value of M_2 can be calculated as

$$\mu_2 = E(M_2) = \frac{rt(1)}{rt(1)+1}\mu_1 + \frac{b}{rt(1)+1}.$$

In order to obtain the successive values μ_k, it is proceeded similarly

$$h_{M_k}(M_{k-1}) = E(M_k|M_{k-1}) = \frac{rt(k-1)}{rt(k-1)+1}M_{k-1} + \frac{b}{rt(k-1)+1}.$$

Since $h_{M_k}(\cdot)$ is linear, the expected value of the global cost can be calculated using (5.12). Thus,

$$E_{GC}(Q) = \frac{rt(0)a+b}{rt(0)+1} + \sum_{k=2}^{Q} \frac{rt(0)a+b}{rt(0)+1} \prod_{i=2}^{k} \frac{rt(i-1)}{rt(i-1)+1} + \sum_{i=2}^{k-1} \frac{b}{rt(i-1)+1} \cdot \prod_{j=i+1}^{k} \frac{rt(j-1)}{rt(j-1)+1} + \frac{b}{rt(k-1)+1}. \quad (5.20)$$

5.4.2 Random formation scenario

In this scenario, robots and tasks are initially positioned randomly in a square area. This is usually the scenario used to test task allocation algorithms related to exploration or navigation. However in this scenario costs do not follow a known distribution and, as far as I know, this is the first effort to model the distribution of the costs in this useful scenario.

Distribution of the euclidean distance for uniformly distributed points

Supposing the initial positions of robots and tasks are calculated at random, the distribution of the distances between each robot and task is required for analysis.

This distribution cannot be modeled with a standard random variable, from my best knowledge, there is no previous work that has calculated the expression of this random variable.

The position of the robots (X_r, Y_r) are random variables that follow a uniform distribution. For simplification and without losing generality, let X_r and Y_r be in the range of $[0, 1]$. The position of the tasks (X_t, Y_t) also follow uniform random variables between $[0, 1]$. Moreover, it is assumed that all the variables are independent. The distribution of $X \sim |X_r - X_t|$ is defined by the following cumulative distribution function

$$F_X(x) = 2x - x^2 \quad \text{if } 0 \leq x \leq 1. \tag{5.21}$$

Next, the distribution of the cost is defined as

$$C \equiv \sqrt{X^2 + Y^2}$$

where X and Y are random variables defined by (5.21) and C is the random variable of the costs defined as the euclidean distance.

The cumulative distribution function of the new random variable, C, can be calculated as follows

$$F_C(c) = \int_{-\infty}^{\infty} F_X(\sqrt{c^2 - y^2}) f_Y(y) dy$$

which has to be defined by parts

$$F_C(c) = \begin{cases} 0 & \text{if } c \leq 0 \\ \int_0^c F_X(\sqrt{c^2 - y^2}) f_Y(y) dy & \text{if } 0 \leq c \leq 1 \\ \int_0^{\sqrt{c^2-1}} F_X(\sqrt{c^2 - y^2}) f_Y(y) dy + & \\ \int_{\sqrt{c^2-1}}^{1} F_X(\sqrt{c^2 - y^2}) f_Y(y) dy & \text{if } 1 \leq c \leq \sqrt{2} \\ 1 & \text{if } c \geq \sqrt{2} \end{cases}$$

Solving these integrals, it is obtained

$$F_C(c) = \begin{cases} 0 & \text{if } c \leq 0 \\ \frac{1}{2}c^4 - \frac{8}{3}c^3 + \pi c^2 & \text{if } 0 \leq c \leq 1 \\ 2c^2 \arcsin\frac{1}{c} - 2c^2 \arcsin\frac{\sqrt{c^2-1}}{c} + & \\ 2c^2\sqrt{c^2-1} - c^4 + \frac{2}{3}(c^2-1)^{\frac{3}{2}} + & \\ \frac{(c^2-1)^2}{2} - \frac{7}{6} + 2\sqrt{c^2-1} - (c^2-1) & \text{if } 1 \leq c \leq \sqrt{2} \\ 1 & \text{if } c \geq \sqrt{2} \end{cases} \tag{5.22}$$

Once the cumulative distribution function is obtained, the random variable that models the cost as the euclidean distance is totally defined and the probability density function can be calculated easily. However, it can be observed that the density function is not continuous for $c = 1$ and this will imply changes in the way to calculate the expected value.

Expected value for the BS-WR algorithm

Since the cost density function (5.22) is not continuous, an alternative way of calculating the expected value is used. By definition

$$E(Y) = \int y \cdot f_Y(y)\, dy.$$

Using the integration by parts rule

$$E(Y) = [y \cdot F_Y(y)] - \int F_Y(y)\, dy. \tag{5.23}$$

Therefore, applying the above rule in (5.7) and with $F_Y(y)$ as $F_{M_k}(x)$ defined in (5.3), the expected value of the global cost is

$$E_{GC}(Q) = \sum_{k=1}^{Q} A - \int_0^{\sqrt{2}} 1 - [1 - F_X(x)]^{r-k+1}\, dx \tag{5.24}$$

where $F_X(x)$ is the cumulative distribution function defined in (5.22) and

$$A = \left[x \cdot \left(1 - [1 - F_X(x)]^{r-k+1}\right)\right]_0^{\sqrt{2}}$$

which is a constant value. As it will be explained in Section 5.5, this integral cannot be solved analytically and numerical methods have to be used.

Expected value for the BS algorithm

The steps explained in Section 5.4.1 are followed. First, μ_1 is calculated using the rule of integration by parts defined in (5.23) with (5.9), where $F_Y(y)$ is $F_{M_1}(x)$ defined in (5.8). Then,

$$\mu_1 = B - \int_0^{\sqrt{2}} 1 - [1 - F_X(x)]^{rt(0)} \, dx \tag{5.25}$$

where $F_X(x)$ is defined in (5.22), $rt(a) = (r-a) \cdot (t-a)$, and B is a constant value equal to

$$B = \left[x \cdot \left(1 - [1 - F_X(x)]^{rt(0)}\right)\right]_0^{\sqrt{2}}.$$

Next, the variable $h_{M_k}(M_{k-1}) = E(M_k|M_{k-1})$ is calculated as

$$h_{M_k}(M_{k-1}) = C - \int_{m_{k-1}}^{\sqrt{2}} 1 - \left[\frac{1 - F_X(x)}{1 - F_X(m_{k-1})}\right]^{rt(k-1)} dx \tag{5.26}$$

where

$$C = \left[x \cdot \left(1 - \left[\frac{1 - F_X(x)}{1 - F_X(m_{k-1})}\right]^{rt(k-1)}\right)\right]_{m_{k-1}}^{\sqrt{2}}.$$

From (5.25) and (5.26), the expected value of the global cost (E_{GC}) is computed using (5.11).

5.4.3 Extension using mixtures

In this section, it is illustrated with an example how to generalize the probabilistic analysis using mixtures. Let imagine a dispersion application (see Figure 5.10), where two doughnuts are considered. The first one with radius $[a, b]$ is inside the other one with radius $[c, d]$, where $c \geq b$. Imagine that tasks are generated by a human operator who chooses with equal probability the inside or the outside doughnut. Therefore, costs can follow a uniform-$[a, b]$ distribution or a uniform-$[c, d]$ distribution with equal probability. And they are modeled with the following mixture of distributions

$$X_{i,j} \sim \begin{cases} U(a,b) & p_1 = 0.5 \\ U(c,d) & p_2 = 0.5 \end{cases}$$

The probabilistic analysis (see Section 5.3) is used to calculate the expected value of the global cost for the BS-WR algorithm. First, (5.13) is applied to calculate

$$F_{M_k}(x) = 1 - [1 - g(x)]^{(r+k-1)}, \tag{5.27}$$

where

$$g(x) = \begin{cases} 0 & x \leq a \\ \dfrac{x-a}{2(b-a)} & a < x < b \\ \dfrac{1}{2} & b < x \leq c \\ \dfrac{1}{2} + \dfrac{x-c}{2(d-c)} & c < x \leq d \\ 1 & x \geq d \end{cases} \tag{5.28}$$

$E(M_k)$ is expressed as an integral by parts using (5.5), where the cumulative distribution function is expressed by (5.27).

$$E(M_k) = \int_a^b x(r-k+1)\left[1 - \frac{x-a}{2(b-a)}\right]^{r-k} \frac{1}{2(b-a)}\,dx \quad + \\ \int_c^d x(r-k+1)\left[1 - \frac{x-c}{2(d-c)}\right]^{r-k} \frac{1}{2(d-c)}\,dx.$$

Once these integrals are computed,

$$E(M_k) = \frac{(r-k+1)a+2b-a}{r-k+2} - \frac{(r-k+1)b+2b-a}{r-k+2}\left(\frac{1}{2}\right)^{r-k+1} + \frac{(r-k+1)c+d}{r-k+2}\left(\frac{1}{2}\right)^{r-k+1}. \tag{5.29}$$

From (5.29), $E_{GC}(Q)$ can be easily calculated using (5.6).

On the other hand, the procedure for the BS algorithm starts calculating μ_1 from (5.9) where $F_{M_1}(x)$ is defined as

$$F_{M_1}(x) = 1 - [1-g(x)]^{rt},$$

and $g(x)$ is defined as in (5.28). Thus, μ_1 has the following expression

$$\mu_1 = \frac{rta+2b-a}{rt+1} - \frac{rtb+2b-a}{rt+1}\left(\frac{1}{2}\right)^{rt} + \frac{rtc+d}{rt+1}\left(\frac{1}{2}\right)^{rt}.$$

Next, in order to calculate $E(M_2|M_1)$, its cumulative distribution is needed:

$$F_{M_2|M_1}(x) = 1 - [1-F_{X_i|M_1}(x)]^{rt(0)} \tag{5.30}$$

It is easily seen that the value of the first minimum, m_1, will characterize the distribution of the remaining costs. There exist two possibilities for m_1: either it lies in $[a,b]$ (with probability $F_{M_1}(b)$) or in $[c,d]$ (with probability $1-F_{M_1}(c)$). In the first case, the remaining costs will follow a mixture composed of a uniform-$[m_1,b]$ distribution with probability 0.5, or a uniform-$[c,d]$ distribution with probability 0.5. If however, m_1 lies in $[c,d]$, the remaining costs must lie in $[m_1,d]$ and thus, they follow a uniform-$[m_1,d]$ distribution. The condition cumulative function of the cost given the first minimum is computed applying the total probability theorem as

$$F_{X_i|M_1}(x) = F_{M_1}(b) \begin{cases} 0 & x \leq m_1 \\ \dfrac{x-m_1}{2(b-m_1)} & m_1 < x < b \\ \dfrac{1}{2} & b < x \leq c \\ \dfrac{1}{2} + \dfrac{x-c}{2(d-c)} & c < x \leq d \\ 1 & x \geq d \end{cases} + [1 - F_{M_1}(c)] \begin{cases} 0 & x \leq c \\ \frac{x-c}{d-c} & c < x < d \\ 1 & x \geq d \end{cases} \tag{5.31}$$

where $F_{M_1}(b) = F_{M_1}(c) = 1 - (0.5)^{rt(0)}$. Once the expected value $h_{M_2}(M_1) = E(M_2|M_1)$ is computed from (5.30) and (5.31), it is easy to check (analogously to the previous sections) that it is linear with respect to M_1. Therefore, it can be said that h_{M_k} is a linear function of M_k. Finally, $E_{GC}(Q)$ can be calculated using (5.12).

5.5 Simulation results

Although the probabilistic analysis has been developed without restrictions between the number of robots and tasks, the first simulations are run with $r = t$. For the dispersion scenario, the expected value of the global cost (E_{GC}) for the BS-WR algorithm can be calculated using (5.16) and for the BS algorithm using (5.20) considering the case $r = t = Q$. In Figure 5.11, the results when costs follow a uniform distribution between $[5, 100]$ are shown, and can be observed that the theoretical E_{GC} remains very close to the values obtained in simulation. Also, BS algorithm obtains better results than BS-WR algorithm proving that reallocation of tasks reduces the global cost. As was commented in Section 5.4.1, when the minimum value of the costs (a) is not zero, E_{GC} increases linearly with the number of robots since all the robots, no matter which task they choose, have to navigate at least a distance equal to a. The slope of this linear dependency is proportional to the value of a as can be seen in Figure 5.12.

