MULTI-ROBOT COALITION FORMATION

By

Lovekesh Vig

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

December, 2006

Nashville, Tennessee

Approved:

To my parents...

To friends in need...

ACKNOWLEDGEMENTS

This dissertation owes its completion to many quarters...

In Dr. Fisher, Dr. Noelle, Dr. Parker, Dr. Sarkar and Dr. Adams, I was fortunate to have had access to a committee of outstanding and eclectic researchers. Each views problems from entirely different perspectives and despite being such huge established figures in their respective fields, are equally approachable. At some point during the doctoral program, I suspect every graduate student has moments of self-doubt. I certainly had my fair share of such moments. I don't think I would have been able to work past them were it not for the fact that I had a very understanding and patient committee.

Coming to research, I would like to thank Dr. Fisher for my initiation into AI and also for his helpful insights into the project. Dr. Noelle for being such an exhaustive reservoir of knowledge and ideas, Dr. Parker for serving on my committee despite her packed schedule and Dr. Sarkar for his valuable feedback.

I would also like to thank Dr. Peter Molnar for his hospitality during my visits to the distributed robotics lab in Clark Atlanta University. It would not have been possible to get my experiments done without his enthusiastic support. The good people in the Player/Stage Community also have my gratitude, in particular Brian Gerkey for being so prompt to respond to any queries I had regarding the software.

Of course every word in this dissertation went under Dr. Adams' scrutinizing red pen. We often disagreed, but I must admit that her attention to detail makes the dissertation look much better. Prior to enrolling as a full time research assistant, I did not fully realize how difficult supervising a thesis or dissertation could be. My experience in graduate school has made me better appreciate the enormous responsibility that a supervisor is expected to shoulder. I can now also fully appreciate the outstanding job that Dr. Adams' did while guiding me these past years. Advisors often place students lower down in their list of priorities but Dr. Adams was extremely conscientious, I cannot recall a single instance when Dr. Adams was late for a meeting, or a paper revision, even when she was traveling.

She is also a terrific manager, I never felt overworked, and was never allowed to slack either. Professional matters aside, I will always be grateful to Dr. Adams' for her unwavering support during some very stressful and difficult times over the past two years.

On the subject of stress, I must emphasize how vitally important it is to have a close group of friends to interact with during the graduate program, both professionally and for recreation. I was fortunate to have made many close friends during my stay in Nashville and I would be remiss if I did not acknowledge their role in overcoming my homesickness, in mitigating stress and for never allowing me to feel isolated. In particular I would like to thank Akhil, Jelena, Avinash, Ashish, Aditya, Sameer, Sachin, Matt, Trent, Chris, Curtis, Karun, Ramya and Sebastian for making my stay in Nashville as pleasant as it has been. Finally, I would also like to thank Sandy and Lori for their friendly support and for being so wonderfully efficient in resolving any administrative difficulties.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The past decade has seen significant advances in the capabilities of multi-robot systems. Numerous mechanisms for coordination and cooperation of robots have evolved to enable ever increasing levels of autonomy. An issue that is pivotal to the performance of such cooperative multi-robot systems is task allocation. In particular, the problem of multi-robot task allocation has received considerable attention and many innovative schemes have been proposed for distributing tasks amongst a team of robots. Gerkey and Matarić (2003) provide a taxonomy for classifying Multi-Robot Task Allocation problems based on the number of tasks per robot (Single-Task **(ST)** or Multiple-Task **(MT)** robots), number of robots required for a task (Single-Robot **(SR)** or Multiple-Robot **(MR)** Tasks), and the schedule for allocation (Instantaneous **(IA)** or Time extended Allocation **(TA)**). Typically, the multi-robot task allocation problem comprises of a set of indivisible tasks and the problem involves assigning robots to tasks so as to optimize task performance (ST-SR problem).

As the community strives towards more autonomous multi-robot systems, the complexity of the tasks involved has increased considerably. In many cases, the tasks are too complex to be performed by a single robot alone, i.e. tasks must be allocated to a team of robots. This problem is considerably harder than the ST-SR problem and is commonly referred to as the Single-Task Multiple-Robot (ST-MR) problem (Gerkey and Matarić, 2003).

The **ST-SR** problems have been studied and numerous high quality solutions have been proposed (Parker, 1998; Gerkey and Matarić, 2000; Botelho and Alami, 1999; Werger and Matarić, 2000). However, the single-task multi-robot (**ST-MR**) problem has potentially significant applications and thus far has received relatively little attention. We believe that with advances in multi-robot coordination and cooperation algorithms and improved sensing capabilities, this somewhat neglected problem will assume greater significance.

Gerkey and Matarić (2003) formulate the task allocation problem as an instance of the Optimal Assignment Problem (OAP) and provide a taxonomy for multi-robot task allocation problems. The same work draws attention to the following limitation of their framework:

> ”Perhaps the most constraining aspect of our OAP framework is the assumption that we are working with single-robot tasks. In seeking to relax this assumption, we inevitably face a problem known in the multi-agent community as *coalition formation*. In its most general form, the problem of coalition formation is intractable. To optimally solve this problem for an arbitrary set of tasks, one must search the combinatorial space of possible coalitions. This search is unlikely to be practical for even moderately sized static coalition formation problems, and the situation is worse for Multi Robot Task Allocation **(MRTA)** domains, in which the coalition structures must be dynamic in order to respond to changing task requirements. Some heuristics to the coalition formation problem for multi-agent systems have been proposed (e.g. Sandholm and Lesser 1997;Shehory and Krauss 1998), but they have not been demonstrated in robotic domains.”

Thus dealing with multi-robot (**MR**) tasks still remains an open problem in the robotics community and this problem is the central theme of this dissertation. The solution to this problem lies in the formation of multi-robot teams or coalitions. The optimal solution to the coalition formation problem is unfortunately NP-hard. Fortunately problems very close to the coalition formation problem have been extensively studied, (e.g. Set Partitioning and Set Covering problems) and many heuristics for approximate solutions have been devised (Balas and Padberg, 1976; Chu and Beasley, 1996; Fisher and Kedia, 1990; Hoffman and Padberg, 1993). Game theorists and economists have also studied the coalition formation problem with regard to market based selfish agents. They have investigated various types of equilibria that lead to the formation of stable coalitions amongst selfish agents. In fact, coalition theory is now considered a field in its own right.

Distributed Artificial Intelligence (DAI) researchers have built upon the work in game theory and theoretical computer science to produce practical solutions to the multi-agent coalition formation problem. There has been considerable progress in the DAI literature in the area of multi-agent coalition formation algorithms. Despite this progress and the numerous coalition formation algorithms that have been proposed; to the best of our knowledge none of these algorithms have been demonstrated in the multi-robot domain. The reason for this is that multi-robot systems, unlike software agents, must address real world constraints. Thus there exists a divide between the multi-agent coalition formation literature and its application to the multi-robot domain. Our work aims to bridge this divide.

There are two ways to approach the problem; the first is to view the problem at a high level where the robots deliberately cooperate in an effort to increase the overall utility. Distributed Problem Solving (DPS) is used to model these types of task environments and solutions are usually a distributed implementation of an algorithm. The second manner in which to view the problem is at the agent (robot) level, where robots are modeled as selfish agents that attempt to increase their individual utilities. Thus in these task environments, the distribution of payoffs to individual agents is important. The agents are required to follow protocols for auctions or negotiations and the environment is modeled as an economy. These types of task environments are commonly called Multi-Agent System (MAS) environments.

The problem becomes more complicated if tasks are dynamically introduced into the system and the robots must reconfigure to execute the new tasks in real time. In this dissertation we adapt and extend the coalition formation techniques present in the DAI literature to facilitate their use in the multi-robot domain. The aim is to develop coalition formation techniques for the formation of multi-robot teams in different task environments.

Despite the similarities between multi-agent and multi-robot systems, the transition from agents to robots is not straightforward. DAI researchers make numerous assumptions while designing algorithms that do not hold when those algorithms are applied to robots. Besides these assumptions, robots must handle real world sensory noise, full or partial robot failures, and communication latency or loss of communications. All of these issues must be addressed before a multi-agent algorithm may be considered viable for robotic applications.

In this dissertation we address these issues and suggest modifications to current multi-agent coalition formation algorithms. We then incorporate our modifications into a chosen multi-agent coalition formation algorithm in order to facilitate its usage in the multi-robot domain. The objective is to develop a generic framework that tailors multi-agent algorithms to the multi-robot domain.

Another contribution of this work is the concept of Coalition Imbalance and its implications with respect to both fault tolerance and task performance. An empirical study of the impact of balance on the performance of multi-robot soccer and foraging teams is conducted, and the results suggest that imbalance information may be utilized to improve overall team performance.

Market based task allocation techniques have gained popularity over the past five years due to their inherently distributed protocols. Most of these auction-based systems draw inspiration from the contract-net protocol (Smith, 1980). This dissertation introduces RACHNA (Vig and Adams, 2006a), which is a novel, market based coalition formation system that leverages sensor redundancy to enable a more tractable formulation of the coalition formation problem. Current task allocation schemes tend to be somewhat task specific and are tightly coupled with the task domain. RACHNA employs a more generic utility based framework to accommodate different types of tasks and task environments. Preliminary experiments yield promising results demonstrating the system's superiority over simple task allocation techniques.

The overall research objective of this dissertation was to design a autonomous task allocation systems that were independent of the nature of the tasks. This was especially true of the RACHNA system which was designed for the urban search and rescue domain, a domain for which task definitions are still somewhat vague as the field is still in its infancy. As such, due consideration was given to ensure that the proposed systems be as generic as possible. This would allow for the system to function across a wide variety of tasks and task environments.

CHAPTER II

LITERATURE REVIEW

Research in Multi-Robot systems is a highly interdisciplinary field, sharing common ground with research in fields such as Game Theory, Linear Programming, Psychology, Distributed Artificial Intelligence, Physics, Mathematics, and Biology. The volume of the material is so vast that it is impossible provide an overview of all of these areas in a few pages. The purpose of this chapter is to acquaint the reader with the areas that are directly relevant to this dissertation.

II.1 The Coalition Formation Problem

The Multi-Robot Coalition Problem is defined as follows:

Definition (Multi-Robot Coalition Formation (MRCF)): Given a collection of n robots R and m tasks T. Each robot is equipped with certain sensor and actuator capabilities. A coalition is defined as a collection of multiple robots that combine to form a team. Also given is a characteristic function $f_c : C, T \mapsto \Re$ that maps coalition-task pairs to numerical values or efficiency ratings. The goal is to find the optimal partitioning of the set of robots into teams such that the subsequent assignment of the teams to tasks results in maximization of the overall performance or utility.

This above definition of MRCF is somewhat constrained. For example, most task allocation problems are not static, they are dynamic decision problems that vary in time with environmental changes and robot failures. Also a task may be performed in many different ways, i.e. a task may have multiple possible decompositions and hence multiple potential allocations. These questions are addressed with the RACHNA system described in Chapter VI.

II.2 Parallel problems

The MRCF is a very difficult problem that belongs to the complexity class of *strongly NP-hard*[1] problems. However, coalition formation shares a similar structure with a number of commonly studied problems in theoretical computer science. This section identifies these problems and examines them from a coalition formation perspective.

II.2.1 Winner Determination in Combinatorial Auctions

Combinatorial auctions are auctions in which bidders can place bids on combinations of items, called 'packages' rather than individual items. Formally, the problem is defined as follows (DeVries and Vohra, 2003):

Definition: Let N be a set of bidders and M the set of m distinct items. For every subset S of M let $b^j(S)$ be the bid that agent $j \in N$ has announced it is willing to pay for S. Let $b(S) = max_{j \in N} b^j(S)$. Then the winner determination problem can be formulated as:

$$max \sum_{S \subset M} b(S) x_S \tag{II.1}$$

$$\text{s.t. } \forall i \in M \sum_{S \ni i} x_S \leq 1 \tag{II.2}$$

$$\forall i \in M, x_S = 0, 1 \tag{II.3}$$

where x_S is a binary variable whose value depends on whether or not $b(S)$ is selected in the final list of bids. The MRCF problem can be cast as a combinatorial auction with the bidders being represented by the tasks, the items by the different robots, and the bids by the utility that each task has to offer for a particular subset of the set of robots (items). Unfortunately, the problem is inapproximable (Sandholm, 2002), however some empirically

[1]The complexity class of decision problems that are still NP-hard even when all numbers in the input are bounded by some polynomial in the length of the input.

strong algorithms do exist. Leyton-Brown et al. (2000) present a heuristic based algorithm to efficiently search the space of bids by utilizing a demand based ordering of the different items. Sandholm (2002) provides an algorithm that allows auctions to scale up to significantly larger numbers of items and bids by leveraging the fact that the space of bids is sparsely populated in practice. It remains to be seen if such algorithms can be sufficiently decentralized to apply them beneficially to a multi-robot setting. A generalized version of the winner determination problem in combinatorial auctions was utilized in the conception of the RACHNA system described in Chapter VI.

II.2.2 Optimization: Linear Programming

The coalition formation problem can also be cast as a 0-1 integer programming problem. Given a set of n agents and m candidate coalition-task pairs, the integer programming problem is cast as follows (Schrijver, 1986):
Given matrices A and U where:

$$U_j = \text{The utility gained when the } j^{th} \text{ coalition-task pair is selected.} \tag{II.4}$$

$$a_{ij} = \begin{cases} 1 \text{ if agent } i \text{ is a part of } j^{th} \text{ coalition-task pair.} \\ 0 \text{ otherwise.} \end{cases}$$

$$\text{Maximize} \quad \sum_{j=1}^{n} U_j x_j \tag{II.5}$$

$$\text{Subject to:} \ \sum_{j=1} a_{ij} x_j = 1, \ i = 1, ..., n \tag{II.6}$$

$$\text{where } x_j \in \{0,1\} \ \ j = 1, ..., m. \tag{II.7}$$

Dantzig (1972) introduced the simplex method for solving linear programming problems, which has since become the algorithm of choice for solving linear programs. Although the worst case complexity is exponential in the size of the input (Klee and Minty, 1972), the average case complexity for certain classes of problems is polynomial (Borgwardt, 1982), and the method is known to work very well in practice (Spielman and Teng, 2001). However variants of the simplex method appear to be heavily centralized. Consequently, these matrix based approaches appear to have limited potential for distributed applications and to the best of our knowledge, none have been successfully demonstrated in robotic domains.

II.2.3 Job Shop Scheduling Problems

Job Shop Scheduling (*JSS*) problems are characterized by (Garrido et al., 2000):

- A Job Set $J = \{j_1, j_2, \dots, j_n\}$.
- Machine set $M = \{m_1, m_2, \dots, m_m\}$.
- Operations $O = \{o_1, o_2, \dots, o_n\}$, $O_i = \{o_{i1}, o_{i2}, \dots, o_{im_i}\}$.
- Each operation has a processing time $\{\tau_{i1}, \tau_{i2}, \dots, \tau_{im_i}\}$ on a particular processor.
- On O define A, a binary relation representing a precedence between operations. If v has to be performed before w then $(v, w) \in A$.

The objective of Job Shop Scheduling is to find an optimal schedule such that the net processor time is minimized.

The MRCF problem can be cast as a relaxed instance of the *JSS* problem, with no constraints between jobs (independent job assumption or $A = \phi$). In other words, the order in which the jobs are performed is immaterial, all that matters is the mapping of operations to machines.

The incorporation of precedence constraints gives rise to a more difficult problem by introducing a scheduling component to the coalition formation problem (this problem belongs to the class of single-task multiple-robot extended-assignment problems). Coalition formation in task environments involving precedence ordered tasks are explored in Chapters III and IV of this dissertation.

Some scheduling problems allow for preemption, i.e. a job can be interrupted during execution, moved to a different machine, and then resumed. Since robots operate on real world tasks, robotic tasks cannot easily be traded amongst different robots. Therefore, the research community thus far has not given preemption much consideration for MRTA problems. It is our contention that preemption might be useful, if not necessary in certain dire circumstances such as a fire outbreak, chemical leakages etc. The RACHNA system described in Chapter VI allows for task preemption for urgent tasks.

Many solutions to the Job Shop Scheduling problem have been proposed in the literature. Some solutions view the problem as a constraint satisfaction search (Sadeh et al., 1995; Sadeh and Fox, 1996). More traditional approaches formulate the problem using integer programming (Wagner, 1959), mixed integer programming (Dyer and Wolsey, 1990) and dynamic programming (Srinivasan, 1971). It may be worthwhile to explore the possibility of decentralizing these algorithms without making significant compromises on solution quality.

II.2.4 Set Partitioning and Set Covering

Perhaps the closest problems in theoretical computer science to the coalition formation problem are the set partitioning and set covering problems. Determining which of the two problems is more apt for a particular domain depends on whether the task environment allows for overlapping coalitions. Overlapping coalitions are discussed further in Chapters III and IV.