A special case appears when $a = 0$. As can be observed from Figure 5.13, E_{GC} for both algorithms increases with the logarithm of the number of robots and tasks,

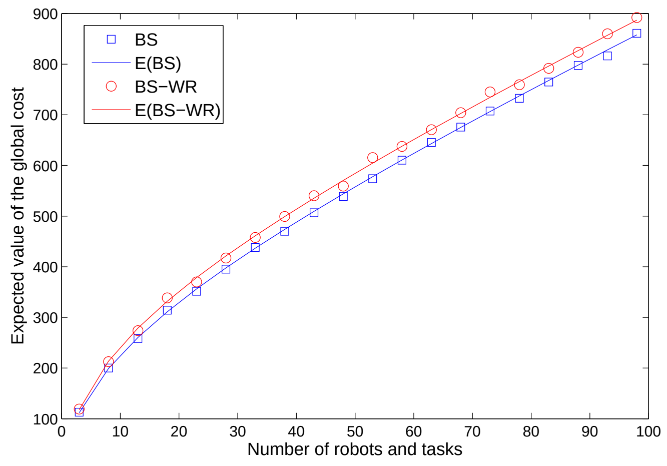


Figure 5.11: Expected value of the global cost over 100 simulations where costs follow a uniform distribution between $[5, 100]$. The circles represent the results from simulation for the BS-WR algorithm and the squares for the BS algorithm. The theoretical results, E(BS) and E(BS-WR) respectively, are shown as solid lines.

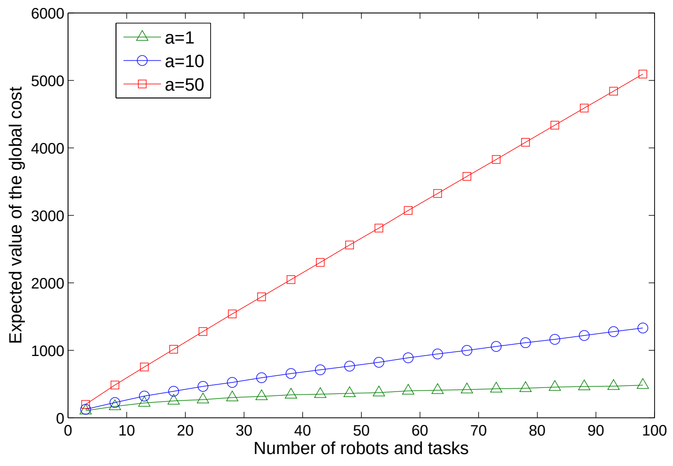


Figure 5.12: Expected value of the global cost over 100 simulations where costs follow a uniform distribution between $[a, 100]$, being a equal to 1, 10 and 50. The slope of E_{GC} is directly proportional to a.

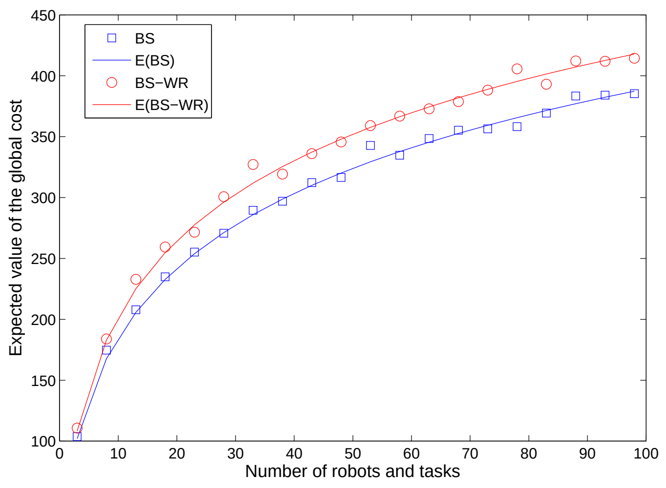


Figure 5.13: Expected value of the global cost over 100 simulations where costs follow a uniform distribution between $[0, 100]$. In this case E_{GC} for both algorithms increases with the logarithm of the number of robots and tasks.

and the theoretical results also show this behavior. This case was explained for the BS-WR algorithm in Section 5.4.1.

Next, the theoretical results are applied to situations when the number of robots and tasks are not the same. When the number of robots is larger than the number of tasks, tasks are allocated to the robots with the lowest costs and the rest of the robots will be idle, waiting for more tasks. In the simulations, it has been considered half as many tasks as robots. Therefore, (5.16) and (5.20) were used with $t = \lfloor r/2 \rfloor$ and $Q = t$. Figure 5.14 shows that the results from the probabilistic analysis still remain close to the simulation results. The expected value of the global cost is smaller than in Figure 5.13 since only the robots with the lowest costs win a task. For the same reason, the difference between BS-WR and BS algorithms decreases. The zigzag pattern of the expected value in Figure 5.14 comes from the fact that t is an integer, and when r is an odd number, t is not exactly half of r. Another interesting fact is that the expected value seems to remain constant when the number of robots is large enough instead of increasing with the logarithm of r as happened when the number of robots and tasks are equal. From (5.16) and (5.17), when $r = J \cdot t$ for $J > 1$, it

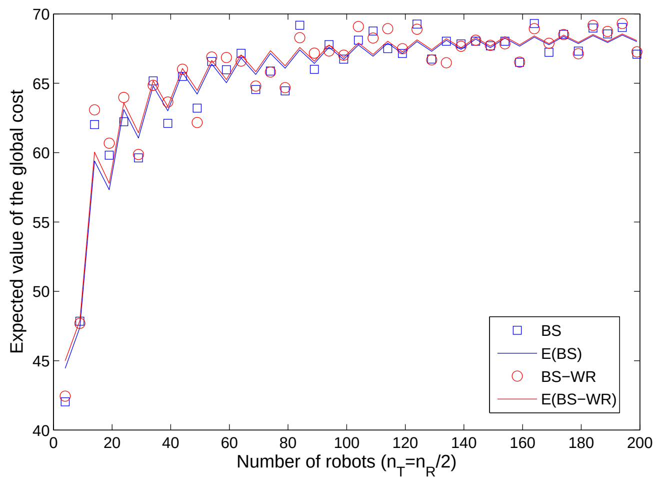


Figure 5.14: Expected value of the global cost over 100 simulations where the costs follow a uniform distribution between $[0, 100]$. The number of tasks is half the number of robots.

can be inferred that

$$E_{GC}(Q) = a{\cdot}t+(b-a)\left(H_{J\cdot t+1} - H_{(J-1)t+1}\right) \underset{t\to\infty}{=} a{\cdot}t+(b-a)\ln\left(\frac{J\cdot t+1}{(J-1)t+1}\right) \quad (5.32)$$

where H_a is the harmonic number of the first a natural numbers. Applying the values used in Figure 5.14 ($a = 0$, $b = 100$ and $J = 2$) to Formula (5.32), E_{GC} tends to 69.26 which matches with the simulated results. It can be seen here the potential of having an analytical formula of the performance of the algorithms. Other interesting parameters, such as the relation between J and how fast the global cost tends to the constant value could be calculated easily.

When the number of robots is smaller than the number of tasks, only the tasks with lowest costs will be allocated to robots. The rest of the tasks will not be allocated to any robot. This way of operation has been decided to simplify the analysis and make easier the comparison between algorithms. (5.16) and (5.20) are again used for the BS-WR and the BS algorithms respectively, with twice as many tasks as robots ($t = 2 \cdot r$ and $Q = r$). Then, Figure 5.15 shows that the BS-WR algorithm performs much worse than the BS algorithm. Since the BS-WR algorithm does not use reallocation,

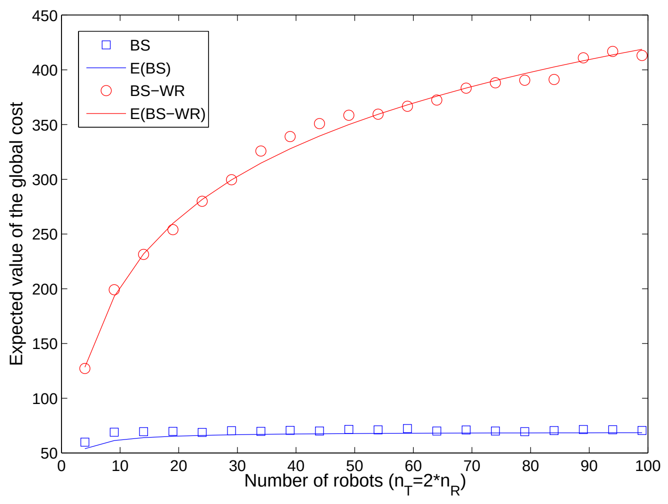


Figure 5.15: Expected value of the global cost over 100 simulations where the costs follow a uniform distribution between $[0, 100]$. The number of tasks is twice the number of robots.

only the first r tasks will be considered and the problem is equivalent to the one with r number of robots and tasks. However, using reallocations, BS algorithm takes into account all the tasks but only allocates the best r tasks to robots. That is the reason why E_{GC} for BS-WR algorithm increases with the logarithm of the number of robots and E_{GC} tends to a constant value as in the previous case.

On the other hand, the algorithms have been simulated for the random scenario. The expected value of the global cost for the BS-WR algorithm is calculated using (5.24). Since it is not possible to calculate an analytical solution for this formula, a numerical method has been used to solve the integral. The trapezoid rule was used since $F_X(x)$, shown in (5.22), is not twice continuously differentiable and other methods, such as the Simpson's rule, have problems in finding an accurate solution for this case. For the BS algorithm, the formulae commented in Section 5.4.2 have been used in combination with the trapezoid rule. In Figure 5.16, it is observed that the theoretical results match the simulation ones. The BS algorithm still obtains better results than the BS-WR algorithm.

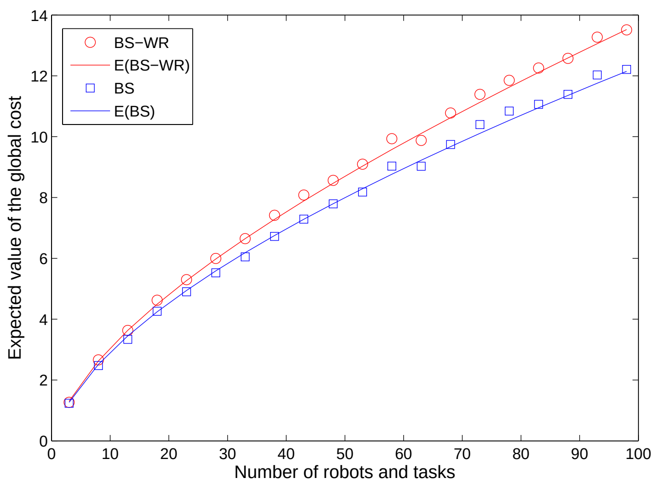


Figure 5.16: Expected value of the global cost over 100 simulations where the costs follow the distribution explained in Section 5.4.2.

Finally, the theoretical results have been validated for the extension scenario. The mixture of distributions was defined by $a = 0, b = 1, c = 2$ and $d = 3$ (see Section 5.4.3). Figure 5.17 shows that the BS algorithm again obtains better results than the BS-WR algorithm. It also shows that the theoretical results model correctly the algorithm behavior in this type of scenario.

It is important to point out that since the expected value or mean of the global cost is used, the presented results are close to the real costs when the number of experiments is large enough. As can be seen from Figure 5.18, the difference between E_{GC} calculated theoretically and the one obtained from the simulations decreases with the number of experiments. It could be interesting to obtain a formula that indicates how large the difference between the theoretical and experimental results could be, depending on the number of experiments.

For this purpose, it is needed to model the sample mean of the global cost. Using the Central Limit Theorem (see Theorem 1), if the number of samples (in this case experiments) is sufficiently large, the sample mean follows a normal distribution and it can be bounded up to a percentage. Usually, it is considered that a sufficiently large number of samples is above 30. So, if more than 30 experiments are run, it can be said

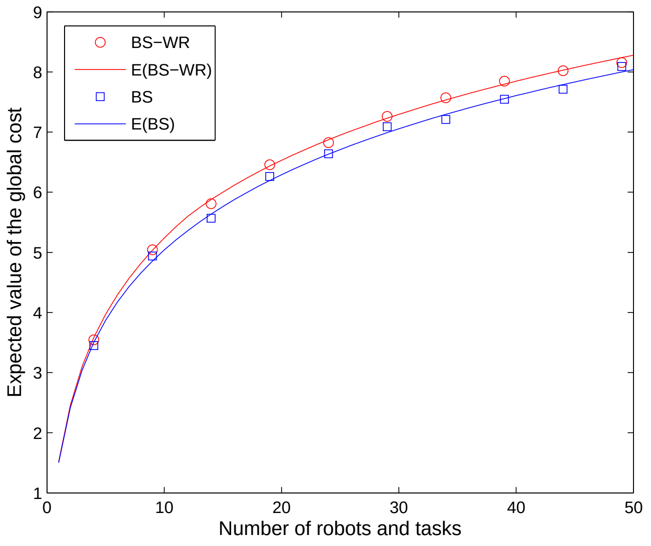


Figure 5.17: Expected value of the global cost over 100 simulations where costs follow a mixture of distributions (uniform-[0, 1] and uniform-[2, 3]).