Balas and Padberg (1976) define the Set Partitioning problem as follows:

Definition (Set Partitioning Problem (SPP)): Given a finite set E, a family F of acceptable subsets of E, and a utility function $u : F \rightarrow \Re_{+}$, find a maximum-utility family X of elements in F such that X is a partition of E.

The coalition formation problem can be cast as an instance of SPP, with E as the set of robots, F as the set of all feasible coalition task pairs, and u as the utility estimate for each such pair. The SPP problem has been studied in depth and is provably NP-hard. However numerous heuristic based SPP algorithms have been proposed in the literature (Chu and Beasley, 1996; Fisher and Kedia, 1990; Balas and Padberg, 1976), although it remains to be seen whether such heuristic algorithms are applicable to MRTA problems. To this end, a potentially important question is whether and how these algorithms can be parallelized. For environments that allow for overlapping coalitions, the set covering problem more closely approximates coalition formation. Balas and Padberg (1976) define the set covering problem as follows:

Definition (Set Covering Problem (SCP)): Given a finite set E, a family of acceptable subsets F of E and a cost function $c : F \rightarrow \Re_+$, find a maximum-utility family X of elements in F such that X is a cover of E.

The MT-MR-TA problem can be cast as an instance of the SCP, with E as the set of robots, F as the set of all feasible (and possibly overlapping) coalition-task pairs, and u as the utility estimate for each such pair. Like the SPP, the SCP problem is also NP hard (Korte and Vygen, 2000), however the coalition space of the SCP is far less constrained. Researchers have developed greedy approximation algorithms for the SCP that yield provably good sub-optimal solutions (Chvatal, 1979; Bar-Yehuda and Even, 1981).

It is important to note that these heuristic algorithms perform well when the space of feasible subsets is limited, and that they perform poorly in the most general case of the SCP, with all subsets allowed. For purposes of multi-robot task allocation, this result suggests that such algorithms would be best applied in environments in which the space of possible coalitions is naturally limited, as is the case with heterogeneous robots. To the best of our knowledge set covering or set partitioning algorithms have not been applied to MRTA problems, and the fact that Shehory and Kraus (1998) successfully adapted and distributed Chvatal's approximation algorithm (Chvatal, 1979) suggests that SCP may be viable for MRTA problems. Chapter III explores the applicability of Shehory and Kraus' algorithm to the MRCF problem.

II.2.5 Dynamic Programming

The complexity of the coalition formation problem generally decreases when the algorithm can observe individual coalition values instead of merely coalition structures. Sandholm et al. (1999) provide a dynamic programming algorithm for the coalition formation problem when individual coalitions can be evaluated. The algorithm runs in O(3^a) time (significantly less than exhaustive enumeration of all coalitions O(a^a)). However, this algorithm requires O(3^a) time even when the space of coalitions is restricted. This condition may not be feasible for task allocation, especially given that a coalition's value can only be evaluated when it is paired with a task.

The above list of problems is by no means exhaustive. Many other problems share a similar problem structure to coalition formation. The objective of listing the above problems was to direct the readers attention to the large number of potentially untapped solutions in theoretical computer science that may lead to fruitful algorithms for the coalition formation problem.

II.3 Game Theory

Game Theory as related to coalition formation, has primarily focused on stability and profit distribution, and not coalition formation algorithms. A thorough survey of coalition formation as studied in cooperative Game Theory is provided in Rappaport and Kahan (1984).

Work by Vohra (1995) and Rapoport (1970) describes which coalitions will form in n-person games and how the players will distribute the benefits of cooperation among themselves. Wu (1977) presented a transfer scheme that converges to the core (Definitions of terms in this section are provided in the Appendix) if the core is non-empty. An interesting property of Wu's model is that it implicitly allows passing from one coalition configuration to another and does not necessarily concentrate on a specific coalitional configuration. The transfer scheme leads to a point in the core but does not provide an algorithm to pass from one coalitional configuration to another.

However, in so far as allocation of tasks is concerned, the work by Vohra (1995) or Rapoport (1970) does not consider the specific constraints of a multi-agent environment, such as distribution, communication costs and limited computation time, etc. Also most robotic environments tend to be cooperative and hence payoff stability may be ignored in favor of better overall performance (utility).

The dynamic theory of bargaining sets was introduced by Stearns (1968). This work presented methods ("transfer schemes") for players in n-person games that enabled them, starting with an arbitrary payoff, to reach a payoff allocation within the kernel (see Appendix) or the bargaining set. The main deficiency of the dynamic theory of bargaining sets is that its dynamics deal solely with situations where the coalition configuration is given and payoffs are transferred. This theory is utilized to draw conclusions about the payoff stability of the RACHNA system described in Chapter VI.

Traditionally, game theorists have studied coalition formation from a perspective that is entirely focussed on predicting outcomes and payoff stability of competitive coalition games. Therefore, despite being a well studied problem, the motivations for research in coalition theory are not the same as they are in this dissertation, i.e. to design algorithms for obtaining optimal coalition structures. *Advances in game theory provide a good basis for this research; however they do not address the most relevant issues, i.e. the explicit protocols and strategies to be followed by the agents when forming coalitions.*

II.4 Multi-Agent Systems

Largely due to inherent differences between the respective systems under study, the multi-agent and multi-robot research communities have each developed their own methods for perception, reasoning, and action in individual agents/robots. In particular, the multi-robot community has historically studied both explicit and implicit coordination techniques (Balch and Arkin, 1994; Parker, 1995). Implicit coordination techniques employ dynamics of interaction among the robots and the environment in order to achieve the desired collective performance, often in the form of designed emergent behavior. Explicit coordination techniques, in comparison, deal with comparatively more sophisticated agents/robots, and employ intentional communication and collaboration methods much like those employed in multi-agent systems. Therefore at the level of explicit coordination among multiple individuals, the differences in techniques used by multi-agent and multi-robot systems are in fact very few (Gerkey and Matarić, 2004a). Although robotics researchers employ sophisticated techniques while designing single robot control systems, they have tended to use techniques that are already well known in the agent community when designing explicitly coordinated multi-robot systems (MRS). Having said that, there are irreconcilable differences between the two domains that are important to understand. This section aims to acquaint the reader with the concept of a software agent and the types of tasks that software agents and robots are commonly required to execute in an effort to highlight the differences in the two domains.

II.4.1 Software-Agents

A Software Agent is an artificial agent that operates in a software environment. Software environments include operating systems, computer applications, databases, networks, and virtual domains. Software agents differ from conventional software in that they are long-lived, semi-autonomous, proactive, and adaptive.

Wooldridge and Jennings (1995) define a software agent as "an agent that interacts with a software environment by issuing commands and interpreting the environment's feedback". Shoham (1997) defines a software agent as "a software entity, which functions continuously in a particular environment, often inhabited by other agents and processes". Shoham's definition is widely accepted in the research community.

A mobile agent is a software agent that can autonomously move between locations. This definition implies that a mobile agent is also characterized by the basic agent model. Green et al. (1997) point out that in addition to the basic model, any software agent defines a life-cycle model, a computational model, a security model, and a communication model. A mobile agent is additionally characterized by a navigation model. All issues referring to transporting an agent (with or without its state) between two computational entities residing in different locations are handled by the navigation model. The size of mobile agents depends on what they do. White and Pagurek (1998) indicate that in swarm intelligence, the agents are very small. On the other hand, configuration or diagnostic agents might get quite big, because they need to encode complex algorithms or reasoning engines. *Mobile agents can extend their capabilities on-the-fly, on-site by downloading required code off the network. They can carry only the minimum functionality, which can grow depending on the local environment and needs. This capability is facilitated by code mobility.* Many researchers utilize these properties while designing software agent coalition formation algorithms. These properties must be taken into account while translating these algorithms to robotic domains. Chapter III of this dissertation investigates these issues in greater depth.

II.4.2 Software Agent Tasks

Typically, an agent is given a very small and well-defined task. Although the theory behind agents has been around for some time, agents have become more prominent with the growth of the Internet. Information gathering is the most popular task for these agents. Many companies now sell software that enables one to configure an agent to search the Internet for certain types of information. These agents are called information agents. An information agent (Klusch and Shehory, 1996) is defined as an agent that has access to at least one, and potentially many information sources, and is “able to collate and manipulate information obtained from these sources to answer queries posed by users and other information agents.” Information agents can be further classified as:

1. *Learning Agents:* tailors to an individual’s preferences by learning from the user’s past behavior (Boicu et al., 2004).

2. *Shopping Agents:* compares the best price for an item (Doorenbos et al., 1993).

3. *Information Retrieval Agents:* helps the user to search for information in an intelligent fashion (Sycara and Zeng, 1996).

4. *Helper Agents:* performs tasks autonomously without human interaction (Khoo et al., 1998).

To date, agents have been successfully employed in multiple application endeavors like Data Filtering (Jennings and Higuchi, 1992), Pattern Recognition (Vuurpijl and Schomaker, 1998), Event Notification (Tripathi et al., 2004), Planning and Optimization (Fox et al., 2000), Rapid Response Implementation (Chen et al., 2001), and Discovery (Schramm et al., 1998). While this set of applications appears quite diverse, it should be noted that software agents, unlike robots, do not concern themselves with real world physical constraints. They operate exclusively in software environments that are far more predictable and less constrained.

II.4.3 Multi-Agent Cooperation

Distributed Artificial Intelligence (DAI) research is divided into two basic classes: Cooperative Distributed Problem Solving (CDPS) and Multi-Agent Systems (MAS). Research in CDPS considers how the efforts required for solving a particular problem can be distributed among a number of modules or nodes. Research in MAS is concerned with coordinating intelligent behavior among a collection of autonomous heterogeneous intelligent agents. Global control and globally consistent knowledge may not exist in MAS. MAS agents are assumed to be self-interested, attempt to achieve their own goals, and maximize their own personal payoff. Agents in CDPS systems have a common goal and attempt to maximize the system's global payoff. MAS designers only control their own agents. These classes are the two extreme poles in the DAI research spectrum.

Software agents are often required to form teams in order to perform tasks. DAI researchers have focussed on constructing agents that are able to cooperate during task execution, thus increasing either individual agent benefits or the systems benefits. Smith

(1980) discusses the Contract Net Protocol (CNP) in which the agents may attempt to satisfy a task by dividing it into sub-tasks and sub-contract each sub-task to another agent via a bidding mechanism. CNP allocates tasks to single agents and a procedure for task partitioning is necessary. Many CNP inspired MRTA systems have been proposed in the literature as mentioned in Chapter II, Section II.5.3.

Gasser (1993) focuses on the social aspects of agent knowledge and action in multi-agent systems. As in human societies, social mechanisms can dynamically emerge. This approach is effective when agents are interacting in environments where there is no agreed upon, well defined interaction mechanism, or in continuously evolving domains. This approach also bears a strong likeness to swarm based approaches to task allocation in that it allows a system to converge to appropriate robot/task ratios based on local robot interactions.

Environments with heterogeneous agents require that protocols must be agreed upon and enforced by the designers. The need for protocols increases further in coalition formation. As was noted by Shapely and Shubik (1973), "in situations where every cooperative demand by every coalition cannot be satisfied, some constraints must be placed on coalitional activity lest it be trapped in an endless loop of rejected suggestions for coalition formation." This conclusion gave rise to the concept of stability and various notions of equilibrium.

Sandholm and Lesser (1995) developed a coalition formation model for bounded rational agents and presented a general classification of coalition games. The value of a coalition in their model depends on the computation time; however all configurations and possible coalitions are considered when computing a stable solution.

Shehory and Kraus (1998) proposed algorithms for autonomous agents to form coalitions and distribute the joint payoffs to members in Non-Super-additive environments. The suggested protocol guarantees that if the agents follow it, a certain stability (kernel-stability) is met. The same paper presented an alternative protocol that offers a weaker

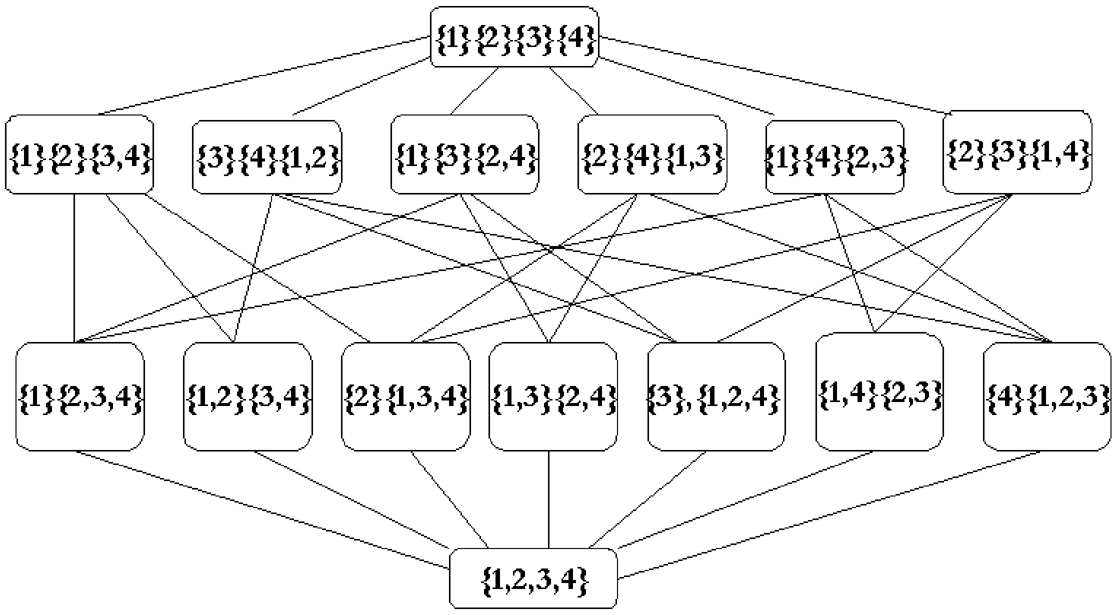


Figure II.1: Coalition Structure Graph (Sandholm et. al. 1999).

form of stability with polynomial running time. However, in both cases no bound from the optimal is guaranteed.

Sandholm et al. (1999) formally prove that finding the optimal coalition structure is an NP-Complete problem. They view the coalition structure generation process as a search in a coalition structure graph. A coalition structure graph for four agents is shown in Figure II.1. The problem is formulated as a search through a subset of the coalition structures and selection of the best coalition structure encountered.

Sandholm et al. (1999) go on to prove that in order to establish any sort of bound on the quality of the generated coalition structure, it is necessary and sufficient that the search algorithm must search the bottom two levels (or 2^{a-1} nodes) of the graph. The suggested algorithm involves an initial search through these two levels and then is followed by a time bounded breadth first search from the top of the coalition structure graph. Until recently this was the only anytime algorithm that could establish a worst case bound. Dang and

Jennings (2004) have designed an algorithm that improves on the algorithm provided by Sandholm et al. (1999) and analyzes fewer coalitions while establishing small bounds from the optimal.

An algorithm for task allocation via coalition formation in CDPS environments was proposed by Shehory and Kraus (1995). This algorithm yields solutions by limiting the size of the coalitions and uses a greedy heuristic to yield a coalition structure that is provably within a bound of the best possible solution given the limit on the number of agents. This algorithm is especially relevant to this dissertation and further modifications to this algorithm are described in Chapter III. Shehory and Kraus (1996b) extended this algorithm to enable formation of overlapping coalitions for precedence-ordered task-execution.

Shehory and Kraus (1996a) discuss cooperative goal satisfaction using a physics oriented model of multi-agent systems. The attractive feature of this model is that it requires no communication between agents. This approach strictly applies to CDPS environments and the applicability is limited to loosely coupled tasks.

Sycara (1995) presents a coalition formation algorithm for rational agents engaged in decentralized information gathering. The proposed algorithm enables cooperation via formation of kernel-oriented stable coalitions. During the coalition formation process each agent rationally decides to coalesce with other information agents by calculating its own utility on the set of discovered interdatabase dependencies. Based on such interrelational knowledge, each agent can only pose directed, intensional queries on local data to committed members of the same coalition.

Of late, coalition formation has begun to receive a great deal of attention from the DAI research community (Fass, 2004; Li and Soh, 2004; Sorbella et al., 2004; Abdallah and Lesser, 2004). Despite this plethora of agent coalition formation algorithms, none of these algorithms have been demonstrated to form coalitions in the multi-robot domain. There are numerous reasons for this, some prominent ones being the added communication costs in the multi-robot domain, the unreliability of both communication and robots in real world domains, and the non-transferability of resources between robots. This dissertation investigates the reasons behind this divide between the agent and robotic domains and offers solutions to some of the problems that arise while tailoring multi-agent coalition formation algorithms to the multi-robot domain. Transferability of these algorithms are further discussed in Chapter III.

II.5 Multi-Robot Systems

This section provides an overview of the relevant research in Multi-Robot Systems (MRS). Section II.5.1 identifies some common MRS tasks that are utilized as test-beds for MRS architectures. Section II.5.2 provides the various dimensions along which an MRS task or system can be classified. Section II.5.3 discusses some well known task allocation systems present in the literature. Section II.5.4 identifies some of the inherent differences between multi-agent and multi-robot systems.