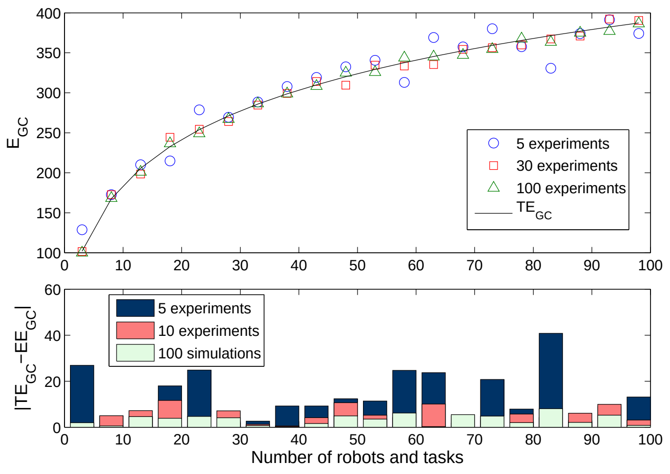


Figure 5.18: Expected value of the global cost for different number of missions. At the top, it can be seen the different expected values obtained from different number of missions. The solid line represents the theoretical value. At the bottom, the bars represents the difference between the expected value of the global cost from the simulations (sample mean) and from the theoretical result (population mean).

that $(1-\alpha)\cdot 100\%$ of the sample means will lie between $t_{\alpha/2}/\sqrt{N}$ sample standard deviations of the population mean (N is the number of experiments). Thus, for α values small enough, it can be obtained an approximation of the maximal difference between E_{GC} calculated from the probabilistic analysis (TE_{GC}) and the one obtained from the experimental results (EE_{GC}) as

$$max(|TE_{GC} - EE_{GC}|) \gtrsim t_{\alpha/2}\frac{s}{\sqrt{N}} \tag{5.33}$$

where $t_{\alpha/2}$ is the t-value with an area $\alpha/2$ to its right (usually obtained from a table), and s is the sample standard deviation. It is considered that $\alpha = 0.1$ is small enough since it means that 90% of the differences between TE_{GC} and EE_{GC} will be smaller than the right part of Formula (5.33). For $\alpha = 0.1$ and $N \geq 30$, it is obtained

$$max(|TE_{GC} - EE_{GC}|) \gtrsim 1.282\frac{s}{\sqrt{N}}.$$

Theorem 1 (Central Limit Theorem) *Given a random variable X with mean μ and variance σ^2, the sampling distribution of the sample mean ($\overline{x}$) follows a normal distribution with mean $\mu_{\overline{x}} = \mu$ and variance equal to $\sigma^2_{\overline{x}} = \frac{\sigma^2}{N}$ for a sufficiently large number of samples (N).*

5.6 Experiments with robots

Thirty-six experiments have been run using the presented robotic testbed (see Appendix B). The main objective of these experiments is to show that the theoretical results are still valid even when real robots are used, with all the noise and imperfections. The BS algorithm has been implemented since it is the most complete of the two, and it has been tested in two different scenarios (dispersion and random).

First, the experiments emulating a dispersion scenario have been run. In this scenario robots were uniformly distributed in a circle of radius one meter and tasks in a doughnut with 6 and 7 meters as the inner and exterior radius respectively. Therefore, costs are uniformly distributed between $[5, 7]$ meters. A method based

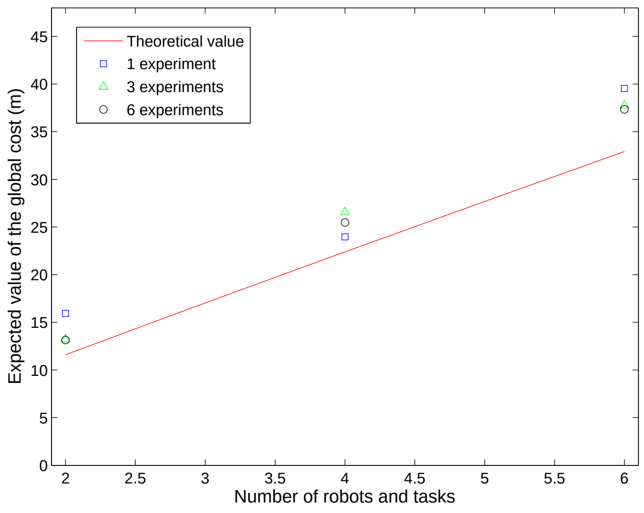


Figure 5.19: Results from the experiments for the dispersion scenario. Costs were uniformly distributed between $[5, 7]$ meters. The mean from different number of experiments have been calculated for 2, 4 and 6 robots.

on (Calafiore et al., 1998) was used to obtain points uniformly distributed on l_p doughnuts instead of l_p balls. As can be observed from Figure 5.19, the sample mean calculated from the experiments gets closer to the theoretical value when the number of experiments increases.

The results from the experiments for the random scenario are shown in Figure 5.20. In this scenario, robots and tasks have been calculated at random in the 15x23m^2 arena. It can be observed how the sample mean gets closer to the theoretical value when the number of experiments increases.

On the other hand, there are cases (4 robots in both scenarios) that the mean from the first experiment gets results closer to the theoretical value than with a higher number of experiments. This can be explained due to the random nature of the costs. The same can happen when the mean is calculated using the samples obtained from a random generator function.

In summary, it has been demonstrated that the theoretical results are still valid when real robots are used and it has been shown that these results accurately model the behavior of the algorithms. Therefore, the theoretical results presented in this

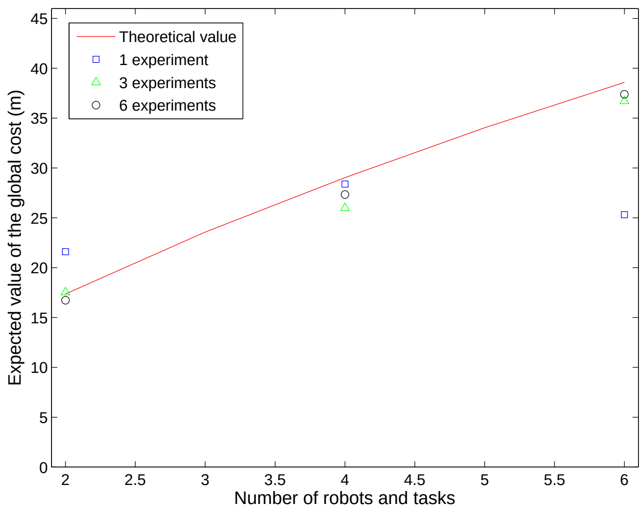


Figure 5.20: Results from the experiments for the random scenario. The position of the robots and locations of the tasks were distributed as: X coordinate follows a uniform distribution between $[0, 15]$ meters and the Y coordinate between $[0, 23]$ meters. The mean from different number of experiments have been calculated for 2, 4 and 6 robots.

chapter can be used to compare different algorithms in different situations and predict their behavior.

5.7 Performance analysis

In this section, it is aimed to compare the performance of the proposed algorithms with respect to the optimal allocation. However, a closed-form theoretical expression cannot always be found for the optimal allocation due to the complexity of the problem, that depends on the underlying distributions. Several works have undertaken the job assignment problem, with the goal of obtaining the expected value of the global cost for the optimal allocation (see for example (Mézard and Parisi, 1987) and (Walkup, 1979)). As has been done in this work, a matrix formed by independent and equally distributed random variables (costs), is used to model the assignment problem. To reduce the mathematical complexity, only costs following either exponential or uniform-$[0, 1]$ distributions have been considered. In the case of a uniform-$[0, 1]$ distribution, it has been proved in (Aldous, 2001) that the expected value of the optimal global cost converges to a constant value, $\pi^2/6$ (see Figure 5.21), when the number of robots and tasks increases. Up to my knowledge, no work has been devoted to the calculation of the optimal global cost when the distribution is given by (5.22). Thus, it is only possible to compare the performance of the market-based algorithms with respect to the optimal allocation for the dispersion scenario when the number of robots and tasks is large.

In Figure 5.21, it is shown that the performance of the BS-WR and BS algorithms gets worse when Q increases. From (5.16) and (5.20), it can be observed that the expected values increase towards infinity for large number of robots and tasks while the optimal expected value tends to a constant value.

However, the optimal expected value does not tend to a constant value when the cost distribution is a uniform-$[a, b]$ with $a > 0$. As was said before, there is no theoretical result that calculates this optimal expected value for a distribution different from a uniform-$[0, 1]$. Therefore, simulations have been run to calculate the optimal expected value using the Hungarian method (Kuhn, 1955). However,

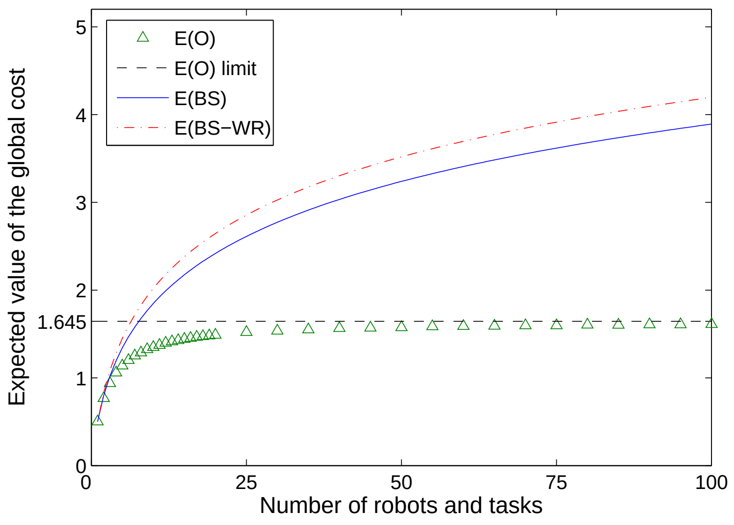


Figure 5.21: Expected value of the global cost for the BS-WR and BS algorithms in comparison with the optimal expected values, $E(O)$. Costs follow a uniform distribution between $[0, 1]$. The optimal expected values have been calculated from 1000 simulations per case. The BS-WR and BS expected values are calculated from the theoretical results.

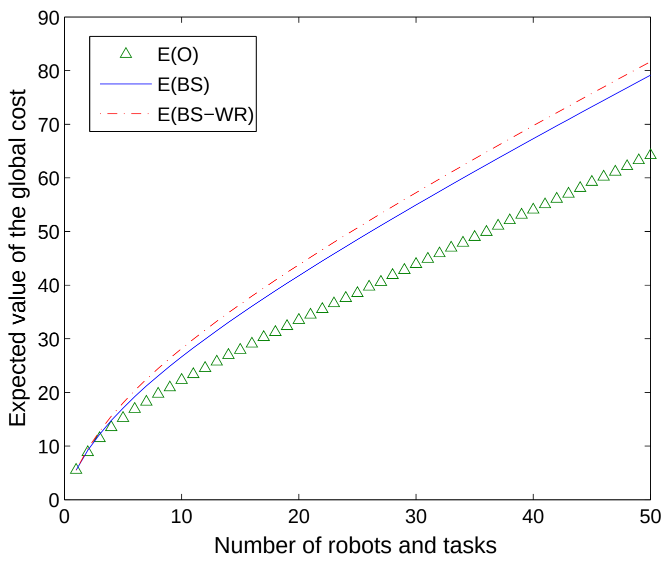


Figure 5.22: Expected value of the global cost for the BS-WR and BS algorithms in comparison with the optimal expected values, $E(O)$. Costs follow a uniform distribution between $[1, 10]$. The optimal expected values have been calculated from 1000 simulations per case. The BS-WR and BS expected values are calculated from the theoretical results.

the BS-WR and BS expected values were computed from the theoretical results. In Figure 5.22, it can be seen that the optimal allocation does not converge to a constant value. As was explained in Section 5.4.1, the expected value increases linearly with Q when $a > 0$. However, it can be observed that the slope associated with the optimal allocation is smaller than the ones associated with the market-based algorithms. Therefore, the performance of the market-based algorithms decreases with higher values of Q. The same conclusion can be drawn for the random scenario.