II.5.1 Multi-Robot Tasks

This section provides an overview of the various test-beds that are commonly used in the multi-robot system literature to evaluate and validate the various coordination techniques. Farenelli et al. (2004) provide a survey of the various MRS tasks that researchers employ to validate their systems. The tasks mentioned include:

Foraging and Coverage: The Multi-Robot foraging task requires the robots to search the environment for objects of interest and to retrieve these objects to a specified destination. Foraging is a common MRS test-bed due to its application to real world tasks such as toxic waste clean up, demining, and service robotics. Typical applications are provided by Murphy et al. (2002) and Jung and Zelinsky (2000). Similarities between the foraging and coverage task were pointed out by Choset (2001). Coverage requires the robots to process all the points of free space in the environment. The central issue for coverage is to find effective techniques for cooperatively scanning the environment. Applications include: demining, snow removal, lawn mowing etc.

Multi-Target Observation: Also known as cooperative observation of multiple moving targets (CMOMMT), multi-target observation is a recent test-bed, first introduced by Parker (1999). The task involves the detection and tracking of a set of moving objects in the environment. The robots are required to maximize the time during which each moving target is under observation by at least one robot. Werger and Matarić (2000) use the CMOMMT test-bed to evaluate their Broadcast of Local Eligibility (BLE) task allocation system. The system describes a task allocation scheme for robots based on Port Arbitrated Behaviors. Multi-target observation has many connections with security and surveillance where targets moving around in a bounded area must be monitored and observed.

Box Pushing and Object Transportation: The box pushing task requires robots to push boxes from a starting location and orientation to a destination location and orientation. Important applications include construction, stockage, and truck loading and unloading. Boxes are generally assumed to be on a plane, although Simmons et al. (2000) focuses on lifting and carrying objects, thus substantially increasing the task complexity. Both Gerkey and Matarić (2002b) and Parker (1999) have studied the problem in depth and have conducted experiments in the box-pushing domain to validate their coordination architectures for tightly coupled tasks.

Exploration and Flocking: Exploration and flocking are regarded as two distinct tasks but both require coordination between the MRS members as they move through the environment. The goal during the flocking task is for the robotic agents is to move together, similar to a flock or herd. If the relative positions the robots are to maintain is required to be a certain shape, then the task is called a formation task. Cooperation among the robotic agents is also used to localize each other and to fuse information acquired from the environment. Map building of unknown environments is a common issue related to exploration. A more complex version of this task, as presented in Fenwick et al. (2002), is cooperative localization and mapping, in which the robots have to localize while moving and building a map of the environment.

Soccer: Kitano et al. (1997) outline how robotic soccer has become a standard test-bed for research in multi-agent and multi-robot cooperation. The uncertain dynamics and hostile environment in which the robots operate makes multi-robot coordination a very challenging problem. The different environments for each robotic league presents several MRS coordination issues. The middle-size league and the four-legged league require that all robot sensors be on-board. Therefore the robots are more autonomous and encounter high uncertainty levels when reconstructing global environmental information. The small-size league provides the robotic agents with an overhead camera view of the field, therefore coordination approaches in this league are centralized.

The above list is by no means an exhaustive list of MRS application domains. With new sensor technologies constantly being developed, new innovative MRS applications continue to surface. A new task that has recently begun to receive attention is the pursuit evasion task where robots are required to clear an area of mobile contaminants (Gerkey et al., 2005).

II.5.2 Taxonomies of Multi-Robot Systems

The previous section provided a description of some common multi-robot tasks. As researchers attempted to provide solutions for these different tasks, it was apparent that a classification of these tasks was needed along various dimensions, such as communication bandwidth requirements, platform capability requirement, robot to task ratio, etc. Various taxonomies were proposed to classify tasks along these dimensions.

Balch (2002) provides a taxonomy of multi-robot tasks with an eye towards incorporating reinforcement learning in the multi-robot task domain. This work classified tasks on the basis of time for evaluation, action performed, resource limitations, movement and platform capabilities. Dudek et al. (2002) provide a taxonomy for characterizing a robot collective. The taxonomic axes included in this work were the Collective Size, Communication Range, Communication Topology, Communication Bandwidth, Collective Reconfigurability, Processing Speed, and Collective Composition. Farenelli et al. (2004) further proposed a taxonomy of multirobot systems while focussing on the level of coordination in the system. The coordination dimensions in this taxonomy include the level of cooperation in the system, the level of awareness or knowledge of the robots about each other, the mechanisms used for coordination (weak, strong), and the organization of the system (distributed, centralized). This work also groups systems along dimensions of communication (direct, indirect), team size, architecture (deliberative, reactive), and composition (heterogeneous, homogeneous).

However, the taxonomy that is most relevant to this dissertation is the one provided by Gerkey and Matarić (2004b). This work provides a taxonomy of Multi-Robot Task Allocation (MRTA) problems based on the following three axes:

- **Single-task robots (ST) vs. multi-task robots (MT):** ST means that each robot is capable of executing at most one task at a time, while MT means that some robots can execute multiple tasks simultaneously.

- **Single-robot tasks (SR) vs. multi-robot tasks (MR):** SR means that each task requires exactly one robot to achieve it, while MR means that some tasks can require multi-robots.

- **Instantaneous assignment (IA) vs. time-extended assignment (TA):** IA means that the available information concerning the robots, the tasks, and the environment permits only an instantaneous allocation of tasks to robots, with no planning for future allocations. TA means that more information is available, such as the set of tasks that will need to be assigned.

An MRTA problem is denoted by a triple of two letter abbreviations drawn from the three dimensions. For example a problem in which multi-robot tasks must be allocated once to single-task robots is designated **ST-MR-IA**. Robotic researchers have primarily focussed their attention on Single-Task Single-Robot **(ST-SR)** problems, consequently the **ST-MR-IA** and **ST-MR-TA** problems have not previously received much attention in the literature. The **ST-MR-IA** problem is also called the multi-robot coalition formation problem and is the primary focus of this dissertation. Novel solutions for the multi-robot coalition formation problem are provided in Chapters III and VI of the dissertation.

II.5.3 Task Allocation

Task allocation is proving to be a challenging problem due to the unpredictable nature of robot environments, sensor failure, robot failure, and dynamically changing task requirements. A number of elegant solutions to the task allocation problem have been proposed. The ALLIANCE (Parker, 1998) architecture uses motivational behaviors to monitor task progress and dynamically reallocate tasks. ALLIANCE employs a variant of the subsumption architecture (Brooks, 1986) and makes use of ”behavior sets” to enable a robot to perform versatile tasks. Recently Low et al. (2004) proposed a swarm based approach for the cooperative observation of multiple moving targets (CMOMMT). This scheme mimics ant-behavior to regulate the distribution of sensors in proportion to that of the mobile targets. Dahl et al. (2003) present a task allocation scheme based on “Vacancy Chains,” a social structure modeled on the creation and filling up of vacancies in an organization. The Broadcast of Local Eligibility system (BLE) (Werger and Matarić, 2000) system uses a Publish/Subscribe method to allocate tasks that are hierarchically distributed.

Market based task allocation systems have traditionally found favor with the software-agent research community. The inspiration for these systems stems from the Contract Net protocol (Smith, 1980). Variations of the Contract Net protocol have found applications in numerous software-agent negotiation scenarios (Sandholm, 1993; Collins et al., 1997; Sandholm and Lesser, 1996; Sycara and Zeng, 1996). Stentz and Dias (1999) were the first to utilize a market-based scheme to coordinate multiple robots for cooperative task completion. This work introduced the methodology of applying market mechanisms to intra-team robot coordination as opposed to competitive inter-agent interactions in domains such as E-commerce. Laengle et al. (1998) implemented the KAMARA system that uses a negotiation based task allocation scheme for controlling the different components of a complex robot.

Caloud et al. (1990) developed the GOPHER architecture that utilizes a centralized auction protocol to allocate tasks with a high level of commitment. Gerkey and Matarić (2000) developed MURDOCH; a completely distributed auction-based task allocation scheme that utilized a Publish/Subscribe communication model. Tasks in MURDOCH are allocated via a single round, first price auction in a greedy fashion. M+, another auction based task allocation protocol was developed by Botelho and Alami (1999). The novelty of the M+ system lies in that it allows for dynamic task reallocation of subcomponents of complex tasks. Dias (2004) designed the Traderbots architecture for multirobot control. Traderbots agents called traders are responsible for trading tasks via auctions. When an auction is announced agents compute bids based on their expected profit for the tasks, and the robots that can perform the tasks for the lowest price are awarded contracts.

The common underlying factor in all of the above systems is the single robot-single task assumption that assumes the indivisibility of tasks and that each task may be performed by a single robot. As research in the field matures and multi-robot tasks become more and more complex, this assumption is proving to be an oversimplification. Many task domains require that a team of robots work on a task simultaneously, making the task allocation problem far more difficult.

Thus, a relatively unexplored problem in multi-robot systems is the allocation of multi-robot teams to different tasks (the ST-MR problem) commonly known as the *Multi-Robot Coalition Formation (MRCF)* problem. Recently researchers have offered a variety of market based solutions to the (ST-MR) single-task multiple-robot task allocation problem (Zlot and Stentz, 2005; Lin and Zheng, 2005; Schneider et al., 2005; Tang and Parker, 2005a). However, none of these task allocation schemes utilize the inherent redundancy in robot sensory capabilities. The RACHNA system described in Chapter VI leverages this redundancy to enable a more tractable formulation of the MRCF. Thus, one of the primary objectives of this dissertation is to develop generic task allocation schemes for allocating complex multi-robot tasks.

II.5.4 Software Agent Tasks vs. Multi-Robot Tasks

The previous sections highlight how the tasks assigned to robotic agents can vary significantly from tasks in software domains. Robotic agents have to operate in the real world and have to account for issues of interference between robots, obstacle avoidance, etc. Robotic agents also often deal with more restricted resource constraints and are unable to extend their capabilities on the fly. A software agent does not have to worry about losing communication capabilities but MRS have to deal with more restricted and delayed communication. Failures occur with higher frequency and in a wider variety in robotic systems. Additionally, robotic capabilities do not vary dynamically whereas mobile software agents often extend their capabilities by importing relevant code from another agent. Furthermore, robotic systems have to be able to accommodate larger error bounds in performance since they often deal with faulty sensors and interact with real world environments. Finally, robotic systems often require more creative solutions to recover from faults. Thus, controlling multiple robots can be a significantly different problem compared to controlling multiple agents.

II.6 Teamwork

It is well established in the Human Factors and Teaming theory that a team is more than a collection of individuals and teamwork is more than the aggregate of their individual behaviors. The dynamics of the interactions between individual team members has significant bearing on the overall performance of a team (Bass, 1980). Understanding effective teamwork performance entails understanding how groups of individuals function to produce effective synchronized output, rather than just summed or aggregate responses (Steiner, 1972; Hackman, 1983; Nivea et al., 1978; Fleishman and Zaccaro, 1992).

Initial attempts at studying team processes focussed largely on military teams and team processes that enabled them to function more effectively under extreme time pressure, stress, and circumstances. Over the years, the focus has shifted to team failures, particularly those tied with high visibility (for example aircraft and military accidents) (Ilgen, 1999). Summarizing, research over the last 50 years has produced many theories, most of which incorporate a general input-process-output approach, whereby certain variables are fed into the system, followed by the execution of team processes, and finally the recording the team performance results (Ilgen, 1999).

Besides developing team theories and models, researchers have struggled to identify those critical skills that enable teams to coordinate and synchronize effectively to fulfill their goals and ambitions. Early research focussed on orientation, resource distribution, timing, motivation, and team morale (Nivea et al., 1978; Ruffell-Smith, 1979). Slowly the focus shifted to self efficacy (Bandura, 1986), implicit and explicit coordination activities (Kleinman and Serfaty, 1989), and providing motivational reinforcement (Oser et al., 1999). More recent work outlines the most vital components to team performance as mutual performance monitoring (Hackman, 1983), collective orientation, adapting to novel and unpredictable situations, and flexibility (Prince and Salas, 1993).

Due to the inherent differences between the objectives and operation of human and robot teams, it is not trivial to map results from one domain to another. Besides having to be selective about which results to map, the results generally have to be formalized to a higher degree before application to robotic teams. One successful example where researchers have exploited the teamwork theories is the TEAMCORE project (Pynadath and Tambe, 2002) which utilizes the Belief, Desire, Intention (BDI) framework (Cohen and Levesque, 1991) to devise a theoretical foundation for multi-agent teaming.

Robotic domains offer the advantage of allowing numerous experiments to be conducted under similar conditions. This allows for the acquisition of consistent, reliable data that can be utilized for stronger validations of teamwork theories. Results from robotic team experiments may also be mapped back to the human teams (with appropriate considerations). Chapter V of this dissertation examines the impact of the balance parameter on the performance of a multi-robot team. The results offer interesting insight into how variance in individual contributions affects overall team performance.

Theories of team performance abound in the literature and our understanding of which variables affect team performance has greatly improved over the years. However, at this point the field needs deeper, better specified, and validated models of team performance to more precisely target the key areas for improvement.

CHAPTER III

HEURISTIC BASED COALITION FORMATION

In this Chapter a well-known multiple-agent coalition formation algorithm is investigated, modified and extended to the multiple-robot domain. The algorithm by Shehory and Kraus (1998) is designed for task allocation via coalition formation in Distributed Problem Solving (DPS) environments. This algorithm utilizes a heuristic to constrain the space of coalitions considered, thereby enabling a more tractable formulation of the coalition formation problem. The suggested heuristic fits well within the multiple robot framework and is designed for distributed environments (Vig and Adams, 2006b). A description of the algorithm is provided followed by the modifications and extensions required to facilitate application of the algorithm to the multiple-robot domain.

III.1 Shehory and Kraus' Coalition Formation Algorithm

The algorithm by Shehory and Kraus (1998) is designed for task allocation via software agent coalition formation in DPS environments. The heuristic utilized in this algorithm constrains the space of coalition structures under consideration by limiting the maximum possible size of a coalition within a coalition structure. It is often the case in multi-robot domains that one can safely place a limit on the number of robots required for any particular task prior to allocation. The heuristic therefore fits well with the multi-robot domain. This Section highlights some of the assumptions that the algorithm makes and provides a detailed description of the algorithm.

III.1.1 Assumptions

Assume a set of n agents, $A = A_1, A_2 ..., A_n$. The agents communicate with each other and are aware of all tasks to be performed. Each agent A_i has a p-dimensional vector of real non-negative capabilities, $B_{A_i} = < b_1^{A_i}, b_2^{A_i} ..., b_p^{A_i} >$, where each capability is a property that quantifies the ability to perform an action. An evaluation function is attached to each capability type that transforms capability units into monetary units. It is assumed that there is a set of m independent tasks, $TS = t_1, t_2, ..., t_m$. Each task t_l has a p-dimensional capability requirement vector $B_{t_l} =< b_1^{t_l}, ..., b_p^{t_l} >$. The utility gained from performing the task (or *taskvalue*) depends on the capabilities required for execution. A coalition is a group of agents that decide to cooperate to perform a common task and each coalition performs a single task. A coalition C has a p-dimensional capability vector B_c representing the sum of the capabilities that the coalition members contribute to this specific coalition. A coalition C can perform a task t_l only if t_l's capability requirement vector B_{t_l} satisfies $\forall\, 0 \leq u \leq p,\ b_u^{t_l} < b_u^C$.

III.1.2 Shehory and Kraus' Algorithm

Shehory and Kraus' algorithm consists of two primary stages:

1. Calculate the coalitional values for comparison.

2. Determine, via an iterative greedy process, the preferred coalitions and form them.

Stage one is more relevant to this work. During this stage the evaluation of coalitions is distributed amongst the agents via extensive message passing. After this stage, each agent has a list of coalitions for which it calculated coalition values. It also has all necessary information regarding the capability member requirements for each coalition-task pair. In order to calculate the coalition values, each agent proceeds to:

1. Determine the necessary capabilities for each task execution $t_i \in T$, by comparing the required capabilities to the coalition capabilities.

2. Calculate the best-expected task outcome of each coalition and choose the coalition yielding the best outcome.

Distributed calculation of coalition values: Each agent will perform the following steps in order to decide which coalitions to evaluate:

1. Calculate all of the possible coalitions, up to size k in which you are a member and form a personal list of coalitions.

2. For each coalition in the personal list, contact each member and ask for its task-performing capabilities.

3. Inform the agent whom you have approached that you are committed to the calculation of the coalitional values of the coalitions in which you are both members.

4. Construct a personal list of agents that you have approached and avoid repeated approaches to the same agents.

5. In case you were approached by another agent and it had committed to the calculation of the values of the common coalitions, erase all of your common coalitions from your personal list of coalitions.