Finally, the theoretical results are used to compare the performance of the BS-WR and the BS algorithms. Figure 5.23 depicts the difference between the expected values for the BS-WR and the BS algorithms when the number of robots and tasks lies in $[1, 500] \times [1, 500]$ and the costs follow a uniform-$[0, 1]$ distribution. It can be observed how the difference is always zero or positive, implying that the BS-WR algorithm always obtains the same or higher expected value than the BS algorithm, and highlighting the benefits of the reallocation. Therefore, the BS algorithm performs

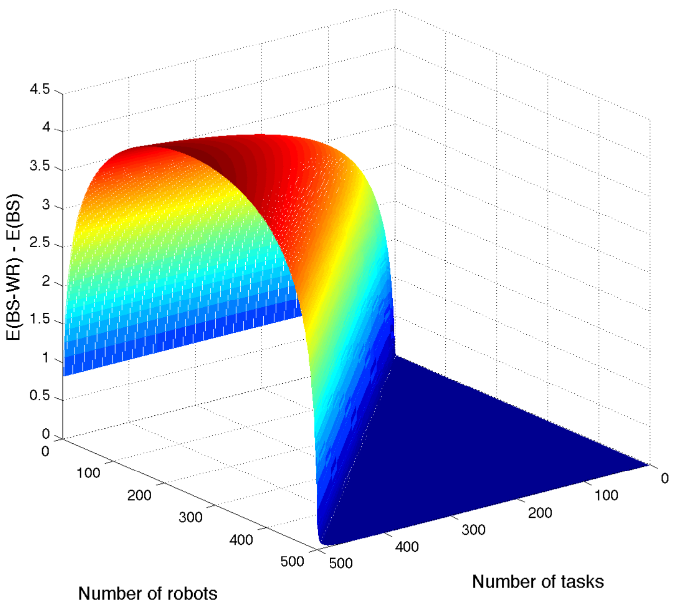


Figure 5.23: Difference between the expected value of the global cost for the BS-WR and the BS algorithms. It can be observed that the BS-WR algorithm always obtains the same or higher expected value than the BS algorithm.

better than the BS-WR algorithm for less than 500 robots and tasks which arguably covers any type of application. It is important to point out that the maximum difference between both algorithms is given when the number of robots is around half the number of tasks. However, both algorithms obtained similar results when the number of tasks is smaller than the number of robots. This comparison shows the potential of the theoretical results; in order to achieve similar results with simulations, more than 5000 million simulations would have been run. Similar results have been obtained with the rest of scenarios and distributions.

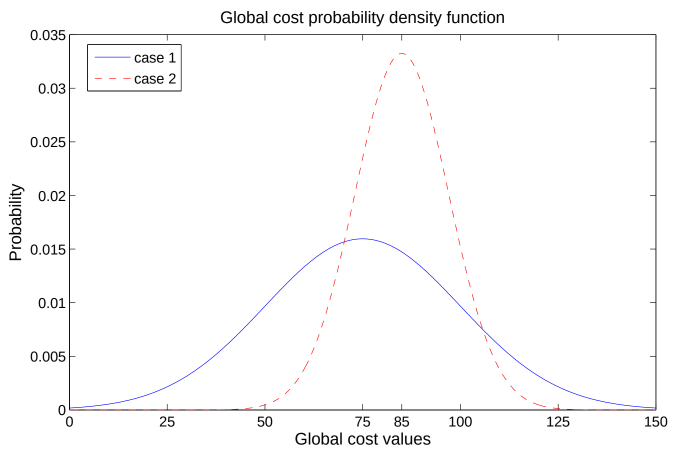


Figure 5.24: Probability distributions that model two different global costs. The solid line case has a lower mean but a higher variance than the other case.

5.8 Extension of the probabilistic analysis for the BS-WR algorithm

The presented probabilistic approach is continued and extended to calculate the variance of the global cost. The motivation for this work comes from the fact that even though the mean is a better metric than that worst-case metric, it is not enough to understand completely the behavior of a task allocation algorithm. For example, in Figure 5.24 it is shown the distribution of the global cost for two different cases. The solid line case has a lower mean but a higher variance. In this situation, it is not easy to decide which case is the best, i.e., which one has the highest probability of obtaining a lower global cost. In this section, it will be shown how the mean and the variance can be used to model the global cost by a normal distribution. This enables the computation of such probabilities, and thus, choosing between both situations. From my best knowledge, this is the first time that the solution (global cost) of a distributed task allocation algorithm is modeled, and an analytical methodology of comparison is explained.

The calculation of the standard deviation is a complex operation and only valid results have been obtained for the BS-WR algorithm. It is planned to use similar procedures to extend this calculation to the rest of the MRST algorithms.

5.8.1 Calculation of the standard deviation

As was shown in Section 5.2.1, the global costs for the BS-WR algorithm and the column-scan method are the same and it is defined as

$$E_{GC}(Q) = \sum_{k=1}^{Q} E(M_k)$$

where $E(M_k)$ is the expected value of each of the costs that come from the executed tasks (minimum elements of each column of the cost matrix) which is calculated using (5.5).

Since the executed tasks are independent (one task allocation does not depend on the already allocated tasks), the variance of the random variable representing the global cost can be calculated as the sum of the variances of the random variables M_k, i.e.,

$$V_{GC}(n) = \sum_{k=1}^{Q} V(M_k)$$

where $V(M_k)$ is defined as

$$V(M_k) = E(M_k^2) - E(M_k)^2. \tag{5.34}$$

From basic probabilistic theory,

$$E[g(Z)] = \int g(z) f_Z(z)\, dz.$$

Then, $E(M_k^2)$ can be easily calculated as

$$E(M_k^2) = \int_{-\infty}^{\infty} x^2 \cdot (r-k+1)\left[1 - F_X(x)\right]^{r-k} f_X(x)\, dx \tag{5.35}$$

where $f_{M_k}(x)$ is defined in (5.4).

Finally, the variance of the global cost is

$$V_{GC}(n) = \sum_{k=1}^{Q} \left[E(M_k^2) - E(M_k)^2\right]$$

where $E(M_k^2)$ is computed from (5.35) and $E(M_k)^2$ from (5.5).

5.8.2 Application to the dispersion scenario

The dispersion scenario has been chosen to illustrate the calculation of the variance. However, following similar steps, this approach can be applied to other scenarios, such as the random or extension scenarios.

From Section 5.4.1, it is known that the expected value of the global cost for this case is

$$E_{GC}(Q) = \sum_{k=1}^{Q} a + \frac{b-a}{(r-k+1)+1}.$$

Therefore,

$$E(M_k) = a + \frac{b-a}{(r-k+1)+1}.$$

Also, from (5.35)

$$\begin{aligned} E(M_k^2) = \int_a^b x^2 \cdot (r-k+1) \left[1 - \frac{x-a}{b-a}\right]^{r-k} \frac{1}{b-a}\, dx &= \\ \frac{r-k+1}{(b-a)^{r-k+1}} \int_a^b x^2 (b-x)^{r-k}\, dx. \end{aligned}$$

Solving the integral by parts where $u = x^2$ and $dv = (b-x)^{r-k}$, it is obtained

$$E(M_k^2) = a^2 + \frac{2\,(a\,(r-k+2)+b)\,(b-a)}{(r-k+2)\,(r-k+3)}.$$

By (5.34)

$$V(M_k) = \frac{(r-k+1)\,(a-b)^2}{(r-k+3)(r-k+2)^2},$$

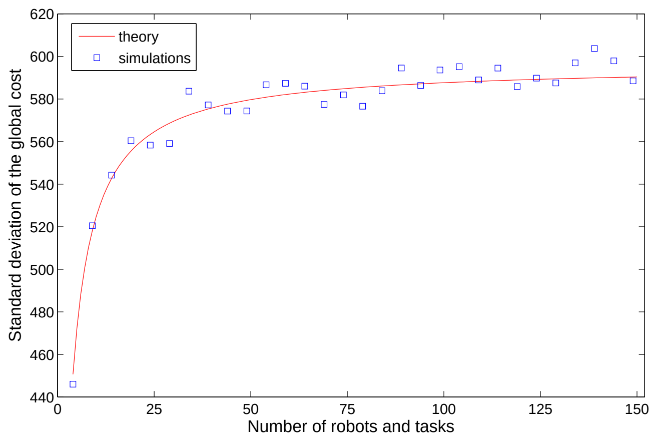


Figure 5.25: Standard deviation of the global cost calculated from the theoretical results and simulations. The squares represent the results from simulation applying the BS-WR algorithm over 1000 simulations per number of robots and tasks. The theoretical results, $\sqrt{V_{GC}(n)}$, are shown as a solid line. Both results where calculated in a dispersion scenario with $a = 0$ and $b = 1000$.

and thus, the variance of the global cost is

$$V_{GC}(n) = \sum_{k=1}^{Q} V(M_k) = \sum_{k=1}^{Q} \frac{(r-k+1)\,(a-b)^2}{(r-k+3)(r-k+2)^2} \tag{5.36}$$

where Q is the minimum between r (number of robots) and t (number of tasks), and a and b are the upper and lower bounds for the uniform distribution that models the costs.

Simulations have been run to validate this theoretical results. Figure 5.25 depicts both the theoretical and estimated (from simulations) standard deviation of the global cost. The standard deviation, defined as the square root of the variance, has been used since it makes the visualization of the results easier. The theoretical results have been calculated using (5.36). For each number of robots and tasks the estimated variance has been computed from 1000 simulations. It can be observed how the simulated and theoretical values are similar.

Finally, simulations have been run to evaluate the situation when the number of robots is higher than the number of tasks. In Figure 5.26, the theoretical standard

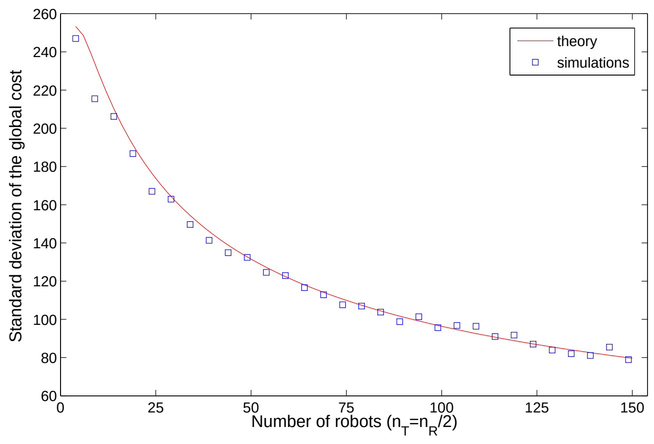


Figure 5.26: Standard deviation of the global cost when the number of tasks is half the number of robots ($n_R = n_T/2$). The squares represent the results from simulation applying the BS-WR algorithm over 1000 simulations. The theoretical results, $\sqrt{V_{GC}(n_T, n_r)}$, are shown as a solid line. Both results where calculated in a dispersion scenario with $a = 0$ and $b = 1000$.

deviation is computed using (5.36), with $Q = t$, $t = \lfloor r/2 \rfloor$, and the simulated standard deviations are calculated from 1000 runs per case (the number of tasks is half the number of robots). It can be observed how the theoretical standard deviation of the global cost almost coincides with the standard deviation obtained from the simulations.

5.8.3 Normal approximation of the global cost distribution

In the previous sections, it has been shown how to compute the mean (expected value) and variance of the global cost distribution, when the MRTA problem is solved using the BS-WR algorithm. Even though these parameters provide certain information about the distribution, they do not completely characterize it. Since the real CDF (cumulative distribution function) is hard to calculate analytically, the information provided by the mean and variance can be used to obtain an approximation by a normal distribution. The normal distribution is well-known and has shown to be suitable to model very different situations. In addition, the Kolmogorov-Smirnov test was used to validate this choice.