6. Repeat the contacting of other agents until you have none to approach.

At this stage, each agent has a list of coalitions for which it had committed to calculate the values. It also has all of the necessary information about the capabilities of the members of these coalitions. Now in order to calculate these values, each agent shall perform the following steps:

1. Check which capabilities are necessary for the execution of each task $t_i \in T$. Compare them to the capabilities of the members of the coalition, thus determining the tasks that can be performed by the coalition.

2. Calculate the expected outcome of the tasks that can be performed by the coalition. For each task, perform the following: First, calculate the monetary values of the tasks capability requirements and sum them. Then calculate the monetary values of the capabilities of the coalitions which are not used for the fulfillment of the task and sum them. Subtract the second sum from the first. This value is the expected outcome of the task.

3. Among all of the expected outcomes, choose the maximal one. This will be the coalitional value, V_c.

The protocol ensures that no coalitions are lost, although it does not preclude some redundancy in the coalition lists of each agent. For more details the reader is referred to Shehory and Kraus (1998).

Choosing coalitions: The second stage of the algorithm involves the selection of the preferred coalitions and the gradual achievement of the coalitional configuration. At the end of the first stage of the algorithm each agent will have calculated a list of coalitions and their values. Each agent will choose the best coalition from its list, i.e., the coalition C_i that has the largest value w_i. Next, each agent will announce the coalitional value it has chosen, and the highest among these will be chosen by all agents. The members of the coalition that was chosen will be deleted from the list of candidate members for new coalitions. In addition, any possible coalitions from the coalition list of any agent that includes deleted agents, will be deleted from its list. The calculation of coalitional values and selection of the preferred coalitions will be repeated until all agents are deleted, or until there are no more tasks to be allocated, or none of the possible coalitions is beneficial. The coalitional values will be calculated repeatedly since they are affected by the coalitional configuration. This is because each value is calculated subject to the tasks that should be performed. Any change in the coalitional configuration means that a task was assigned to a coalition, so this specific task no longer affects the coalitional values that may previously have been affected by it. Therefore, the coalitional values that have been calculated with reference to a task that has just been allocated must be re-calculated. All other values remain unchanged.

III.2 Issues in Multi-Robot Coalition Formation

Shehory and Kraus' algorithm (Section III.1) yields results that are close to optimal, and utilizes a heuristic that can be easily applied to multi-robot domains, especially where limits can be imposed on the size of the multi-robot team. However, the presented algorithm cannot be directly applied to multi-robot coalition formation. This section identifies issues that must be addressed when the algorithm is applied to the multi-robot domain.

III.2.1 Computation vs. Communication

The algorithm by Shehory and Kraus (1998) requires extensive communication and synchronization during the computation of coalition values. While this may be inexpensive for disembodied agents, it is often desirable to minimize communication in multi-robot domains, even at the expense of extra computation. The modified algorithm presented in this chapter requires that each agent assume responsibility for evaluating all coalitions in which it is a member, thereby eliminating the need for communication. An added assumption is that a robot has a priori knowledge of all robots and their capabilities (Shehory and Kraus, 1998). Robot capabilities do not typically change, therefore this is not a problem unless a partial or total robot failure is encountered (Ulam and Arkin, 2004). It is necessary to analyze how each robot's computational load is affected. The total space of examined coalitions includes all coalitions of sizes less than or equal to the maximum allowed coalition size (k). Suppose there are n identical robots with a perfect computational load distribution, then the number of coalitions each robot must evaluate with communication is:

$$\eta_{with} = \sum_{w=0}^{k} \binom{n}{w} / n. \tag{III.1}$$

It is unlikely that the load will be perfectly distributed, rather some agents will complete their computations before others and remain idle until all computations are completed. The worst case communicational load per agent is O(n^{k-1}) during the calculation-distribution stage. Alternatively, if each agent is responsible for only computing coalitions in which it is a member, then the number of coalitions evaluated with no communication becomes:

$$\eta_{without} = \sum_{w=0}^{k-1} \binom{n-1}{w}. \tag{III.2}$$

Equation (III.2) represents the number of coalitions of size $\leq k$ in which a particular agent A_i is always a member. Equation (III.1) requires fewer computations than Equation (III.2) but this is not an order of magnitude difference. The agents' computational load is O(n^k) per task in both cases. The communication load per robot is O(1) in the calculation-distribution stage. The additional computation may be compensated for by reduced communication time. Experiments described in Chapter IV, Section IV.1 demonstrated a significant decrease in execution time when communication is removed from the first stage of the algorithm.

A desirable side effect of this modification is additional fault tolerance (Ulam and Arkin, 2004). If a robot R_A fails during coalition list evaluation, information relevant to coalitions containing R_A is lost. Since coalitions involving R_A cannot be formed post failure, this information is no longer necessary. Thus a robot failure does not require information retrieval from the failed robot. However, the other robots must be aware of the failure so that they can delete all coalitions containing the failed robot R_A.

III.2.2 Task Format

Current multi-agent coalition formation algorithms assume that the agents have a capability vector, $< b_1^{A_i}, ..., b_r^{A_i} >$. Multi-robot capabilities include sensors (e.g. laser range finder or ultrasonic sonars) and actuators (e.g. wheels or gripper). Shehory and Kraus' algorithm assumes that the individual agent resources are collectively available upon coalition formation and that the formed coalition can freely redistribute resources among the software agents. However, this is not possible in a multi-robot domain, as robots cannot autonomously exchange capabilities.

Correct resource distribution is also an issue. The box-pushing task (Gerkey and Matarić, 2002a) is used to illustrate this point. Three robots, two pushers (with one bumper and one camera each) and one watcher (with one laser range finder and one camera) cooperate to complete the task. The total resource requirements are: two bumpers, three cameras, and

Table III.1: Box-pushing task TAM.

	Bumper$_1$	Bumper$_2$	Camera$_1$	Camera$_2$	Camera$_3$	Laser$_1$
Bumper$_1$	X	0	1	0	0	0
Bumper$_2$	0	X	0	1	0	0
Camera$_1$	1	0	X	0	0	0
Camera$_2$	0	1	0	X	0	0
Camera$_3$	0	0	0	0	X	1
Laser$_1$	0	0	0	0	1	X

one laser range finder. However, this information is incomplete, as it does not represent the constraints related to sensor locations. Correct task execution requires that the laser range finder and camera reside on a single robot while the bumper and laser range finder reside on different robots. This implies that a multi-robot coalition that simply possesses the necessary resources is not necessarily capable of performing a task, the capability locational constraints have to be represented and met.

Initially, we proposed a matrix-based constraint representation for the multiple-robot domain in order to resolve the problem. The task is represented via a capability matrix called a Task Allocation Matrix (TAM). Each matrix entry corresponds to a capability pair (for example [sonar, laser]). A 1 in an entry indicates that the capability pair must reside on the same robot while a 0 indicates that the pair must reside on separate robots. Finally an X indicates a do not care condition and the pair may or may not reside on the same robot. Every coalition must be consistent with the TAM if it is to be evaluated as a candidate coalition. The box-pushing TAM is provided in Table III.1. The entry (Laser$_1$, Camera$_3$) is marked 1, indicating that a laser and a camera must reside on the same robot. Similarly the (Bumper$_1$, Laser$_1$) entry is marked 0 indicating the two sensors must reside on different robots.

Unfortunately utilizing the TAM matrix to verify the locational constraints on the individual sensors and actuators in a coalition is computationally inefficient. The constraints on sensor locations can alternatively be represented as a Constraint Satisfaction Problem

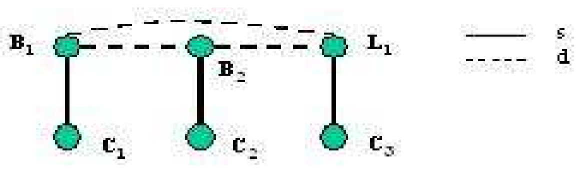


Figure III.1: Box-pushing task constraint graph Vig and Adams (2005).

(CSP). The CSP variables are the required sensors and actuators for the task. The domain values for each variable are the available robots possessing the required sensor and actuator capabilities. Two types of constraints exist, the sensors and actuators must reside on the same robot or on different robots. A constraint graph evolves with locational constraints represented as arcs labeled *s* (same robot) or *d* (different robot).

Fig. III.1 provides the box-pushing task constraint graph. This task's resource constraints between Bumper$_1$ and Bumper$_2$ (labeled B_1 and B_2) are implied by their locational constraints. Since Bumper$_1$ and Bumper$_2$ must be assigned to different robots, there cannot be a solution where a robot with one bumper is assigned to both Bumper$_1$ and Bumper$_2$.

The domain values for each variable in Fig. III.1 are the robots that possess the capability represented by the variable. A coalition can be verified to satisfy the constraints by applying arc-consistency. If a sensor has an empty domain value set, then the current assignment fails and the current coalition is deemed infeasible. A successful assignment indicates the sub-task to which each robot was assigned.

Using arc-consistency, each candidate coalition is checked against the constraint graph to verify if its coalition is feasible. A caveat is that arc-consistency does not detect every possible inconsistency (an NP-complete problem). This limitation may be overcome by solving the CSP for the best coalition selected. If no solution exists (i.e. false positive), then the next best coalition is chosen. Solving the CSP also automatically assigns each robot to the appropriate subtask.

Experiments measuring the additional overhead imposed by the CSP formulation are presented in Chapter IV, Section IV.2. The experiments demonstrate that the effect of the CSP formulation on the execution time was on the order of milliseconds even for hundreds of agents.

III.2.3 Coalition Imbalance

The **Coalition imbalance** or lopsidedness is defined as the degree of unevenness of resource contributions made by individual members to the coalition. This characteristic is not considered in other coalition formation algorithms. A coalition in which one or more agents have a predominant share of the capabilities may have the same utility as a coalition with evenly distributed capabilities. Robots are unable to redistribute their resources, therefore coalitions with one or more dominating members (resource contributors) tend to be heavily dependent on those members for task execution. These dominating members then become indispensable. Such coalitions should be avoided in order to improve fault tolerance as over-reliance on dominating members can cause task execution to fail or considerably degrade. If robot R_A is not a dominating member (does not possess many sensors) then it is more likely that another robot with similar capabilities can replace robot R_A.

Rejecting lopsided coalitions in favor of balanced ones is not entirely straightforward. When comparing coalitions of different sizes, a subtle trade-off between lopsidedness and the coalition size can arise. The argument may be made both for fault tolerance and for smaller coalition size. Coalitions with as few robots as possible may be desirable. Conversely, there may be a large number of robots thus placing the priority on fault tolerance and balanced coalitions.

There are some desirable properties for a metric quantifying coalition imbalance. Consider a coalition C with a resource distribution $(r_1, r_2, ..., r_n)$ (i.e. coalition member 1 contributes net resources r_1, member 2 contributes net resources r_2, etc.). The chosen balance function should be continuous in r_i, also any change towards a more equable distribution of $r_1, r_2 \ldots r_n$ should increase the value of the metric. Considering these properties, we introduced the **Balance Coefficient** (*BC*) to quantify the coalition imbalance level. For coalition

C, the BC with respect to a particular task can be calculated as follows:

$$BC = \frac{r_1 \times r_2 \times \ldots r_n}{\lceil \frac{taskvalue}{n} \rceil^n}. \tag{III.3}$$

BC measures the deviation from the perfectly balanced coaltion where each member contributes equally $(taskvalue/n)$ to the task. Clearly, the BC is continuous in r_i.

Result: The higher the BC, the more balanced the coalition.

Proof: Consider any coalition of size n with resource distribution $(r_1, r_2, ..., r_n)$ and assume any task t_l with $taskvalue\ T$. Further, consider an integer $s < n$ such that:

$$r_i = T/n + \alpha_i, for\ i = 1\ to\ s$$
$$r_i = T/n - \delta_i, for\ i = 1\ to\ n - s$$

where, $\alpha_i \geq 0,\ \delta_i \geq 0, \forall i$.

The BC for this coalition is:

$$\begin{aligned} \gamma BC_1 &= (\frac{T}{n} + \alpha_1)(\frac{T}{n} + \alpha_2)\cdots(\frac{T}{n} + \alpha_s) \times \\ & (\frac{T}{n} - \delta_1)(\frac{T}{n} - \delta_2)\cdots(\frac{T}{n} - \delta_{n-s}). \end{aligned} \tag{III.4}$$

where, $\gamma = (\frac{T}{n})^n$.

Adding a factor ε to one of the αs and an equal amount μ to all δs (or vice versa) makes the coalition more imbalanced (and should decrease the BC). Thus, the BC becomes:

$$\begin{aligned} \gamma BC_2 &= (\frac{T}{n} + \alpha_1 + \varepsilon)(\frac{T}{n} + \alpha_2)\cdots(\frac{T}{n} + \alpha_s) \times \\ & (\frac{T}{n} - \delta_1 - \mu_1)\cdots(\frac{T}{n} - \delta_{n-s} - \mu_{n-s}) \end{aligned} \tag{III.5}$$

$$\text{where,} \quad \mu_1 + \mu_2 + \mu_{n-s} = \varepsilon, \varepsilon > 0, \mu_i > 0. \tag{III.6}$$

Factoring out the ε and the μ's we obtain:

$$\begin{aligned}
\gamma BC_2 \;=\; & [(\frac{T}{n}+\alpha_1)(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_1)(\frac{T}{n}-\delta_2)\cdots(\frac{T}{n}-\delta_{n-s})] \\
& +[\varepsilon(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s})] \\
& -[\mu_1(\frac{T}{n}+\alpha_1)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_2-\mu_2)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s})] \\
& -[\mu_2(\frac{T}{n}+\alpha_1)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_1)(\frac{T}{n}-\delta_3-\mu_3)\cdots \\
& (\frac{T}{n}-\delta_{n-s}-\mu_{n-s})] \\
& \cdots \\
& -[\mu_{n-s}(\frac{T}{n}+\alpha_1)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_1)(\frac{T}{n}-\delta_3)\cdots(\frac{T}{n}-\delta_{n-s-1})].
\end{aligned} \tag{III.7}$$

Substituting from Equation (III.4) into Equation (III.7):

$$\begin{aligned}
\gamma BC_2 \;=\; & \gamma BC_1 \\
+ \; & [\varepsilon(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s})] \\
- \; & [\sum_{i=1}^{n-s}\mu_i(\frac{T}{n}+\alpha_1)\cdots(\frac{T}{n}+\alpha_s) \\
& (\frac{T}{n}-\delta_1)\cdots(\frac{T}{n}-\delta_{i-1}) \\
& (\frac{T}{n}-\delta_{i+1}-\mu_{i+1})\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s})].
\end{aligned} \tag{III.8}$$

Substituting the ε from Equation (III.6) into Equation (III.8), we obtain:

$$\begin{aligned}
\gamma BC_2 &= \gamma BC_1 \\
&+ [\sum_{i=1}^{n} \mu_i(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s)\times \\
&(\frac{T}{n}-\delta_1)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s})] \\
&- [\sum_{i=s+1}^{n-s} \mu_i(\frac{T}{n}+\alpha_1)\cdots(\frac{T}{n}+\alpha_s) \\
&(\frac{T}{n}-\delta_1)\cdots(\frac{T}{n}-\delta_{i-1}) \\
&(\frac{T}{n}-\delta_{i+1}-\mu_{i+1})\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s})].
\end{aligned} \tag{III.9}$$

Separating the common factor results in:

$$\begin{aligned}
\gamma BC_2 &= \gamma BC_1 \\
&+ \sum_{i=1}^{n} \mu_i(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\
&(\frac{T}{n}-\delta_{i+1}-\mu_{i+1})\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s}) \\
&[(\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_i-\mu_i) \\
&-(\frac{T}{n}+\alpha_1)(\frac{T}{n}-\delta_1)\cdots(\frac{T}{n}-\delta_{i-1})]
\end{aligned} \tag{III.10}$$

now, $(\frac{T}{n}-\delta_y-\mu_y) < (\frac{T}{n}-\delta_y)\ \forall y$.

Therefore from Equation (III.10) we obtain:

$$\begin{aligned}\gamma BC_2 &< \gamma BC_1 \\ &+ \sum_{i=1}^{n} \mu_i(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\ &\quad (\frac{T}{n}-\delta_{i+1}-\mu_{i+1})\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s}) \\ &\quad [(\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_i-\mu_i) \\ &\quad -(\frac{T}{n}+\alpha_1)(\frac{T}{n}-\delta_1-\mu_1) \\ &\quad \cdots(\frac{T}{n}-\delta_{i-1}-\mu_{i-1})]. \end{aligned} \tag{III.11}$$

Separating the common factors a second time results in:

$$\begin{aligned}\gamma BC_2 &< \gamma BC_1 \\ &+ \sum_{i=1}^{n} \mu_i(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\ &\quad (\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s}) \\ &\quad [(\frac{T}{n}-\delta_i-\mu_i)-(\frac{T}{n}+\alpha_1)]. \end{aligned} \tag{III.12}$$

Canceling equal terms and factoring out the minus sign yields:

$$\begin{aligned}\Rightarrow \gamma BC_2 &< \gamma BC_1 \\ &- \sum_{i=1}^{n} \mu_i(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}+\alpha_s) \\ &\quad (\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s}) \\ &\quad (\delta_i+\mu_i+\alpha_1). \end{aligned} \tag{III.13}$$

$$\Rightarrow \gamma BC_2 < \gamma BC_1 - \Psi. \tag{III.14}$$

$$\begin{aligned}\text{where, } \Psi &= \sum_{i=1}^{n} \mu_i(\frac{T}{n}+\alpha_2)\cdots(\frac{T}{n}-\alpha_s) \\ &(\frac{T}{n}-\delta_1-\mu_1)\cdots(\frac{T}{n}-\delta_{n-s}-\mu_{n-s}) \\ &(\delta_i+\mu_i+\alpha_1) > 0. \end{aligned} \qquad \text{(III.15)}$$

$$\text{Hence, } BC_2 < BC_1.\diamond \qquad \text{(III.16)}$$

The proof for the case when we increase one of the αs and multiple δs is similar.

Result: The *BC* for a perfectly balanced coalition is always 1.

Proof: Consider any coalition with n members, each contributing equally to the utility of the coalition (i.e. the resource distribution is $r_1, r_2, \ldots, r_n$ where $r_1 = r_2 = r_3 = \ldots = r_n = taskvalue/n$). From Equation (III.3), the *BC* is given by:

$$\begin{aligned} BC &= \frac{r_1 \times r_2 \ldots \times r_n}{(taskvalue/n)^n} \\ &= \frac{(taskvalue/n)^n}{(taskvalue/n)^n} = 1. \end{aligned} \qquad \text{(III.17)}$$

Therefore, a perfectly balanced coalition always has a $BC = 1.\diamond$

Corollary: The value of the *BC* can never exceed 1.

Proof: Since a perfectly balanced coalition has a *BC* of 1, and the *BC* increases with the level of balance, then no coalition can have a *BC* in excess of a perfectly balanced coalition.$\diamond$

The *BC* is useful for comparing the level of imbalance across coalitions of the same

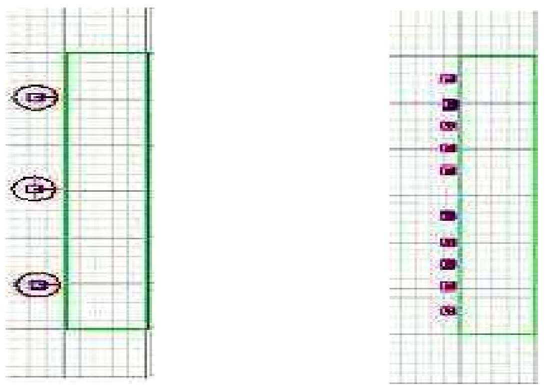

(a) Three robot coalition. (b) Ten robot coalition.

Figure III.2: Two perfectly balanced coalitions of different sizes.

size. However, the *BC* alone may not permit comparison of variable sized coalitions from a fault tolerance perspective. For example, Fig. III.2 shows two perfectly balanced coalitions performing the same box-pushing task. Fig. III.2(a) shows a coalition comprised of three large, more capable robots and Fig. III.2(b) shows a coalition comprised of ten small, less capable robots. The *BC* for both coalitions is 1, thus the *BC* alone cannot discrimate between two differently sized coalitions. Generally, larger coalitions imply that the average individual contribution and the capability requirements from each member is lower. Thus the comparison across coalitions of different sizes requires that the metric subsume elements of both balance and size. The **Fault Tolerance Coefficient (*FTC*)** is such a metric and has the following form:

$$FTC = w_1 \times (balance\ metric) + w_2 \times (size\ function). \tag{III.18}$$

where $w_1 + w_2 = 1.0$.

The $sizefunction$ may be a monotonically increasing or decreasing function of coalition size (n) depending on whether the user favors smaller or larger coalitons. The weights may be adjusted in accordance with the importance attached to both the balance and the coalition

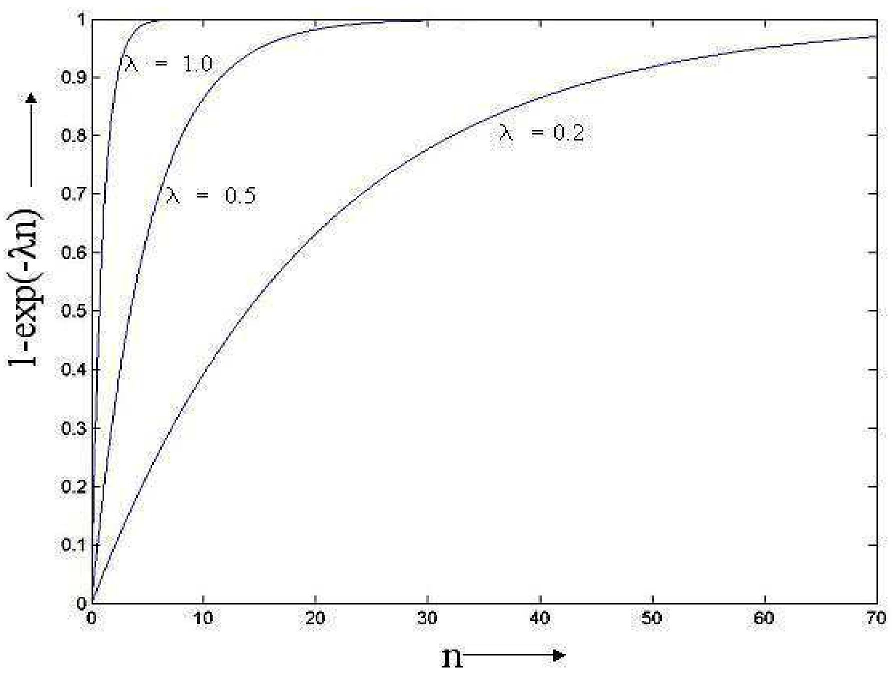


Figure III.3: Size function with λ values of 0.2, 0.5 and 1.0 as size increases.

size. The following size function is utilized in this dissertation:

$$f(n) = 1 - e^{-\lambda n}, \ \ 0 < \lambda < 1. \tag{III.19}$$

The function in Equation (III.19) is montonic and asymptotically approaches 1 (like the *BC*, it never exceeds 1, see Fig. III.3). Another important property is that after a particular point, increasing n does not result in a significant increase to the function value, i.e. the function converges to 1. This is desirable from a coalition formation perspective since after a certain point, increasing coalition size does not yield improved performance. The exact size at which the function stabilizes can be altered by varying λ in Equation (III.19). The balance metric in Equation (III.18) is any appropriate metric that satisfies the properties mentioned earlier. The balance coefficient (Equation (III.3)) is the chosen balance metric in the Chapters III and IV.

The question is how to incorporate the FTC into the algorithm in order to select better coalitions. Initially the algorithm proceeds as in Section III.1, determining the best-valued coalition without considering lopsidedness. As a modification, a list of all coalitions is maintained whose values are within a certain range (5%) of the best coalition value. The modified algorithm then calculates the FTC for all these coalitions and chooses the one with the highest FTC. This ensures that if there exists a coalition whose value is within a bound of the highest coalition value and is more fault tolerant, then the algorithm favors the coalition with higher FTC.

III.2.4 Further Optimizations

The algorithm complexity can be significantly reduced if the robots are classified according to capability requirements. For example, if the number of identical robots exceeds the maximum coalition size k, then the number of robots in that category can be assumed to be equal to k. If there are 100 available robots, each with one camera and one laser range finder, and the maximum coalition size is ten, then the coalition enumeration can assume that there are 100 identical robots instead of treating each one as a unique robot. Since the robots of a particular type are identical, coalitions with up to ten robots may be composed of any of the 100 robots. Hence, the number of candidate coalitions drops from $\approx 100^{10}$ to ten.

III.3 The Multi-Robot Coalition Formation Algorithm

The coalition formation algorithm is iterative and a task is allocated at each iteration. Within each iteration, the algorithm proceeds in two stages:

1. All possible coalitions are distributively calculated and the initial coalition values are computed.
2. Agents agree on the preferred coalitions and form them.

Stage 1-Preliminary coalition evaluation: Initially each agent A_i has a list of agents A and a list of coalitions C_{list} in which A_i is a member. A_i performs the following steps for each coalition C in C_{list}:

1. Calculate the coalitional capabilities vector (B_c) by summing the capabilities of the coalition members. Formally, $B_C = \sum_{A_i \in C} B_{A_i}$.

2. Form a list (E_c) of the expected outcomes of the tasks in set TS when coalition C performs those tasks. For each task $t_j \in TS$:

   (a) Determine the necessary capabilities for task t_j.

   (b) Compare t_j's capability vector B_j to the sum of the coalition capabilities B_c.

   (c) If $\forall\ i,\ b_i^{t_j} \leq b_i^C$ then utilize the CSP formulation (Section III.2.2) to verify the locational sensor constraints for the coalition members.

   (d) If the constraints are met, then calculate t_j's expected net outcome (e_j) with respect to C by subtracting the cost of unutilized resources from the net task-value. This is the expected net outcome (e_j) of the coalition-task pair $< C,\ t_j >$. Place $< e_j,\ C,\ t_j >$ into E_c.

   (e) Choose the highest valued coalition-task pair from E_c and place it in a set H_{CT} of highest valued coalition-task pairs.

At the end of Stage 1, each agent has a list of coalition-task pairs and coalition values.

Stage 2-Final coalition formation: Each agent (A_i) iteratively performs the following:

1. Locate in H_{CT} the coalition-task pair $< C_{max},\ t_{max} >$ with the highest value e_{max}.

2. Retain in H_{CT} all coalition-task pairs with values within a bound (5%) of e_{max}.

3. Calculate the FTC for all coalition-task pairs in H_{CT}.

4. Broadcast the coalition-task pair with the highest FTC, $< C_{FTC},\ t_{FTC} >$ along with the coalition value e_{FTC}.

5. Choose the coalition, task pair $< C_{high},\ t_{high} >$ with the highest value e_{high} from all broadcasted coalition pairs.

6. If A_i is a member of coalition C_{high}, join C_{high}, return.

7. Delete from C_{list} coalitions containing members of C_{high}.

8. Delete the chosen task t_{high} from TS.

The above steps are repeated until all the agents are deleted, until there are no more tasks to allocate, or none of the possible coalitions is beneficial. The complexity of this algorithm is unchanged from that of the multi-agent algorithm (Shehory and Kraus, 1998). The only additional overhead is due to the application of arc-consistency for constraint checking. Arc-consistency runs in $O(q^2k^3)$ time per coalition where q is the maximum number of capabilities required for a task and k is the maximum coalition size. Since both q and k do not depend on the total number of robots or the number of tasks, the check requires O(1) operations. Therefore, choosing the largest valued coalition is on the order of the number of coalitions, i.e., $O(n^{k-1})$ (Shehory and Kraus, 1998). Thus, the CSP formulation does not alter the complexity of the algorithm. An empirical evaluation of the effect of the CSP formulation on the running time of the algorithm is provided in Chapter IV, Section IV.2.

III.4 Overlapping Coalitions

Section III.3 provides our multi-robot coalition formation algorithm. The described algorithm iteratively assigns each coalition to an independent task, that the coalition is responsible for executing. This section examines the multi-robot coalition formation problem for the special case of precedence ordered tasks or tasks that have a temporal partial ordering between them. The domain introduces new complexities to the coalition formation problem due to the existence of interdependencies between the various tasks. An example where precedence ordered tasks would be of practical interest is the blocks world. Consider the task of arranging a set of blocks starting from an initial configuration into a final configuration as shown in Figure III.4. In this example, blocks A_1 and B_1 have to be arranged into their final position before block D_1 can be placed on top of them. Similarly D_1 and D_2

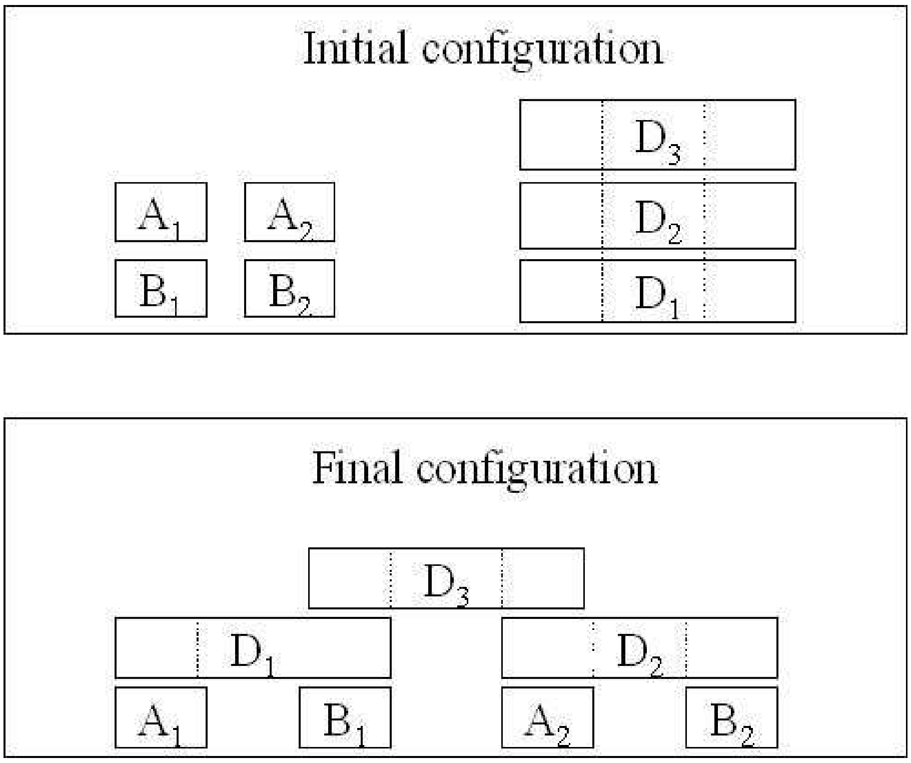


Figure III.4: The Blocks world (Shehory and Krauss 1998)

must be arranged prior to the arrangement of D_3. Thus the order of task execution must be consistent with: $t_{D_2}, t_{D_1} \preceq t_{D_3}$

Assume that Coalitions of robots have been assigned the tasks of arranging the different blocks. Thus we might have a robot X that is a member of all three coalitions responsible for the arrangement of D_1, D_2 and D_3 in Figure III.5 since these tasks have to be executed in order. In other words, we may have coalitions whose members overlap. However, closer examination reveals that the twin tasks of arranging blocks D_1 and D_2 are independent of each other, hence they may be executed in parallel or in any random order. Therefore, it would be more efficient to assign disjoint coalitions to these tasks to allow for parallel arrangement of D_1 and D_2. This idea can be generalized into a sequence of precedence ordered tasks where some intermittent sub-sequences may consist of independent tasks. Efficient solutions in this case would execute the tasks in such a subsequence in parallel by assigning disjoint coalitions to as many tasks in the subsequence as possible. Subsequences

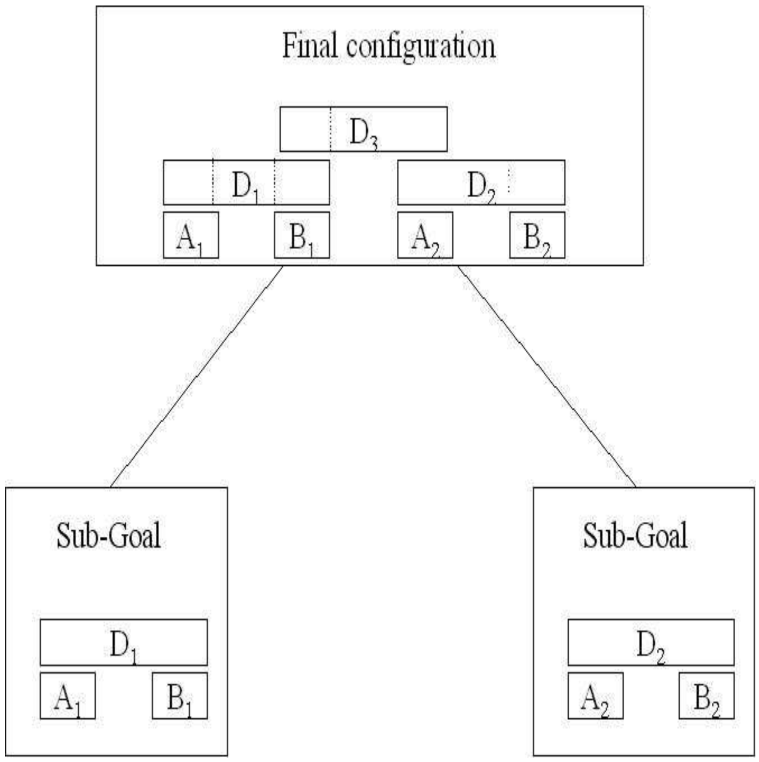


Figure III.5: Independent Tasks (Shehory and Krauss 1998)

of tasks for which interdependencies do exist may be performed using overlapping coalitions.