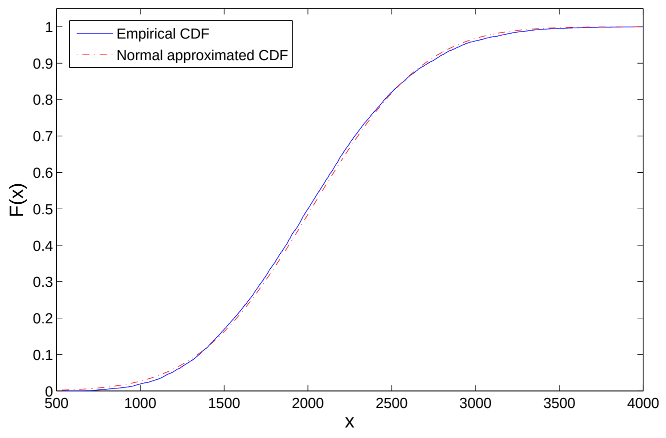


Figure 5.27: Comparison of the empirical CDF and the normal approximation whose parameters are computed using the probabilistic approach. The empirical CDF has been calculated from 10000 simulations using the BS-WR algorithm in a 1000mx1000m area with the costs uniformly distributed between 0 and 1000.

The procedure to be followed is the next one. First, the mean and variance of the global cost are calculated for a specific case (such as the distribution of the matrix costs, and the number of robots and tasks) using the presented probabilistic approach. Since the normal distribution is completely described by its mean and variance, the previously computed parameters are used as those of the normal distribution.

In order to test this methodology, the BS-WR algorithm has been simulated for 10 robots and tasks with costs uniformly distributed between 0 and 1000. In Figure 5.27, it can be observed how the CDF of the real distribution (empirically calculated from 10000 simulations) is very similar to the CDF of a normal distribution calculated from the mean (5.16) and variance (5.36) formulae that have been obtained using the probabilistic approach.

Once the global cost distribution has been approximated by a normal distribution, it is possible to compare analytically different cases and study which is the probability that one case obtains a lower global cost. For example, supposing that the BS-WR algorithm has been applied to two different scenarios obtaining the approximated distributions that are shown in Figure 5.24. It is very difficult to decide which is the best situation, since one distribution has a lower mean but also a higher variance.

However, this can be decided theoretically calculating which distribution has the highest probability to obtain the lower value (global cost). In general, the two random variables that approximate the global cost distribution in these two scenarios will be denoted by X, for scenario number 1, and Y for the scenario number 2. The probability that the random variable X is lower than Y is

$$P(X < Y) = \int_{-\infty}^{\infty} P(X \leq Y|y) \cdot f_Y(y)dy.$$

or equivalently

$$P(X < Y) = \int_{-\infty}^{\infty} F_X(y) \cdot f_Y(y)dy. \tag{5.37}$$

Thus, $P(X < Y)$ represents the probability that the first global cost X will obtain a lower global cost than in the second case. Also, this probability provides information about how much better one algorithm is in comparison with the other one.

The previous formula can be extended to compare N number of cases. In this situation, the global costs are approximated by the probabilistic distributions $X_1, X_2, X_3, \ldots, X_N$. The probability that the global cost X_i will be lower than the rest of the global costs is

$$P(X_i < X_{-i})$$

where $X_{-i} = \{X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_N\}$. Assuming that all the global costs are independent, then

$$P(X_i < X_{-i}) = \prod_{j \neq i} P(X_i < X_j),$$

where each of these probabilities has the same form as in (5.37).

This probability, $P(X_i < X_{-i})$, is calculated for all the cases under study, and the one with the highest probability is chosen, i.e., the best case will be X_i iff

$$P(X_i < X_{-i}) > P(X_j < X_{-j}) \quad \forall j \neq i.$$

The BS-WR algorithm has been applied to two different scenarios with 10 robots: a dispersion scenario with the costs following a uniform distribution $[0, 1000]$, and a random scenario (where the robots and tasks are initially positioned randomly in a square area) with a square area of 707.1 units per side. Therefore, in both cases, the costs range from 0 to 1000 units but with different distributions. Then, the mean and the variance are calculated using the probabilistic approach:

- Dispersion scenario (X): $\mu_1 = 2019.9$ and $\sigma_1^2 = 524.69^2$.
- Random scenario (Y): $\mu_2 = 2174.1$ and $\sigma_2^2 = 484.13^2$.

These values are used to approximate the global costs distributions using a normal distribution. Afterwards, both cases are compared analytically and it is calculated $P(X < Y)$ which is equal to 0.5854. Therefore, the BS-WR algorithm has a highest probability to obtain a lower global cost in the dispersion scenario. Moreover, the BS-WR algorithm, applied to the dispersion scenario, will obtain in 58.54% of the experiments better results than in the random scenario. It is important to point out that this percentage has been calculated without running a single simulation.

Finally, the BS-WR algorithm has been simulated for 10000 experiments for both scenarios. It has been verified that the BS-WR algorithm obtains better results for the dispersion scenario in 58.16% of the experiments which is very similar to the theoretical value. Therefore, this approach can be used to compare analytically the same algorithm in different scenarios or different algorithms in the same scenario without the need of performing thousands of simulations.

5.9 Chapter summary

In this chapter, a probabilistic analysis approach has been developed for MRST market-based algorithms. This analysis consists of calculating the expected value of the global cost which is later used as a descriptor to compare different algorithms. Since the expected value is a measure of the behavior of the algorithm "in average", it becomes a more informative descriptor of the performance of an algorithm. This is in comparison to the typical measure carried out in worst-case analyses, which only provides a comparison of "bad" performance with respect to the optimal solution.

This methodology has been applied to two different algorithms within different scenarios. The chosen algorithms are representative of MRST algorithms with and without task reallocations. For each pair algorithm-scenario, theoretical formulae of the cumulative distribution (and density) function of the cost, of the minimum cost, and of the expected value of the global cost have been developed. The uniform distribution, a probabilistic density usually applied for modeling, has been widely reviewed, and the derivation of the distribution followed by the sum of squared-uniform random variables has been carried out. In order to make the analysis more flexible, the mixture of probability distributions has been introduced. The obtained formulae have been validated with simulated data, and with experiments considering real robots. The performance of the market-based algorithms has also been tested, by comparing with the optimal solution (either theoretically or via simulations).

Next, an extension of the probabilistic study has been developed for the BS-WR algorithm. First, the variance of the global cost distribution has been calculated. This fact allows to obtain a better knowledge of the algorithm behavior since the variance provides information about how much the results may divert from its mean. The theoretical formula of the variance has been validated with simulated data.

Finally, the theoretical mean and variance have been used to describe the normal distribution that approximates the global cost random distribution. Also, an analytical procedure has been explained to compare two or more cases. These cases can be one algorithm in different scenarios, or different algorithms in the same scenario. The theoretical procedure has been validated with the BS-WR algorithm applied to two scenarios (dispersion and random) with very similar results between the simulations and the theoretical values.

Chapter 6

Conclusions and future work

This chapter presents the thesis conclusions and future developments. A summary of the main contributions and an analysis of the objectives achieved are first described. Next, future research activities to extend the work presented in this thesis are detailed.

6.1 Summary of contributions

The aim of this thesis is the development of distributed task allocation algorithms that solves the multi-robot task allocation (MRTA) problem. These algorithms have to be fault tolerant and obtain solutions close to the optimal. A market-based approach has been used since it offers a good compromise between communication requirements and the quality of the solution.

First, a novel task allocation algorithm (S+T) that considers services for the distributed solution of the MRTA problem has been presented. This protocol allows a team of robots to achieve tasks that could not be executed by a single robot. It does not just coordinate the robots but rather introduce cooperation among them in order to increase the capabilities of the robot team. Also, a modification of this algorithm that allows to give priority either to the execution time or to the global cost of the mission has been explained. The problem of execution loops that appears from the relation between tasks and services has been stated and a distributed algorithm that solves this problem presented.

The S+T protocol has been implemented, simulated and tested in a demonstration with three real robots. Simulations have shown that when the number of services increases, the number of messages and the global cost increases, which means that the communication requirements and the energy to execute the mission will be higher. Also, the use of a larger number of services provokes that the maximum cost per robot increases, and therefore, the total time of the mission will be larger. Moreover, it has been shown the effects of some values of the parameter α used to adapt our S+T algorithm to different types of objectives. Finally, regarding the demonstration, it should be pointed out that this is the first experiment of these characteristics and it involved not only exploration, detection and monitoring, but also actuation in order to extinguish the fire.

Two algorithms for the distributed solution of the MRTA problem have been presented. In both algorithms, robots have a local plan and multiple tasks can be allocated to a single robot during the negotiation. Moreover, the second algorithm (SET-MASR), based on the negotiation of subsets of tasks, can be considered as a generalization of the first one (which only negotiates single tasks) designed to improve the solutions. From the simulation results, it is derived that using a local plan during the auction process improves the solutions significantly. Furthermore, the algorithms presented lead to good results when comparing with the global optimal solutions in missions consisting in visiting waypoints. Moreover, the SET-MASR algorithm computes better solutions in mean than the SIT-MASR, and both algorithms performance scale well when the number of robots and tasks increases.

On the other hand, five different MRST algorithms have been developed. The first two (BS-WR and BS) are just an adaptation of the market approach to algorithms without a local plan. However, the other three are original work, and it has been proven that they obtain better results than the basic algorithms. The algorithms have been extensively tested in simulation. Moreover, simulations in a realistic environment have been presented where the task allocation algorithm has been integrated with a path planning algorithm, and the execution of the tasks are within a behavior architecture that combines a path following algorithm with obstacle avoidance. Results from experiments with real robots with and without obstacles have been used to prove that our algorithms work in noisy and realistic environments.

Although the performance of the different algorithms have been evaluated in simulation, it is always more interesting to have a theoretical result that gives an idea of the algorithm performance. One of the main contributions of this thesis is a general probabilistic analysis. This analysis can be applied to any of the presented MRST algorithms in different scenarios. The work presented is unique in the sense that it calculates the expected value of the global cost which is later used as a descriptor to compare different algorithms. Since the expected value is a measure of the behavior of the algorithm "in average", it becomes a more informative descriptor of the performance of an algorithm than the typical measure carried out in worst case analyses, which only provides a comparison of "bad"-performance with respect to the optimal solution.

This methodology has been applied to two different algorithms within different scenarios. The chosen algorithms are commonly found in the literature to represent the typical implementations of market-based algorithms, both with and without task reallocations. For each pair algorithm-scenario, theoretical formulas of the cumulative distribution (and density) function of the cost, of the minimum cost, and of the expected value of the global cost have been developed. In order to make our analysis more flexible, we have introduced the mixture of probability distributions. The obtained formulas have been validated with simulated data, and implemented with real robots. The performance of the market-based algorithms has also been tested, by comparing with the optimal solution (either theoretically or via simulations). In conclusion, this novel methodology provides a new avenue for modeling algorithms that solve the multi-robot task allocation (MRTA) problem; not only defines an alternative and potential framework, but also can be implemented (by applying the same procedure for derivation of theoretical formulas) for other probabilistic distributions that might be of interest for researchers, depending on the situation.

Finally, a complete probabilistic study for the BS-WR algorithm has been developed, where the variance and an approximation of the global cost distribution have been calculated. The theoretical mean and variance have been used to describe the normal distribution that approximates the real one. Once the random variable of the global cost has been modeled with a normal distribution, an analytical procedure has been explained to compare two or more cases. These cases can be one algorithm in different scenarios, or different algorithms in the same scenario. The theoretical procedure has been validated with the BS-WR algorithm applied to two scenarios (dispersion and random) with very similar results between the simulations and the theoretical values. It is important to point out that these are the first steps dealing with the performance comparison of distributed task allocation algorithms without the use of simulations.

6.2 Future work

Future plans consist of extensions of the work presented in this thesis, as well as exploration of new problems raised while developing these techniques. In the following, future research plans are outlined.

6.2.1 Task recovery

In the near future, it could be interesting to develop distributed algorithms that can recover the allocated tasks from robots that lost all their communication capabilities. This type of algorithm together with the presented market-based algorithms will increase the fault tolerance capabilities of the system, making it possible to recover from any type of failure.

There are interesting algorithms from the dependable distributed systems field of study that I think can be used in these situations. For example, algorithms based on quorums (Giord, 1979) which is a technique used for data replication in distributed systems. The basic protocol was designed for distributed storage using clients and servers. However, it could be interesting to explore whether this type of algorithm can be adapted to a multi-robot system. This algorithm should be able to recover the tasks allocated to a uncommunicated robot without the need for a central server that knows the allocations of all the tasks, or the need to save all the tasks in every robot.