III.4.1 Structure of Tasks

A high level planner like GraphPlan (Blum and Furst, 1997) may be used to develop the overall partial ordering of the tasks. From this plan, a precedence order graph can be constructed, with each node of the graph representing a task and edges representing dependencies between tasks. Fig. III.6 provides a precedence graph for a set of tasks where tasks t_1 and t_2 have no outstanding dependencies, whereas task t_3 may be performed only after t_1 and t_2 are completed. Each task has a utility associated with the task. The utility of a task t_i will depend on:

1. The number of tasks dependent on the completion of t_i.

2. The resource requirements for t_i.

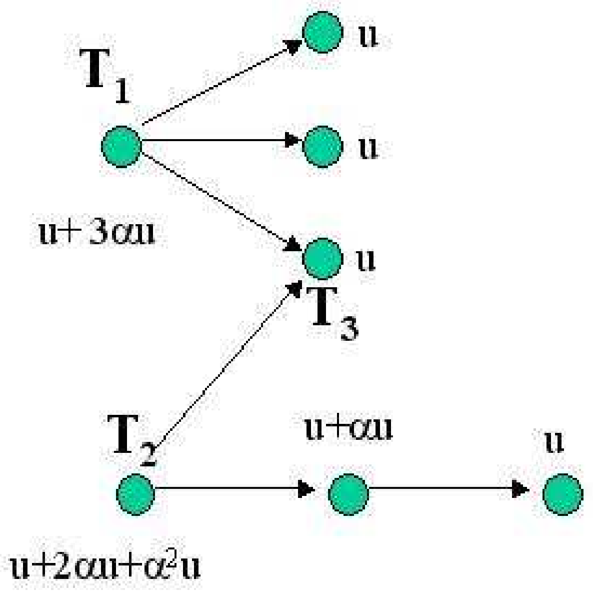


Figure III.6: Precedence order graph for a set of tasks. Each task independently has identical utility (u) and the utilities are propagated backwards from the leaves with a discount factor α.

Formally, the utility of each task is evaluated by calculating the resources it consumes in addition to the utility of dependent tasks, propagated backwards from the leaf nodes and weighed by a discount factor $\alpha (0 < \alpha < 1)$. The utilities from two branches are summed at the intersecting node. Thus, tasks that have a greater number of immediately dependent tasks are assigned a higher utility. For example in the precedence graph in Fig. III.6, assuming that all independent tasks have an identical task utility (u), then in the precedence ordered graph task T_1 will have a higher utility $(u + 3u\alpha)$ than task $T_2 (u + 2u\alpha + \alpha^2 u)$ even though there are three tasks dependent on both T_2 and T_1.

III.4.2 The Precedence Ordered Coalition Formation Algorithm

The idea is to find the largest subset of tasks consistent with the task ordering that can be executed with the currently available set of robots. This is achieved by continuously updating the set of executable tasks and free agents to perform those tasks, while utilizing the coalition formation algorithm outlined in Section III.3. Formally the algorithm is described as follows:

Initialization: Each agent stores in memory the precedence order graph generated by a high level planner. The agents extract all tasks that have no unfulfilled pre-requisite tasks. Call this list of candidate tasks the candidate list, T_{cand}. Each agent also maintains a list, of free agents F_a representing agents not currently engaged in performing a task.

1. Coalitions are formed from the agents in F_a for performing the tasks in T_{cand} using the coalition formation algorithm from Section III.3. Agents that are assigned to a task are removed from F_a. Similarly, allocated tasks are removed from T_{cand}.

2. Upon completion of a task t_j by a coalition C, the lowest numbered member A_i of coalition C broadcasts the coalition task pair $< t_j, C >$.

3. Upon receipt of a task completion message $< t_j, C >$ each agent performs the following:

   (a) Add to the list of free agents, F_a the members of the coalition C that completed task t_j.

   (b) Check the precedence graph and include in T_{cand} any fresh tasks that no longer have outstanding dependencies after the execution of t_j.

The above steps are repeated until there are no more tasks remaining to be executed in the precedence order graph.

As mentioned in Chapter I, despite the existence of a plethora of coalition formation algorithms in the Distributed Artificial Intelligence literature, none of these algorithms have previously been demonstrated in a multi-robot setting with real world tasks. This chapter identifies reasons for this divide between the multi-agent and multi-robot domains and provides solutions to the perceived difficulties while modifying and extending a well known multi-agent coalition formation algorithm to the multi-robot domain.

CHAPTER IV

ALGORITHMIC VALIDATION EXPERIMENTS

Chapter III introduced a popular heuristic-based algorithm for software agent coalition formation and provided modifications and extensions for application to the multi-robot domain. This chapter presents experiments testing the validity of the suggested multi-robot coalition formation algorithm. Eight sets of experiments were conducted, with each of the first three highlighting a suggested modification to Shehory and Kraus' coalition formation algorithm. Five additional experiments were conducted to validate the new algorithm with a larger number of robots. The first two experiments demonstrate the impact of the FTC on the resulting coalitions, both in simulation and with real world robots. The next three experiments demonstrate the algorithm's applicability to real world tasks in the multi-robot domain.

IV.1 Communication Experiment

The first experiment measured the variation of time required to evaluate coalitions with and without communication. The number of agents and maximum coalition size were both fixed at five. Communication occurred via TCP/IP sockets over a wireless LAN (Figure IV.1 provides the results). The time for coalition evaluation without communication is significantly less than the time required for evaluation with communication. The time without communication increases at a faster rate as the number of tasks increases. This result occurs because the agent must evaluate a larger number of coalitions when it forgoes communication. Presumably, the two conditions will eventually meet and thereafter the time required with communication will be less than that required without communication. For any practical Agent/Task ratio the time saved by minimizing communication outweighs the extra computation incurred.

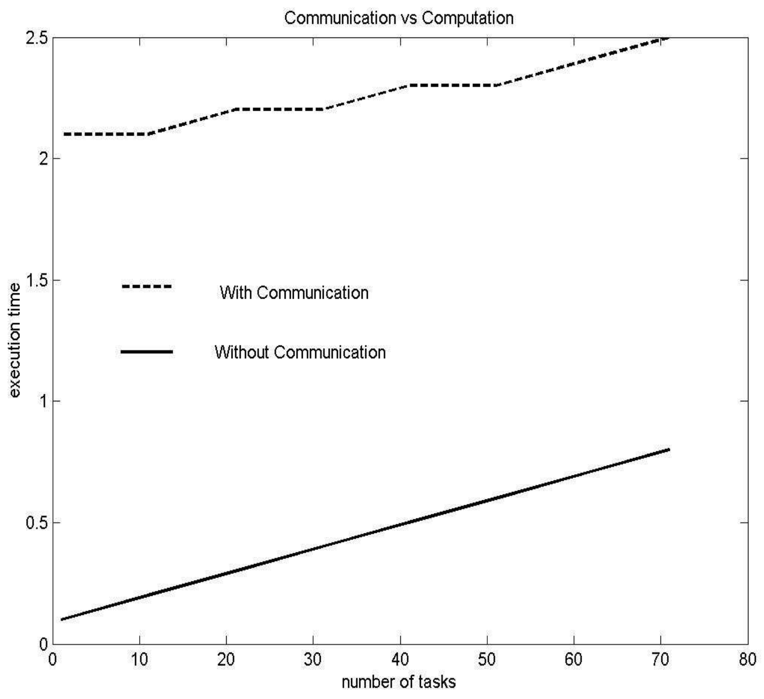


Figure IV.1: Execution time with and without communication.

IV.2 CSP vs. NON-CSP Experiment

The second set of experiments measured the effect of the CSP formulation on the algorithm execution time and demonstrates the algorithm's scalability. Figure IV.2 measures the variation of coalition formation time with and without constraint checking in the constraint satisfaction graph as the number of agents increases. Figure IV.3 shows the variation of execution time as the number of tasks increases. The task complexity in these experiments was similar to the box-pushing task. It can be seen from Figures IV.2 and IV.3 that the CSP formulation does not add a great deal to the algorithm's execution or running time. This implies that the CSP formulation can be used to test the validity of a multiple-robot coalition against a task without incurring much overhead.

In some cases, the CSP formulation actually saves time by disqualifying a large number of coalitions from performing a particular task. These coalitions are then eliminated from future consideration in any of the future iterations. Therefore the net evaluation time is

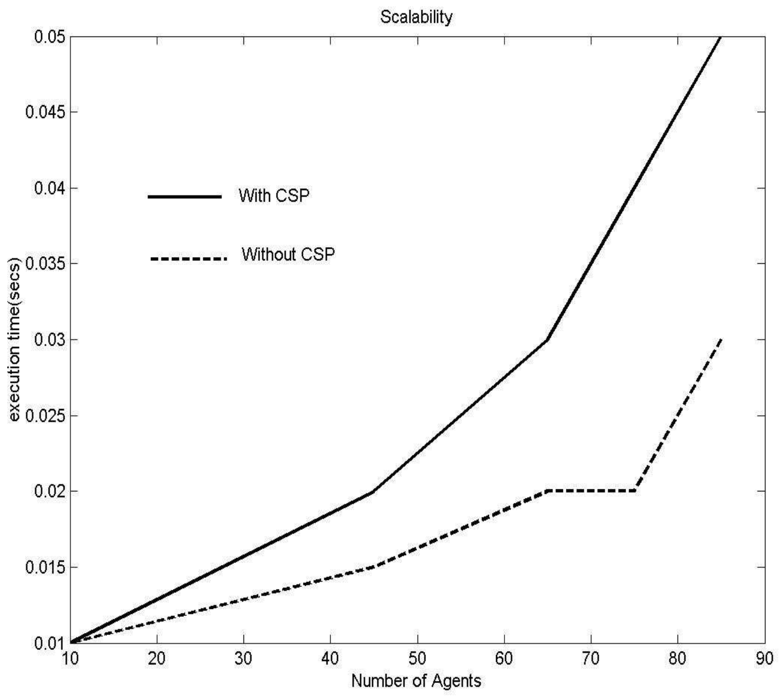


Figure IV.2: Execution time vs. Number of Agents.

actually sometimes slightly reduced as shown in Figure IV.3.

For completeness, Figure IV.4 shows the three dimensional plots for the execution time with and without the CSP formulation over the complete range of values for the number of tasks and the number of agents. The surface plot shows that the CSP formulation does not alter the complexity of the coalition formation algorithm.

IV.3 Fault Tolerance Coefficient Experiment

This experiment demonstrates the effect of utilizing the FTC to favor the creation of more fault tolerant coalitions. The Player/Stage simulation environment (Gerkey et al., 2001) was employed. The tasks required pushing a very large box by jointly exerting forces on the box. The degree of task difficulty was adjusted by varying the box's size and its coefficient of friction with the floor. Adjusting the forces the robots could exert varied the robots' capabilities. (Note: boxes in the simulations did not actually move). The FTC used

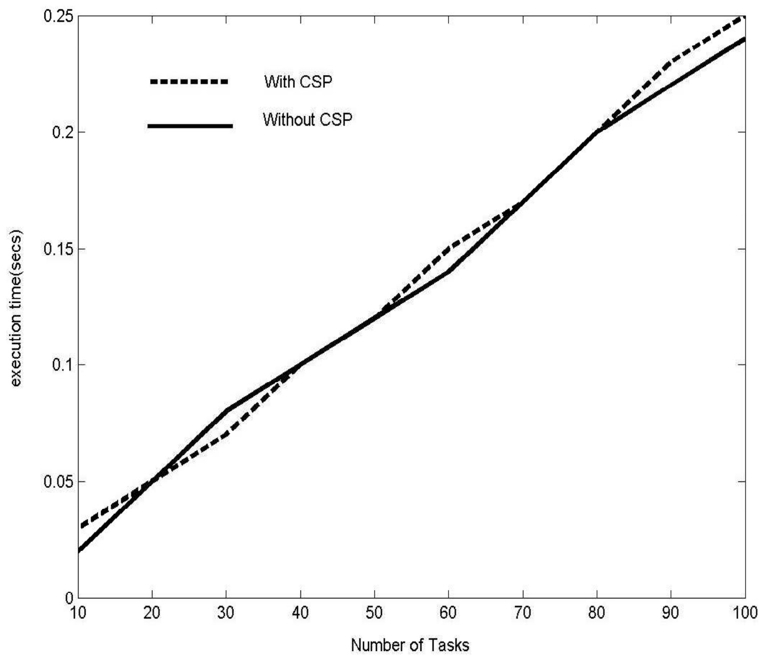


Figure IV.3: Execution time vs. Number of Tasks.

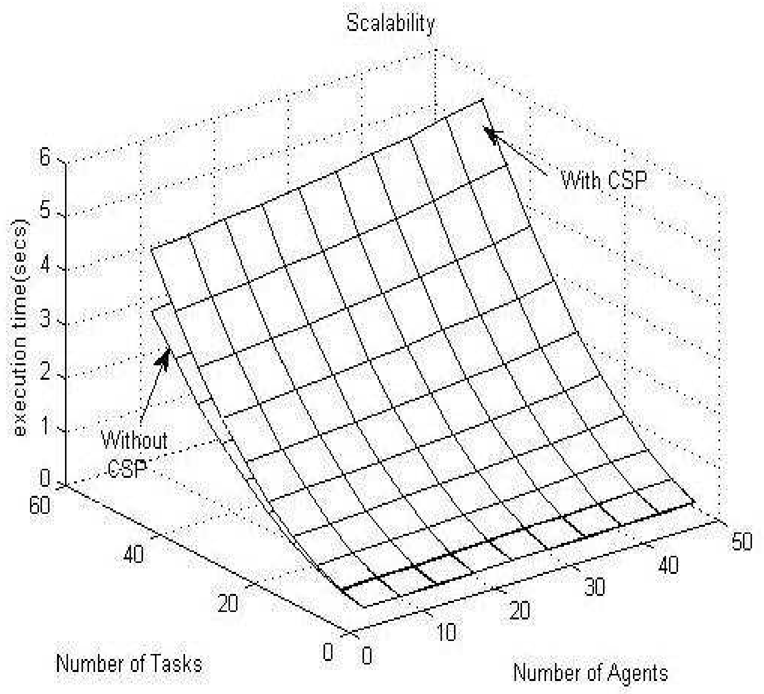


Figure IV.4: Execution time as a function of Number of Tasks and Number of Agents.

for these experiments was:

$$FTC = w_1 \times BC + w_2 \times [f(n)],\ w_1 = w_2 = 0.5 \tag{IV.1}$$

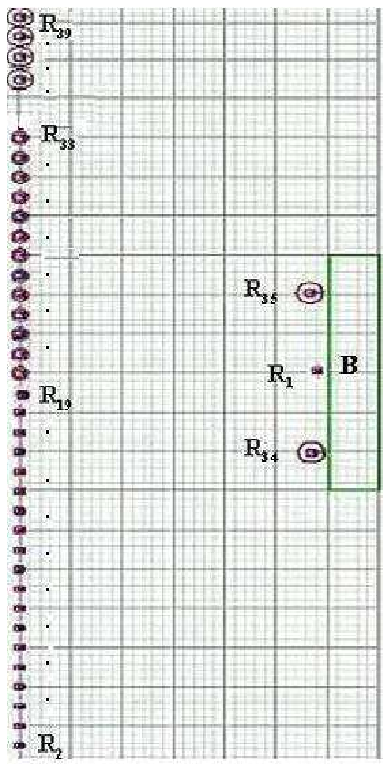
$$\text{where, } f(n) = [1 - exp(-\lambda n)],\ \text{with } \lambda = 0.5. \tag{IV.2}$$

Box-pushing required the robot to possess a laser range finder, be mobile, be able to exert a certain force F, and be able to communicate with coalition members. Thirty nine simulated robots were employed, as shown in Figure IV.5(a). The robots were numbered 1 to 39 from bottom to top along the left side of the figure. Each robot had a specific force capability type: small robots exerted five units of force ($R_1 - R_{19}$), medium sized robots exerted 15 units of force ($R_{20} - R_{33}$) and large robots had 25 units of force ($R_{34} - R_{39}$). The robots used the incremental SLAM algorithm (Gerkey et al., 2001) for localization and the vector field histogram algorithm (Borenstein and Koren, 1991) for navigation and obstacle avoidance. The maximum allowed coalition size (k) was fixed at 15.

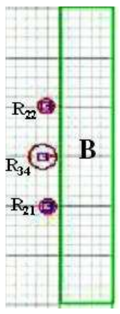
Simulation snapshots are provided for a task requiring 55 units of force. Figure IV.5(a) shows the resulting coalition without incorporating fault tolerance. The coalition is comprised of two large robots (R_{34}, R_{35}) and one small robot (R_1). The BC and FTC values for this coalition are 0.51 and 0.60 respectively.

Figure IV.5(b) shows the same task performed while incorporating the FTC with a decreasing size function, $-f(n)$, placing a low priority on fault tolerance and a high priority on minimizing the number of robots. The resulting coalition is comprised of two medium sized robots (R_{21}, R_{22}) and one large robot (R_{34}). The resulting coalition is more balanced and has a higher BC (0.91) and consequently a higher FTC (0.80) than the FTC value of the coalition in Figure IV.5(a).

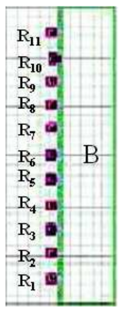
Figure IV.5(c) depicts the experiment conducted with the size function, $f(n)$, that favors the formation of larger coalitions. The resulting coalition consists of eleven small robots ($R_1, R_2, ..., R_{11}$). Thus, a perfectly balanced coalition is obtained ($BC = 1$). The advantage



(a)



(b)



(c)

Figure IV.5: The coalition formed (a) Without the FTC. (b) With the FTC and a size function = $-f(n)$. (c) with FTC and a size function = $f(n)$.

is that a larger number of small, less capable robots should have higher fault tolerance. If one robot fails, it should be easier to replace as opposed to replacing a larger, more capable robot. The coalition's FTC for this coalition is the highest of all possible evaluated coalitions (Figure IV.5(a)-IV.5(c)) at 0.996.

IV.4 Real Robot Experiments

The FTC simulation experiments were ported to real robots. The challenge was to find suitable tasks that the robots could perform and whose difficulty could be varied, while also quantifying the robots' capabilities so that a robot's utility for a particular task could be assessed.

The experiments ported the algorithm to three Pioneer 3-DX robots. The experimental tasks involved pushing a box through a distance of one meter in a straight line from its current position. A rod was inserted through the box, preventing interference between the robots. The maximum coalition size was restricted to two robots. The robots positioned themselves so that the box's net torque was approximately zero in order to ensure that the box did not rotate beyond the acceptable limits. The FTC parameters were identical to those defined for the experiments in Section IV.3, where the size function is as defined in Equation (IV.2). Since the size is constant for these experiments, the FTC is equivalent to the BC i.e. $FTC = 0.5 \times BC + constant$.

Velcro was attached to the rod to prevent slippage. The boxes contained weights that permitted variation in task difficulty. The robots know their initial positions, the box position, and navigate using odometry. The robots' capabilities were changed based on varying the robots' speed. Every 0.05 m/s of speed increases the robots' capability by ten units. Each pound of weight in the box increased the task value by ten. The purpose was to verify the task allocation and coalition formation rather than monitoring the quality of the task execution. Figure IV.6 shows an experimental execution.

The experimental results are tabulated in Table IV.1. Size did not play a role in de-

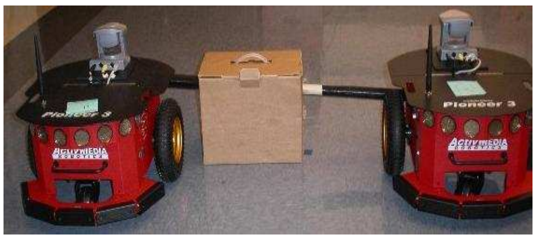


Figure IV.6: Two pioneer DX robots pushing a box after forming a coalition.

Table IV.1: Coalition formation results for real robot box pushing tasks.

Exp.	Robot Capabilities			Task Value		Coalitions Formed			
	Robot1	Robot2	Robot3	Task1	Task2	With *BC*		Without *BC*	
						Members	*BC*	Members	*BC*
1	40	40	60	80	100	1,2	1	1,3	0.96
2	20	20	30	40	50	1,2	1	1,3	0.96
3	45	25	25	50	70	2,3	1	1,2	0.97
4	25	20	20	50	70	-	-	-	-
5	15	20	10	50	60	-	-	-	-
6	35	35	40	100	80	-	-	-	-
7	10	15	15	20	30	2,3	1	2,3	1
8	20	25	25	70	50	2,3	1	2,3	1
9	30	30	30	50	60	1,2	1	1,2	1

termining the coalitions that were formed since only three robots were employed. The coalition formation was influenced primarily by the *BC*. Experiments 1 - 3 demonstrate that different robotic coalitions form depending on whether or not the *BC* is utilized. For example, in Experiment 1, robots 1 and 2 form a coalition to complete task 1 (without *BC*) even though there is the potential for other coalitions ([2, 3] or [1, 3]) to perform the more valuable task 2. This results occurs because the *BC* is higher for the task 1 coalition [1,2] than for the potential task 2 coalitions [1, 3] or [2, 3].

Experiments $4-6$ demonstrate situations where the three robots could form a coalition but do not because the maximum coalition size is limited to two robots. This result demonstrates that the algorithm does not perform an exhaustive search through the coalition space, rather it only considers coalitions with fewer than k (here two) members.

The final experiments 7 - 9 are indicative of situations when the *BC* has no impact on the resulting coalitions. The *BC* value is inconsequential because the highest valued coalition also has $BC = 1$.

The real world robot experiments required minor algorithm adjustments to account for wheel slippage and the mapping of capabilities to real numbers. However, the experiments successfully establish that the algorithm performs satisfactorily with real robots.

IV.5 Coalition Formation Experiments

The Sections IV.3 and IV.4 experiments provided preliminary verification of the algorithm's applicability to real world tasks; however it was necessary to validate the algorithm with a larger number of robots. Experiments were conducted with simulated and real world robots that had heterogeneous sensory capabilities. Each robot possessed a subset of the following capabilities: bumper, laser range finder, camera, mobile, communications, and ultrasonic sonar. The FTC employed in these experiments was identical to that used in Section IV.3. The following tasks were employed for the experiments:

- **Box-Pushing**: This task requires two robots form a coalition, navigate through the environment from their initial locations to either side of a large box and simultaneously push the box through a straight distance of one meter. The task requires that the two robots be equipped with laser range finders, be able to communicate with each other, and be mobile. The robots navigate through the environment using the vector field histogram algorithm (Borenstein and Koren, 1991) and orient themselves perpendicular to the box using the laser range finders. The incremental SLAM algorithm (Gerkey et al., 2001) provided laser corrected odomotery. A synchronizing message ensured that the robots pushed the box simultaneously. This task requires that the coalition members perform their actions in a synchronized fashion and hence is a tightly-coupled task.

- **Cleanup**: This task involves two to five robots forming a coalition to clear a room of small colored boxes by pushing them to the edge of the room. Each robot requires a camera, a set of sonars, and must be mobile. A behavior based approach was employed to perform the tasks. Behaviors include locating the colored boxes and pushing them towards the wall. The robots use a sonar based obstacle avoidance behavior and the cameras for object (box) tracking. This task does not require synchronization between robots and therefore is a loosely-coupled tasks.
- **Sentry-Duty**: This task involves two to four robots forming a coalition to monitor different areas of the environment for motion. The task requires that the robots be equipped with laser range finders and be mobile. Again the vector field histogram algorithm is used for navigation and the laser range finders are used to detect motion. The incremental SLAM algorithm provided laser corrected odometery. The robots do not communicate with each other, but the coalition members combine to monitor a particular area for motion. Therefore, this task falls in between a tightly-coupled and a loosely-coupled task.

IV.5.1 Single Task Robot Validation

These experiments demonstrated that the coalition formation algorithm operates independent of the nature of the tasks and the task methodology utilized to perform those tasks. Separate experiments were performed with the box-pushing, sentry duty, and cleanup tasks. The simulation and real world experiments for each type of task are tabulated in Table IV.2. The maximum coalition size for all experiments was four. All real-world experiments, involved a total of twelve heterogeneous Pioneer DX robots.

The box-pushing task required two robots form a coalition to push a large box through a straight distance of one meter. The tightly coordinated task required synchronous message passing between robots. The Experiment 1 simulation, in Table IV.2, was conducted with four box-pushing tasks and ten simulated robots. All four box-pushing tasks were successfully allocated and performed by four different coalitions, as shown in Figure IV.7. The corresponding real world experiment was comprised of two box-pushing tasks, each requiring two robots that were successfully allocated, Figure IV.8 provides an example run.

The cleanup task (see Figure IV.10.) involved forming a coalition to clear an area of boxes. The implementation is behavior based with robots performing a random walk until a colored object is identified via image processing. The robot then moved towards the object (and consequently pushed it) until it's sonar sensors indicated that the robot was close to a

Table IV.2: Results from simulated and real robots forming coalitions for a single task

Exp.	Task Type	Task Methodology	Simulation		Real Robots	
			N_{Robots}	N_{Tasks}	N_{Robots}	N_{Tasks}
1	Box-pushing	Synchronized (*Tightly coupled*)	10	4	12	2
2	Foraging	Behavior-Based (*Loosely coupled*)	12	1	12	1
3	Sentry-Duty	Semi-Synchronized (*Intermediate coupling*)	9	2	12	2

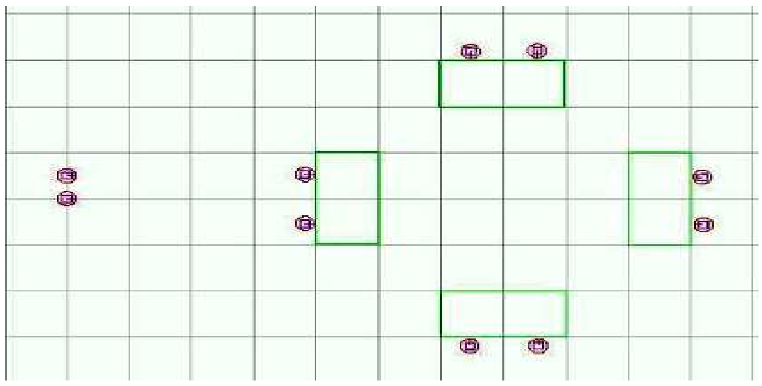


Figure IV.7: Simulated coalitions of two robots performing four box-pushing tasks.

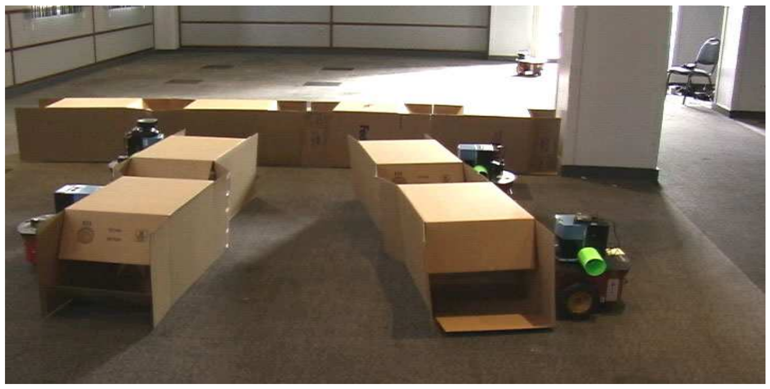


Figure IV.8: Two coalitions of two robots performing two box-pushing tasks.

wall, in which case the object was deposited, obstacle avoidance became active, the robot turned, and resumed a random walk. The corresponding simulation task is the foraging task that involved robots picking up objects of interest (pucks) and depositing them at a goal location. Experiment 2, in Table IV.2, represents a simulation run requiring four robots to form a coalition from twelve simulated robots. Figure IV.9 shows four robots performing the foraging task. The corresponding real world cleanup task involved four robots forming a coalition as shown in Figure IV.10.

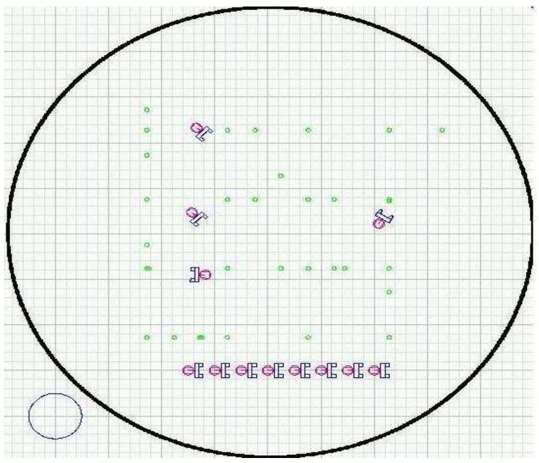


Figure IV.9: A simulated four robot coalition performing a foraging task.

Figure IV.10: Robot coalition performing the real-world cleanup task.

The sentry-duty task involved a team navigating through the environment to fixed positions (normally an entrance or exit) and performing motion detection from these positions. Figure IV.11 shows a simulation run in which two coalitions were formed comprised of two robots each from a possible nine robots. This simulation is represented by Experiment

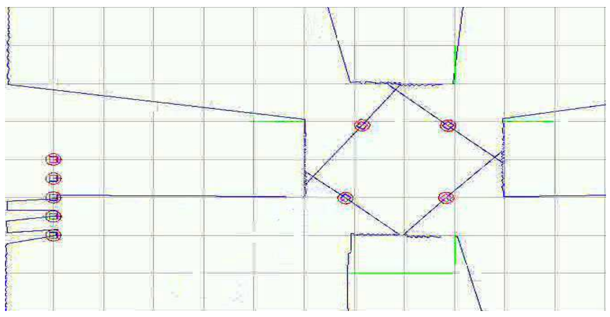


Figure IV.11: Two coalitions (two robots each) performing a sentry-duty task.

3 in Table IV.2. During the real robot sentry-duty experiments, two tasks were successfully allocated, each requiring two robots.

All tasks were successfully allocated and performed by the coalition formation algorithm in both simulation and real robot experiments. These results demonstrate that the algorithm operates independently of the nature of the tasks (loosely-coupled vs. tightly-coupled), or the methodology utilized to perform the tasks (behavior-based vs. synchronized).

IV.5.2 Multiple Task Robot Validation

These experiments demonstrated that the task allocation operates independently of task diversity. Separate coalitions were formed for combinations of the Box-pushing (tightly-coupled) task, the Cleanup task (loosely-coupled, behavior-based), and the Sentry-Duty task (intermediate-coupling). Table IV.3 illustrates three simulation runs with different combinations of each of the three tasks. The coalitions were chosen from 40 heterogeneous robots. Figure IV.12 shows a simulation that involved four distinct box-pushing tasks, a foraging task, and two sentry-duty tasks (corresponding to Experiment 1 in table IV.3). All experiments involved combinations of the box-pushing task requiring two robots, the cleanup task requiring four robots, and the sentry-duty task requiring two robots. Experiment 2 represents a simulation with three box-pushing tasks, two foraging tasks, and two sentry duty tasks. Experiment 3 represents a simulation with four box-pushing tasks, two foraging tasks, and two sentry duty tasks. All three runs represent simulations where

Table IV.3: Results from simulated robots forming coalitions for multiple tasks.

Experiment	box-pushing		foraging		sentry-duty	
	N_{tasks}	N_{alloc}	N_{tasks}	N_{alloc}	N_{tasks}	N_{alloc}
1	4	4	1	1	2	2
2	3	3	2	2	2	2
3	4	4	2	2	2	2

Table IV.4: Results from real world robots forming coalitions for multiple tasks.

Experiment	box-pushing		cleanup		sentry-duty	
	N_{tasks}	N_{alloc}	N_{tasks}	N_{alloc}	N_{tasks}	N_{alloc}
1	2	2	1	1	2	2
2	2	2	4	1	10	4
3	1	1	1	1	2	2

the algorithm successfully allocated all tasks.

The real world robot experiments employed thirteen robots. Eight robots were equipped with laser range finders and five were equipped with cameras. All real robot tasks required the same number of robots as defined for the simulation experiments. Figure IV.13 shows an experiment with a cleanup task, 2 sentry duty tasks, and a box-pushing task (Experiment 3 in Table IV.4). Table IV.4 illustrates experiments with different combinations of the three tasks performed by the real robots. Experiment 1 represents a situation in which the task requirements are met and all tasks are successfully allocated. Experiment 2 represents a situation in which all tasks could not be successfully allocated because of insufficient resources. Experiment 3 represents a situation where the resources exceed the resource requirements. Both Tables IV.3 and IV.4 demonstrate that the algorithm is applicable to a combination of task types (and task methodologies) when multiple tasks must be simultaneously allocated to coalitions.