6.2.2 Extension of the probabilistic performance analysis

This novel methodology provides a new avenue for modeling algorithms that solve the MRTA problem. In this respect, a number of extensions are possible. Firstly, it is aimed to extend these results to more complex task allocation algorithms, such as MRMT algorithms where more than one task can be allocated to a single robot. However, one question still unanswered is whether this approach can model market-based scenarios that make use of marginal costs (Dias et al., 2006).

On the other hand, the proposed methodology could be used to predict the performance of the algorithms when a set of robots fails. This problem can be modeled as a generalization of the assignment problem called the k-assignment problem (Coppersmith and Sorkin, 1999), where k is the difference between the total number of robots and the number of robots that fails.

Moreover, the fact that there are few closed-forms of the expected value of the optimal global cost opens a new line of future work, whose aim would be to calculate it for different cost distributions.

Finally, an interesting perspective would be to consider the situation where costs vary with time (time-based cost situations). In this case, the theory of the so-called *stochastic processes* would be needed. Within this framework the value of the cost at time t is denoted by X_t and, both transient analyses ($t < \infty$) or stationary ones ($t \to \infty$) can be carried out to analyze the behavior of the global cost.

6.2.3 Probabilistic costs

It could be interesting to explore the idea of using probabilistic distributions, instead of deterministic numbers, to model task costs. Therefore, the complete market-based algorithm changes its philosophy. For example, when one robot has to decide which task wants to keep, instead of comparing two numbers, probabilistic theory has to be used to calculate which task has the highest probability to have the lowest cost.

It is true that the market-based algorithm will become more complicated, but it is thought that this new approach will be able to obtain better results when there is a large uncertainty associated to task costs.

Appendix A

Multi-robot architecture

A complete system architecture has been developed in order to make easier the integration of heterogeneous robots. This architecture is designed to reuse the common components to all the robots and minimize the time needed to incorporate a new robot to the system. Some of the robots that have been integrated into this architecture are: ROMEO-4R (Ollero et al., 1999), HERO (Ferruz et al., 2009) and iRobot Create (Viguria and Howard, 2009b).

Moreover, the same architecture has been used for simulated and real robots. Therefore, the same high level software can be used in both types of robots, with the only differences that in the simulated robot the hardware components are virtual devices (see Section B.2).

A.1 Multi-robot architecture overview

A global view of the architecture components is presented in Figure A.1, where three main blocks can be identified:

- **Monitoring and Planning Station** (MPS): provides means to the human operator for preparing plans, sending missions and monitoring its execution. It also encompasses the Alarm Monitoring Station, which is in charge of performing autonomous cooperative perception processing (Merino et al., 2006a), and specialized image processing activities (such as fire detection) providing different alarms to the operator. Finally, all the information related to each mission is saved in a database for mission debriefing purposes.

- **Communication Network**: is the support for every communication between the different components of the system. It deals with task requests/status and data transmissions, such as images or robots telemetry. It should be noted that the robots currently integrated in the architecture are using the BBCS (BlackBoard Communication System) developed by the Technical University of Berlin (Remußet al., 2004). It is a robust communication system implemented via a distributed shared memory, the blackboard (BB), in which each network node has a local copy of the BB portion it is accessing.

- **Robot Team**: the software architecture of each robot (see Figure A.2) is based on hierarchical layers, with the higher levels "decoupled" from the particular characteristics of the robot. Therefore, the high level software modules can be reused in different types of robots with minor changes. Three layers have been developed: RAL, MML and RIL. Since this thesis only deals with high level aspects of the robot team behavior, only the *Robot Abstraction Layer* (see Figure A.2) will be commented in this appendix. The MML layer manages the modules that implement the robot functionalities that are implemented in the RIL layer. For example, the MML layer starts and stops the necessary modules (RIL) for each task, makes the appropriate connections between the modules and passes the correct parameters. For more information about the MML and RIL layers see (Viguria and Maza, 2006).

Regarding the task allocation process it should be pointed out that this architecture supports two different modes of operation (see Figure A.2):

- Manual allocation mode: the human operator allocates individual elementary tasks and sequences of elementary tasks from the MPS to the robots. Each *Robot Abstraction Layer* (RAL) manages those tasks and reports the robot status during the mission execution.

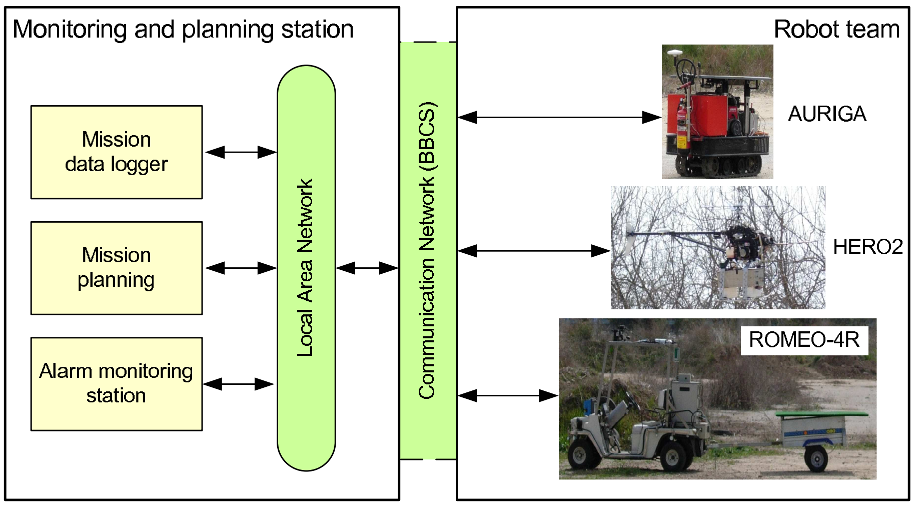


Figure A.1: Global architecture illustrated with the vehicles used in the demonstration described in Section 2.4. The communication network is used for both the communication between the robots and the communication between the robot team and the monitoring and planning station.

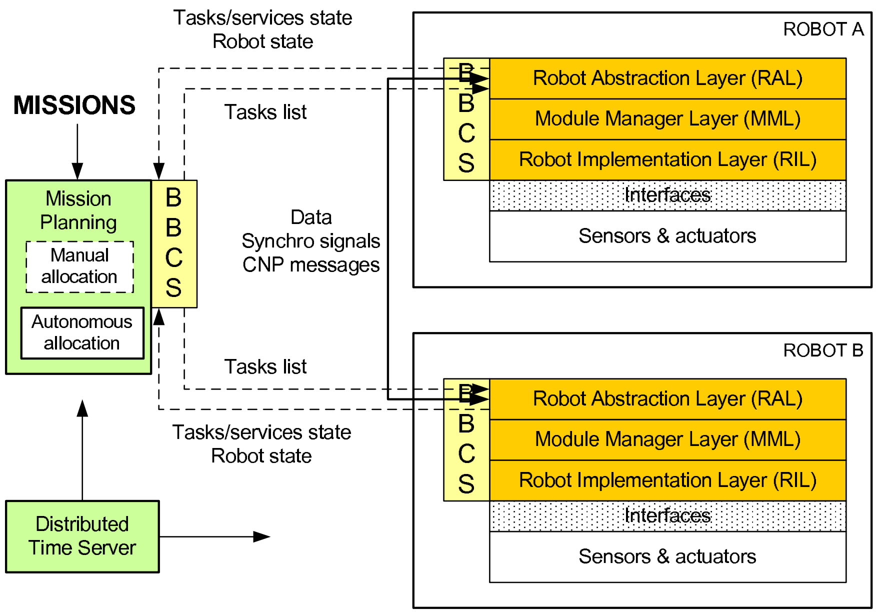


Figure A.2: Robot team architecture (*dashed lines* correspond to the manual allocation mode and *solid lines* to the autonomous allocation mode). Each robot has a layered architecture with three layers: RAL, MML and RIL. The RAL layer deals with task allocation and synchronization issues.

- Autonomous allocation mode: a Distributed Task Allocation Module in the RAL allows to autonomously negotiate task allocation in a distributed way. In this mode, the MPS should only provide a list of elementary tasks to be executed by a group of robots.

The system architecture supports the use of both modes of operation during a mission execution, allowing the operator for example to manually allocate a task to a given robot whereas the rest of tasks are being allocated autonomously.

A.2 Robot Abstraction Layer (RAL)

As the RAL has been designed to be mainly independent from the particular characteristics of the robot, a similar implementation can be used in different heterogeneous robots. Some modules in this layer are:

- Distributed Task Allocation Module (DTAM): this module allows robots to autonomously negotiate task allocations in a distributed way by using a market-based approach. At this point, different auction algorithms have been implemented: SIT-MASR, SET-MASR (Viguria et al., 2007), S+T (Viguria et al., 2008), BS-WR, BS, RMA, TMA, and RTMA (Viguria, 2008) and (Viguria and Howard, 2009b).

- Task Manager Module: this module manages tasks and their states. It receives tasks from the MPS (manual allocation mode) or from the DTAM (autonomous allocation mode). Furthermore, it sends basic tasks to the lower software layers and reports the state of the tasks to the MPS.

- Task Synchronization Module: tasks can be synchronized using preconditions, i.e., a task will not be executed until all its preconditions are satisfied. Once every task is allocated, the Task Manager Module communicates the preconditions of each task to this module. Before a task is going to be executed, the Task Manager Module asks to this module if the task can start. This information is available because when a robot finishes a task, it will transmit a message to the rest of the team.

- Environment Model Module: it builds a local map of the environment and transmit it to other robots in order to combine them and generate a more complete and accurate model for the whole team (Merino et al., 2006a), (Merino et al., 2006b).

Appendix B

Multi-robot task allocation testbed

A team of 6 mobile robots are used in order to test the presented task allocation algorithms. Each of these robots is based on the iRobot Create platform (see Figure B.1). This platform is ideal for the robotics research community due to its low price and open communication protocol with access to all its sensors and actuators. However, this platform is only suitable for indoor or flat terrain outdoor experiments.

B.1 Hardware description

A micro linux computer, the Connex 400XM processor from Gumstix, has been added to the platform. This motherboard contains a 400MHz ARM processor, two serial ports compatible with TTL levels, and bluetooth capabilities. Additionally, two more boards were used: a Robostix board was added to have easy access to the serial ports and power supply pins, and a Wifistix board that provides wireless 802.11b/g capabilities to the linux computer. The Robostix board also includes an Atmel ATMega 128 RISC microcontroller, providing both SPI and I2C serial ports, general purpose IO pins, PWM outputs, and an ADC unit. These characteristics enable the Robostix to function as an ideal board to implement low level controllers or to be used as an interface between the robot sensors and the linux computer. The three boards create a compact, cheap and small embedded computer suitable for applications that involve small robots. The iRobot battery was used to power the embedded computer.

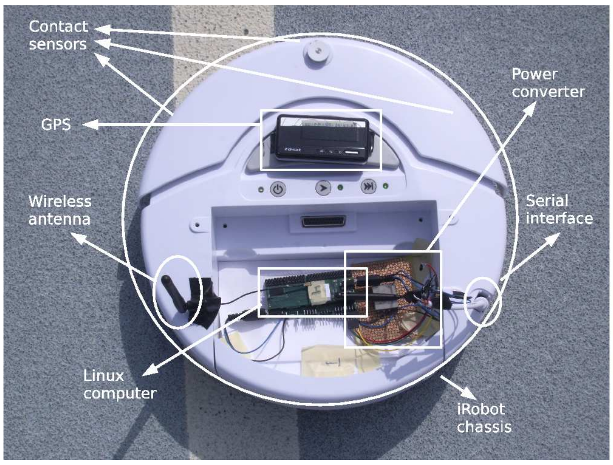


Figure B.1: Robot used in the experiments. iRobot Create upgraded with micro linux computer, wireless communication and GPS.

A DC/DC converter was used to stabilize the voltage of the iRobot battery and bring it down to the required level.

With respect to the sensors, a GPS unit with a bluetooth interface was added to each robot as the main localization sensor. The GPS is an off-the-shelf product designed for cell phones and PDAs with meter precision. The only problem with this GPS was that it presented some trouble dealing with bluetooth interference between the different GPS units. Since the quality of the GPS is not good enough, a Kalman Filter (Thrun et al., 2005) is used to combine the local odometry and the GPS measurements to obtain a decent global localization. The local odometry was obtained from the wheel encoders using the open communication protocol. The last sensors are the three front contact sensors that come with the iRobot platform (see Figure B.1). The actuators are the electric motors that move the wheels in a differential drive configuration. Information between the robots is exchanged over the bidirectional wifi link using standard UDP sockets.

B.2 Software architecture

The multi-robot architecture explained in Appendix A has been used for the integration of the task allocation algorithms in the real robots. This fact combined with the implementation of a Player driver (Gerkey et al., 2003) to control the iRobot platform enabled the use of the same software both in simulation (using Player/Gazebo) and real experiments. After the allocation process has concluded, the high level algorithm passes a path to the robot controller. The relation between the task allocation and path planner modules is the same as shown in Figure 4.8.

In this case, the robot controller is composed of two behaviors: obstacle detection and path follower. Both behaviors are combined using a DAMN architecture. The first behavior is used to move the robot backwards when one of the contact sensors is activated. The second behavior uses the Pure Pursuit algorithm (Ollero and Amidi, 1991) to follow the received path. The votes from both behaviors are combined by an arbiter that communicates the desired speed and turn rate to the Player driver (see Figure B.2). Then, the Player driver communicates these values (as left and right wheel speeds) to the iRobot platform by means of the serial port and using the open communication interface. This Player driver also reads the NMEA data units from the GPS and the sensor values transmitted by the iRobot platform, and transforms them to the data structures used in the Player project. These data structures are combined in the Kalman Filter module to obtain an estimated global position that is used by the Path Follower module. One of the main advantages in using software from the Player project with commercial robots was that the development time was reduced significantly since the low level communication with sensors and actuators was already implemented in the open source project.

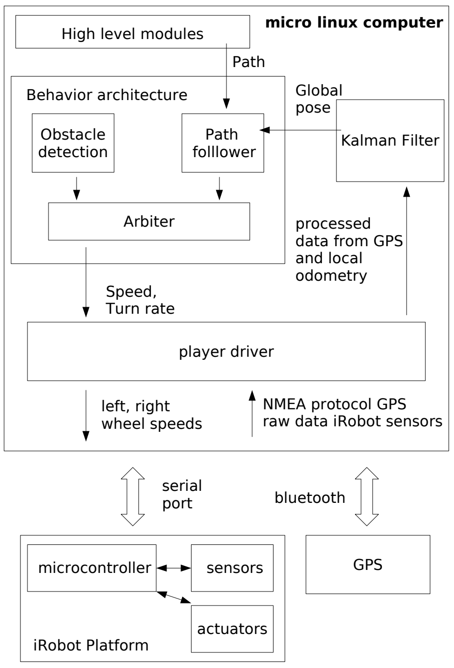


Figure B.2: Scheme that shows the integration of the micro linux computer with the robot platform and its sensors. It also depicts the relation between the different modules that implement the robot controller.

References

Agassounon, W. and Martinoli, A. (2002). Efficiency and Robustness of Threshold-based Distributed Allocation Algorithms in Multi-Agent Systems. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1090–1097, Bologna, Italy.

Aldous, D. J. (2001). The $\zeta(2)$ Limit in The Random Assignment Problem. *Random Structures Algorithms*, 18(4):381–418.

Alighanbari, M. How, J. P. (2005). Decentralized Task Assignment for Unmanned Aerial Vehicles. In *Proceedings of the IEEE Conference on Decision and Control and European Control Conference*, pages 5668–5673, Seville, Spain.

Alighanbari, Mehdi How, J. P. (2008). A Robust Approach to the UAV Task Assignment Problem. *International Journal of Robust and Nonlinear Control, Special Issue on Cooperative of Unmanned Aerial Vehicles*, 18(2):118–134.

Applegate, D., Cook, W., Dash, S., and Rohe, A. (2002). Solution of a Min-Max Vehicle Routing Problem. *INFORMS Journal on Computing*, 14(2):132–143.

Arrue, B., Martínez-de Dios, J., and Ollero, A. (2000). An Intelligent System for False Alarm Reduction in Infrared Detection of Forest Fires. *IEEE Intelligent Systems*, 15(3):64–73.

Atay, N. and Bayazit, B. (2007a). Emergent Task Allocation for Mobile Robots. In *Proceedings of Robotics: Science and Systems*, Atlanta, USA.

Atay, N. and Bayazit, B. (2007b). Emergent Task Allocation for Mobile Robots Through Intentions and Directives. Technical report, Dept. of Computer Science and Engineering, Washington University in St. Louis, St. Louis, USA.

Beard, R. W., McLain, T. W., Nelson, D. N., Kingston, D., and Johanson, D. (2006). Decentralized Cooperative Aerial Surveillance Using Fixed-Wing Miniature UAVs. *Proceedings of the IEEE*, 94(7):1306–1324.

Bertsekas, D. P. (1981). A New Algorithm for the Assignment Problem. *Mathematical Programming*, 21(1):152–171.

Bethke, B., Valenti, M., and How, J. P. (2008). UAV Task Assigment. *IEEE Robotics and Automation Magazine*, 15(1):39–44.

Botelho, S. and Alami, R. (2001). Multi-Robot Cooperation Through the Common Use of "Mechanisms". In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 375–380, Maui, USA.

Botelho, S. C. and Alami, R. (1999). M+: A Scheme for Multi-Robot Cooperation Through Negotiated Task Allocation and Achievement. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1234–1239, Detroit, USA.

Brooks, R. A. (1986). A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23.

Brumitt, B. and Stenz, A. (1998). GRAMMPS: A Generalized Mission Planner for Multiple Mobile Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1564–1571, Leuven, Belgium.

Calafiore, G., Dabbene, F., and Tempo, R. (1998). Uniform Sample Generation in l_p Balls for Probabilistic Robustness Analysis. In *Proceedings IEEE Conference on Decision and Control*, pages 3335–3340, Tampa, USA.

Caloud, P., Choi, W., Latombe, J., Le Pape, C., and Yim, M. (1990). Indoor Automation with Many Mobile Robots. In *Proceedings of the IEEE International*

Workshop on Intelligent Robotics and Systems (IROS), volume 1, pages 67–72, Ibaraki, Japan.

Cao, Y. U., Fukunaga, A. S., and Kahng, A. (1997). Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots*, 4(1):7–27.

Coppersmith, D. and Sorkin, G. B. (1999). Constructive Bounds and Exact Expectations For the Random Assignment Problem. *Random Structures and Algorithms*, 15(2):133–144.

Dahl, T. S., Matarić, M. J., and Sukhatme, G. (2008). Multi-Robot Task Allocation Through Vacancy Chain Scheduling. *Robotics and Autonomous Systems*. DOI: 10.1016/j.robot.2008.12.001.

Dias, M. (2004). *TraderBots: A New Paradigm for Robust and Efficient MultiRobot Coordination in Dynamic Environments*. PhD thesis, Carnegie Mellon University.

Dias, M., Zinck, M. B., Zlot, R. M., and Stentz, A. (2004). Robust Multirobot Coordination in Dynamic Environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3435–3442, New Orleans, USA.

Dias, M., Zlot, R., Karla, N., and Stentz, A. (2006). Market-based Multirobot Coordination: A Survey and Analysis. *Proceedings of the IEEE*, 94(7):1257–1270.

Dias, M. B. and Stenz, A. (2002). Opportunistic Optimization for Market-based Multirobot Control. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2714–2720, Lausanne, Switzwerland.

Dias, M. B. and Stenz, A. (2003). A Comparative Study between Centralized, Market-based, and Behavioral Multirobot Coordination Approaches. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2279–2284, Las Vegas, USA.

Dudek, G., Jenkin, M., and Milios, E. (2002). *Robot Teams: From Diversity to Polymorphism*, chapter A Taxonomy of Multirobot Systems. A. K. Peters Ltd.

Farinelli, A., Iocchi, L., and Nardi, D. (2004). Multirobot Systems: A Classification Focused on Coordination. *IEEE Transactions on Systems, Man and Cybernetics–Part B: Cybernetics*, 34(5):2015–2028.

Farinelli, A., Iocchi, L., Nardi, D., and Ziparo, V. A. (2005). Task Assignment with Dynamic Perception and Constrained Tasks in a Multi-Robot System. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1523–1528, Barcelona, Spain.

Farinelli, A., Iocchi, L., Nardi, D., and Ziparo, V. A. (2006). Assignment of Dynamically Perceived Tasks by Token Passing in Multirobot Systems. *Proceedings of the IEEE*, 94(7):1271–1288.

Ferruz, J., Ollero, A., and Vega, V. (2009). Embedded Control and Development System for the HERO Autonomous Helicopter. In *Proceedings of the IEEE International Conference on Mechatronics*, Malaga, Spain.

Fua, C.-H. and Ge, S. S. (2005). COBOS: Cooperative Backoff Adaptive Scheme for Multirobot Task Allocation. *IEEE Transactions on Robotics*, 21(6):1168–1178.

Fua, C.-H., Ge, S. S., and Lim, K. W. (2004). BOAS: BackOff Adaptative Scheme for Task Allocation with Fault Tolerance and Uncertainty Management. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 162–167, Taipei, Taiwan.

Gerkey, B. and Matarić, M. J. (2000). MURDOCH: Publish/Subscribe Task Allocation For Heterogeneous Agents. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 203–204, Barcelona, Spain.

Gerkey, B. and Matarić, M. J. (2002). Sold!: Auction Methods for Multi-Robot Coordination. *IEEE Transactions on Robotics and Automation, Special Issue on Multi-Robot Systems*, 18(5):758–768.

Gerkey, B. and Matarić, M. J. (2004). A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *International Journal of Robotics Research*, 23(9):939–954.

Gerkey, B., Vaughan, R. T., and Howard, A. (2003). The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. In *Proceedings of the 11th International Conference on Advanced Robotics (ICAR)*, pages 317–323, Coimbra, Portugal.

Gil, A., Passino, K., Ganapathy, S., and Sparks, A. (2003). Cooperative Scheduling of Tasks for Networked Uninhabited Autonomous Vehicles. In *Proceedings of the IEEE Conference on Decision and Control*, volume 1, pages 522–527, Maui, USA.

Giord, D. (1979). Weighted Voting for Replicated Data. In *Proceedings of the 7th ACM Symposium on Operating Systems Principles*, pages 150–162, Pacific Grove, USA.

Golfarelli, M., Maio, D., and Rizzi, S. (1997). A Task-Swap Negotiation Protocol based on the Contract Net Paradigm. Technical report, CSITE (Research Centre For Informatics And Telecommunication Systems), University of Bologna, Bologna, Italy.

Guerrero, J. and Oliver, G. (2004). Multi-Robot Task Allocation Method for Heterogeneous Tasks with Priorities. In *Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, Toulouse, France.

Howard, A., Seraji, H., and Werger, B. (2005). Global and Regional Path Planners for Integrated Planning and Navigation. *Journal of Robotic Systems*, 22(12):767–778.

Howard, A. and Viguria, A. (2007). Controlled Reconfiguration of Robotic Mobile Sensor Networks using Distributed Allocation Formalisms. In *NASA Science Technology Conference (NSTC)*, Maryland, USA.

Hu, X. and Egerstedt, M. (2001). Formation Constrained Multi-Agent Control. *IEEE Transactions on Robotics and Automation*, 17(6):947–951.

Jamil, M., Batta, R., and Malon, D. (1994). The Traveling Repairperson Home Base Location Problem. *Location Science*, 28(2):150–161.

Ji, M. and Egerstedt, M. (2006). Role-Assignment in Multi-Agent Coordination. *International Journal of Assistive Robotics and Mechatronics*, 7(1):32–40.

Kalra, N., Stentz, A., and Ferguson, D. (2005). Hoplites: A Market Framework for Complex Tight Coordination in Multi-Agent Teams. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1170–1177, New Orleans, USA.

Kalra, N., Stentz, A., and Ferguson, D. (2007). A Generalized Framework for Solving Tightly-coupled Multirobot Planning Problems. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1050–4729, Rome, Italy.

Karla, N. and Martinoli, A. (2006). A Comparative Study of Market-based and Threshold-based Task Allocation. In *Proceedings of the 8th International Symposium on Distributed Autonomous Robotic Systems (DARS)*.

Ko, J., Stewart, B., Fox, D., Konolige, K., and Limketkai, B. (2003). A Practical Decision-Theoretic Approach to Multi-Robot Mapping and Exploration. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 3232–3238, Las Vegas, USA.

Koenig, S., Tovey, C., Zheng, X., and Sungur, I. (2007). Sequential Bundle-Bid Single-Sale Auction Algorithms for Decentralized Control. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1359–1365.

Krieger, M. J. B. and Billeter, J. B. (2000). The Call of Duty: Self-Organized Task Allocation in a Population of Up to Twelve Mobile Robots. *Robotics and Autonomous Systems*, 30(1–2).

Kuhn, H. (1955). The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly 2*, pages 83–97.

Kurtzberg, J. M. (1962). On Approximation Methods for the Assigment Problem. *Journal of the ACM*, 9(4):419–439.

Lagoudakis, M., Berhault, M., Koenig, S., Keskinocak, P., and Kleywegt, A. (2004). Simple Auctions with Performance Guarantees for Multi-Robot Task Allocation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 698–705, Sendai, Japan.

Lagoudakis, M., Markakis, V., Kempe, D., Keskinocak, P., Koenig, S., Kleywegt, A., Tovey, C., Meyerson, A., and Jain, S. (2005). Auction-based Multi-Robot Routing. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA.

Laporte, G. and Nobert, Y. (1980). *The Multi-Depot Travelling Salesman Problem.* C.R.T.

LaValle, S. M. and Kuffner, J. J. (2001). Randomized Kinodynamic Planning. *International Journal of Robotics Research*, 20(5):378–400.

Lawler, E. L. (1985). *The Traveling Salesman Problem.* John Wiley and Sons.

Lawton, J., Beard, R., and Young, B. (2003). A Decentralized Approach to Formation Maneuvers. *IEEE Transactions on Robotics and Automation*, 19(6):933–941.

Lemaire, T., Alami, R., and Lacroix, S. (2004). A Distributed Task Allocation Scheme in Multi-UAV Context. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3622–3627, New Orleans, USA.

Lerman, K., Jones, C., Galstyan, A., and Matarić, M. J. (2006). Analysis of Dynamic Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, 25(3):225–241.

Lima, P. U. and Custodio, L. M. M. (2004). Artificial Intelligence and Systems Theory: Applied to Cooperative Robots. *International Journal of Advanced Robotic Systems*, 1(3):141–148.

Lin, L. and Zheng, Z. (2005). Combinatorial Bids based Multi-robot Task Allocation Method. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1145–1150, Barcelona, Spain.

Lynch, N. A. (1996). *Distributed Algorithms*. Morgan Kaufmann.

M., B., Huang, H., Keskinocak, P., Koenig, S., Elmaghraby, W., Griffin, P., and Kleywegt, A. (2003). Robot Exploration with Combinatorial Auctions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1957 – 1962, Las Vegas, USA.

Maza, I., Viguria, A., and Ollero, A. (2006). Networked Aerial-ground Robot System with Distributed Task Allocation for Disaster Management. In *Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics*, Gaithersburg, USA.

McLurkin, J. and Yamins, D. (2005). Dynamic Task Assignment in Robot Swarms. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA.

Merino, L. (2008). *Cooperative Perception Techniques for Multiple Unmanned Aerial Vehicles: Applications to the Cooperative Detection, Localization and Monitoring of Forest Fires*. PhD thesis, University of Seville.

Merino, L., Caballero, F., Martínez-de Dios, J., Ferruz, J., and Ollero, A. (2006a). A Cooperative Perception System for Multiple UAVs: Application to Automatic Detection of Forest Fires. *Journal of Field Robotics*, 23(3):165–184.

Merino, L., Caballero, F., Wiklund, J., Moe, A., Martínez-de Dios, J., Forssen, P., Nordberg, K., and Ollero, A. (2006b). Vision-based Multi-UAV Position Estimation. *IEEE Robotics and Automation Magazine*, 13(3):53–62.

Mézard, M. and Parisi, G. (1987). On The Solution of The Random Link Matching Problems. *Journal de Physique*, 48(9):1451–1459.

Mosteo, A. R. and Montano, L. (2007). Comparative Experiments on Optimization Criteria and Algorithms for Auction based Multi-Robot Task Allocation. In

Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pages 3345–3350, Rome, Italy.

Mosteo, A. R., Montano, L., and Lagoudakis, M. G. (2008). Multi-Robot Routing under Limited Communication Range. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1531–1536, Pasadena, USA.

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38.

Nanjanath, M. and Gini, M. (2006). Dynamic Task Allocation for Robots via Auctions. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2781–2786, Orlando, USA.

Nilson, N. (1971). *Problem-Solving Methods in Artificial Intelligence.* McGraw-Hill.

Ollero, A. and Amidi, O. (1991). Predictive Path Tracking of Mobile Robots. In *Proceedings of 5th International Conference on Advanced Robotics, Robots in Unstructured Environments (ICAR)*, volume 2, pages 1081–1086.

Ollero, A., Arrue, B., Ferruz, J., Heredia, G., Cuesta, F., Lopez-Pichaco, F., and Nogales, C. (1999). Control and Perception Components for Autonomous Vehicle Guidance. Application to the ROMEO vehicles. *Control Engineering Practice*, 7(10):1291–1299.

Parker, L. E. (1997). L-ALLIANCE: Task-oriented Multirobot Learning in Behavior-based Systems. *Advanced Robotics*, 11(4):305–322.

Parker, L. E. (1998). ALLIANCE: An Architecture for Fault-Tolerant Multi-Robot Cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

Parker, L. E. (2008). Distributed Intelligence: Overview of the Field and its Application in Multi-Robot Systems. *Journal of Physical Agents*, 2(1):5–14.

Remuß, V., Musial, M., and Brandenburg, U. (2004). BBCS Robust Communication System for Distributed System. In *Proceedings of the International Workshop on Safety, Security, and Rescue Robotics*, Bonn, Germany.

Report, N. (2002). National workshop on future sensing systems - living, nonliving, and energy systems. Technical report, National Science Foundation, Lake Tahoe, NV.

Rosenblatt, J. K. (1997). DAMN: a Distributed Architecture for Mobile Navigation. *Journal of Experimentation and Theoretical Artificial Intelligence*, 9(2):339–360.

Sandholm, T. (1993). An Implementation of the Contract Net Protocol based on Marginal Cost Calculations. In *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, pages 295–308, Hidden Valley, USA.

Sariel, S. and Balch, T. (2006a). A Distributed Multi-Robot Cooperation Framework for Real Time Task Achievement. In *Proceedings of the 8th International Symposium on Distributed Autonomous Robotic Systems (DARS).*

Sariel, S. and Balch, T. (2006b). Efficient Bids on Task Allocation for Multi-Robot Exploration. In *Proceedings of the 19th International FLAIRS Conference*, Melbourne Beach, USA.

Sariel, S., Balch, T., and Erdogan, N. (2006a). Robust Multi-Robot Cooperation Through Dynamic Task Allocation and Precaution Routines. In *The 3rd International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, Minneapolis, USA.

Sariel, S., Balch, T., and Erdogan, N. (2008). Naval Mine Countermeasure Missions. *IEEE Robotics and Automation Magazine*, 15(1):45–52.

Sariel, S., Balch, T., and J., S. (2006b). Empirical Evaluation of Auction-based Coordination of AUVs in a Realistic Simulated Mine Countermeasure Task. In *Proceedings of the 8th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, Minneapolis, USA.

Scerri, P., Farinelli, A., Okamoto, S., and Tambe, M. (2005). Token Approach for Role Allocation in Extreme Teams. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 727–734, The Netherlands.

Smith, G. (1980). The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Transactions on Computers*, C-29(12):1104–1113.

Smith, S. L. and Bullo, F. (2007). Target Assignment for Robotic Networks: Asymptotic Performance under Limited Communication. In *Proceedings of the American Control Conference*, pages 3585–3590, New York, USA.

Tabauda, P., Pappas, G. J., and Lima, P. U. (2005). Artificial Intelligence and Systems Theory: Applied to Cooperative Robots. *IEEE Transactions on Robotics*, 21(3):387–392.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press.

Toroslu, I. H. and Üçoluk, G. (2007). Incremental Assignment Problem. *Information Sciences*, 177(6):1523–1529.

Tovey, C., Lagoudakis, M. G., Jain, S., and Koenig, S. (2005). *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III, Proceedings from the 2005 International Workshop on Multi-Robot Systems*, chapter The Generation of Bidding Rules for Auction-based Robot Coordination, pages 3–14. Springer Netherlands.

Vecht, B. and Lima, P. (2004). Formulation and Implementation of Relational Behaviors for Multi-Robot Cooperative Systems. In *Proceedings of RoboCup 2004 Symposium*, Lisbon, Portugal.

Veloso, M., Stone, P., and Blowing, M. (1999). Anticipation as a Key for Collaboration in a Team of Agents: A Case Study in Robotic Soccer. In *Proceedings of SPIE*

Sensor Fusion and Decentralized Control in Robotic Systems, volume 3839, pages 134–141, Boston, USA.

Vig, L. and Adams, J. A. (2006). Multi-Robot Coalition Formation. *IEEE Transactions on Robotics*, 22(4):637–649.

Viguria, A. (2008). Distributed Task Allocation Methodologies for Solving the Initial Formation Problem. Master's thesis, Georgia Institute of Technology.

Viguria, A. and Howard, A. (2007). Upper Bound Analysis of a Cost Market-based Algorithm Applied to the Initial Formation Problem. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 2326–2331, San Diego, USA.

Viguria, A. and Howard, A. (2009a). A Probabilistic Model for the Performance Analysis of a Distributed Task Allocation Algorithm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3117–3122, Kobe, Japan.

Viguria, A. and Howard, A. (2009b). An Integrated Approach for Achieving Multi-Robot Task Formations. *IEEE/ASME Transactions on Mechatronics. Focused Section on Mechatronics in Multirobot Systems*, 14(2):176–186.

Viguria, A. and Howard, A. (2009c). Probabilistic Analysis of Market-based Algorithms for Initial Robotic Formations. *The International Journal of Robotics Research*. DOI: 10.1177/0278364909340333.

Viguria, A. and Maza, I. (2006). RealSim: Simulador MultiRobot Generico. Technical report, University of Seville, Spain.

Viguria, A., Maza, I., and Ollero, A. (2007). SET: An Algorithm for Distributed Multirobot Task Allocation with Dynamic Negotiation based on Task Subsets. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3339–3344, Rome, Italy.

Viguria, A., Maza, I., and Ollero, A. (2008). S+T: An Algorithm for Distributed Multirobot Task Allocation based on Services for Improving Robot Cooperation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3163–3168, Pasadena, EEUU.

Viguria, A., Maza, I., and Ollero, A. (2010). Distributed Service-based Cooperation in Aerial/Ground Robot Teams Applied to Fire Detection and Extinguishing Missions. *Advanced Robotics*. Accepted for Publication January 2010.

Walkup, D. W. (1979). On the Expected Value of A Random Assignment Problem. *SIAM Journal on Computing*, 8:440–442.

Werger, B. B. and Matarić, M. J. (2000). Broadcast of Local Eligibility for Multi-Target Observation. In *Distributed Autonomous Robotic Systems 4*, pages 347–356. Springer-Verlag.

Wurman, R. P., D'Andrea, R., and Mountz, M. (2008). Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses. *AI Magazine*.

Xu, Y., Scerri, P., Sycara, K., and Lewis, M. (2006). Comparing Market and Token-based Coordination. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1113–1115, Hakodate, Japan.

Xu, Y., Scerri, P., Yu, B., Okamoto, S., Lewis, M., and Sycara, K. (2005). An Integrated Token-based Algorithm for Scalable Coordination. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 407–414, The Netherlands.

Yun, S. and Rus, D. (2007). Optimal Distributed Planning of Multi-Robot Placement on a 3D Truss. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1365–1370, San Diego, USA.

Zavlanos, M. M. and Pappas, G. J. (2007). Dynamic assignment in Distributed Motion Planning with Limited Information. In *Proceedings of the American Control Conference*, pages 1173–1178, New York, USA.

Zheng, X., Koenig, S., and Tovey, C. (2006). Improving Sequential Single-Item Auctions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 2238–2244, Beijing, China.

Zlot, R. (2006). *An Auction-based Approach to Complex Task Allocation for Multi-robot Teams*. PhD thesis, Carnegie Mellon University.

Zlot, R. and Stentz, A. (2006). Market-based Multirobot Coordination for Complex Tasks. *International Journal of Robotics Research Special Issue on the 4th International Conference on Field and Service Robotics*, 25(1):73–101.