Each case allocated the tasks to coalitions that successfully performed the task. Figure IV.14 indicates the message traffic rate for a robot participating in the task allocation process. The peaks correspond to the broadcasting of coalition-task values, while the flat regions correspond to periods spent evaluating the coalition lists. As more tasks are allo-

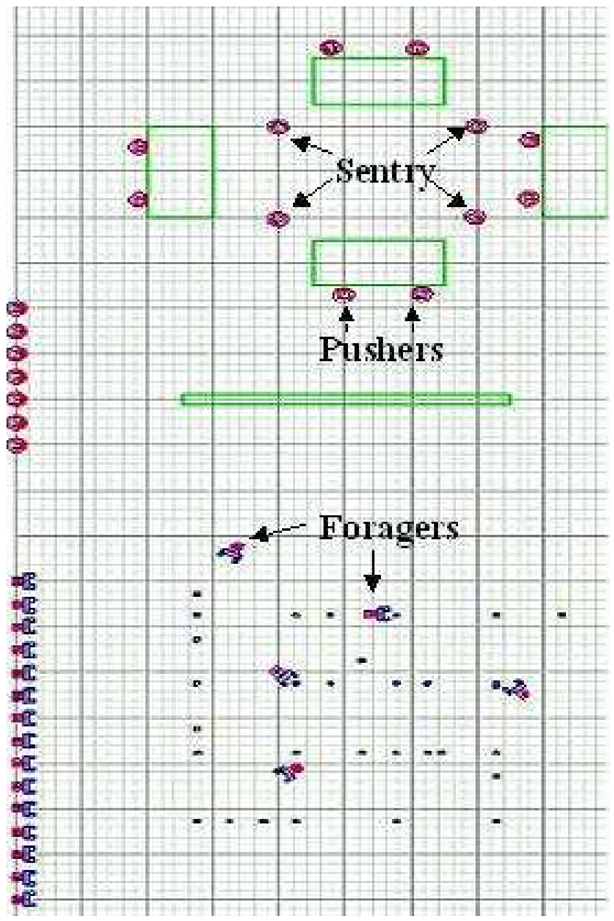


Figure IV.12: Simulation of four two-robot box-pushing tasks, two two-robot sentry-duty tasks, and a four robot foraging task.

cated, fewer robots remain, and the number of transmitted messages decreases with each iteration. The sharpest spike or messaging burst required a bandwidth of approximately 2.34Kbps, which is very acceptable given the available bandwidth on modern networks. This suggests that the algorithm's messaging requirements should scale for an even larger number of robots.

IV.5.3 Coalition Formation in Precedence Ordered Environments

Chapter III, Section III.4.2 presented an algorithm to extend the application of the multi-robot coalition formation algorithm to domains where tasks had a partial ordering between them. This section presents experiments, both in simulation and the real world, demonstrating the successful application of the algorithm.

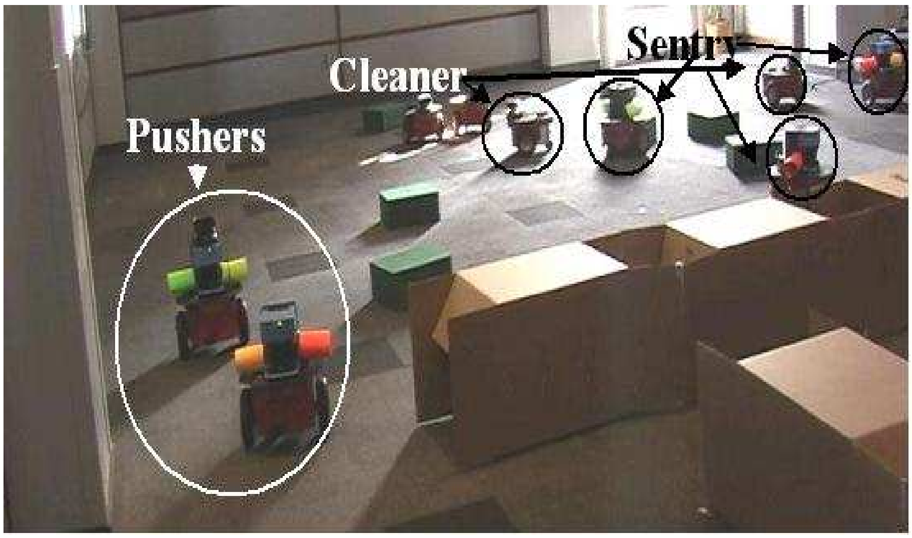


Figure IV.13: The real robots performing a combination of box-pushing, sentry-duty and cleanup tasks.

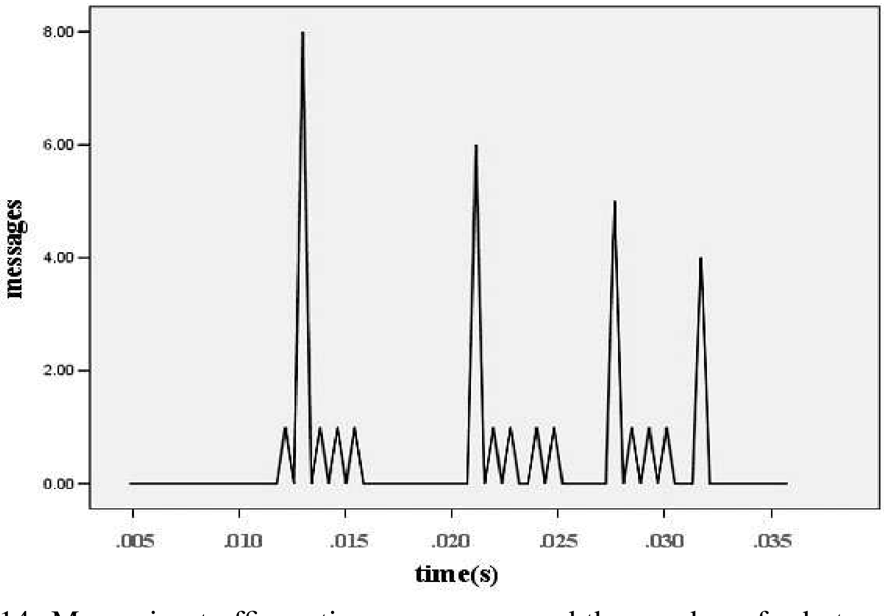


Figure IV.14: Messaging traffic as time progresses and the number of robots participating decreases.

Patrol: An additional task that is employed for these experiments is the Patrol task that involves a pair of patrol robots navigating to and exploring a room. In order to accomplish the task, the robots must be equipped with laser range finders, a map of the environment, and sensors to detect contaminants. This task does not require communication between robots and similar to the sentry duty task, requires intermediate coupling.

Simulation Experiment

The environment is a mapped indoor urban building with rooms and corridors. Boxes are placed at specific locations in the building to block access to rooms or other corridors as shown in the simulated environment in Figure IV.15. Thus there are two types of tasks to be performed by the robots in this environment, namely box-pushing and patrol. Eight robots were simulated for this experiment. The four *pusher robots* (robots $R_1 - R_4$ in Figure IV.15) were equipped with bumpers, laser range finders, and could communicate with each other. The four *patrol robots* (robots $R_5 - R_8$ in Figure IV.15) did not have bumpers but had simulated sensors that could be used to detect contaminants. All robots had a map of the environment. The utility for all independent box-pushing and patrol tasks was identical. Thus, the overall utility of a task T in the precedence order graph depended largely on how many tasks were dependent on T.

The precedence order graph for the given set of tasks depicted in Figure IV.15 is shown in Figure IV.16. The Chapter III, Section III.4.2 algorithm was utilized to allocate and perform tasks dynamically. The robots formed coalitions on the fly to perform tasks as they became eligible for execution. (Note: Due to imperfections in the simulator the robots did not push the boxes in a realistic manner. Due to this problem occasionally the boxes had to be moved manually, however in the real world experiments the boxes were pushed autonomously by the robots).

Figure IV.17 demonstrates the various box-pushing and patrol tasks being performed by the different coalitions based upon the precedence order graph in Figure IV.16. Figure

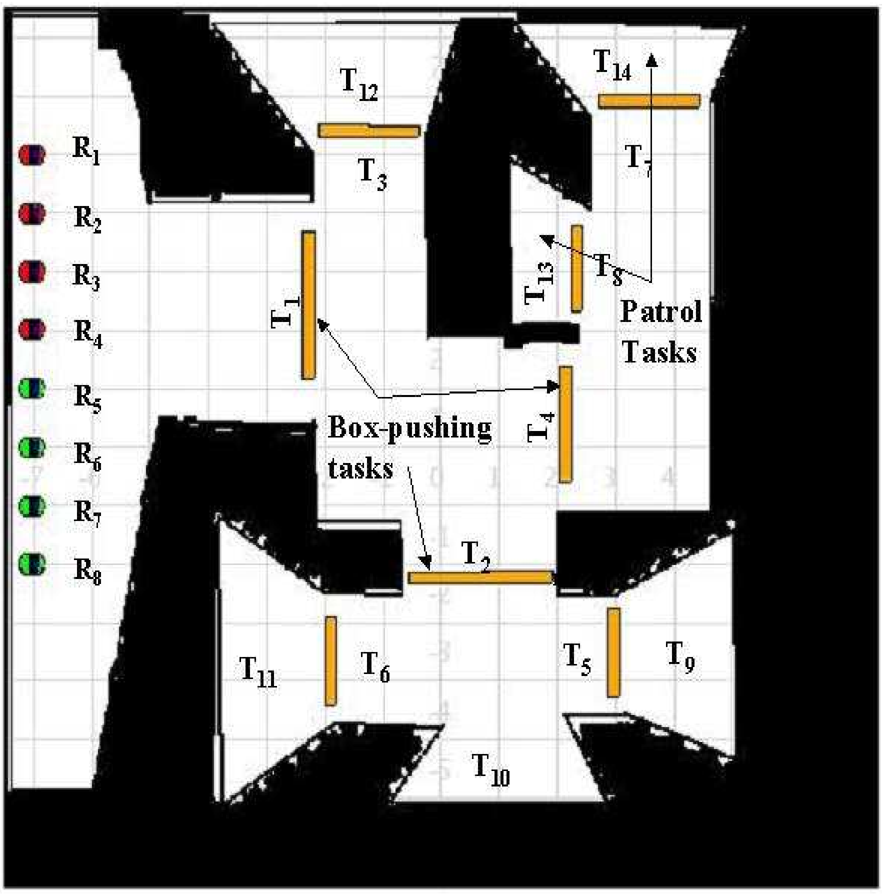
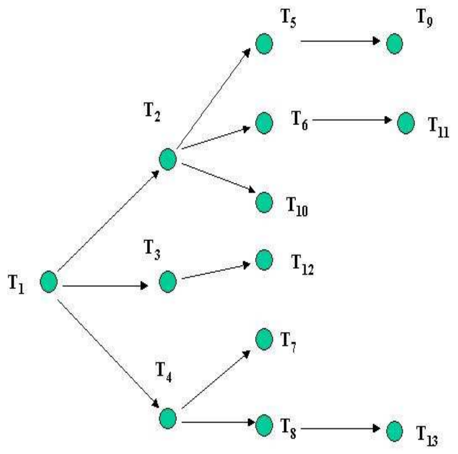
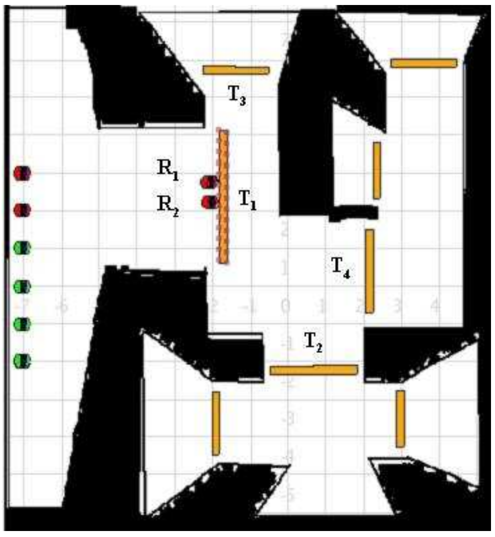
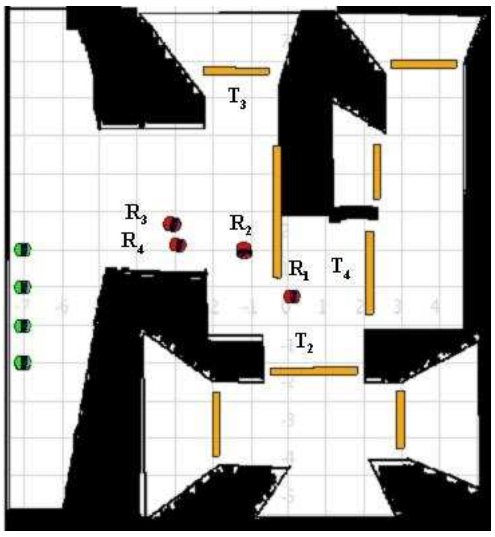


Figure IV.15: The urban indoor task environment.

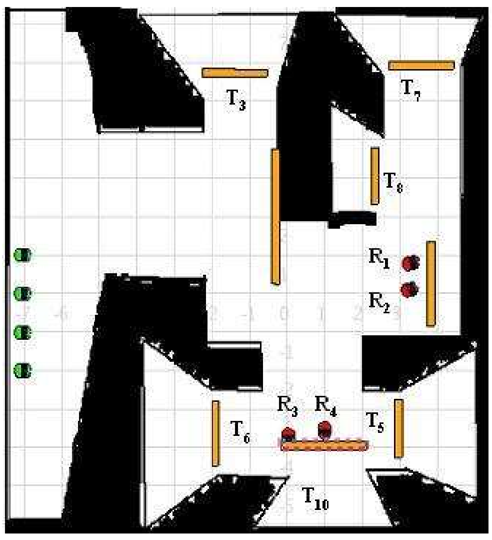

$$\begin{tikzcd}
 & & T_5 \arrow[r] & T_9 \\
 & T_2 \arrow[ur] \arrow[r] \arrow[dr] & T_6 \arrow[r] & T_{11} \\
 & & T_{10} & \\
T_1 \arrow[uur] \arrow[r] \arrow[ddr] & T_3 \arrow[r] & T_{12} & \\
 & & T_7 & \\
 & T_4 \arrow[ur] \arrow[r] & T_8 \arrow[r] & T_{13}
\end{tikzcd}$$
Figure IV.16: The precedence order graph for the simulation environment.


T$_3$
R$_1$
R$_2$
T$_1$
T$_4$
T$_2$

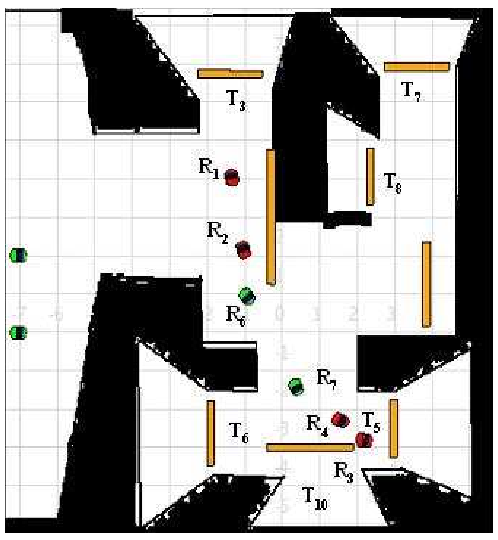
(a)


T$_3$
R$_3$
R$_4$
R$_2$
R$_1$
T$_4$
T$_2$

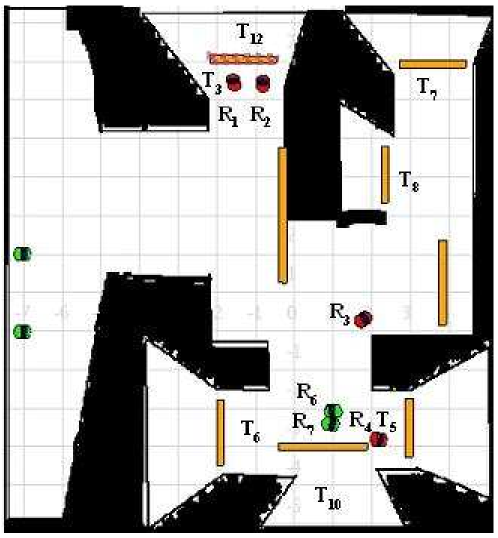
(b)


T$_3$
T$_7$
T$_8$
R$_1$
R$_2$
T$_6$
R$_3$
R$_4$
T$_5$
T$_{10}$

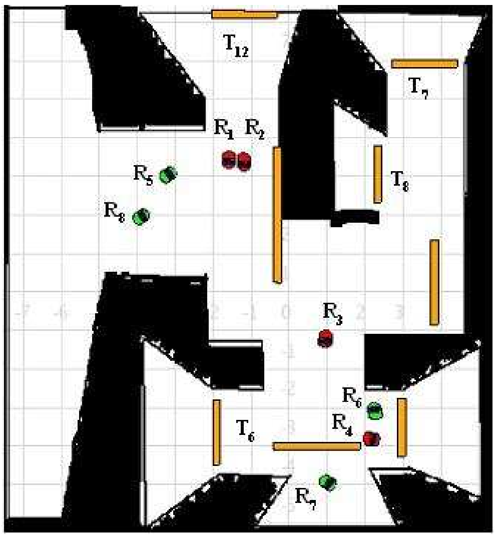
(c)


T$_3$
T$_7$
R$_1$
T$_8$
R$_2$
R$_6$
R$_7$
R$_4$
T$_5$
T$_6$
R$_3$
T$_{10}$

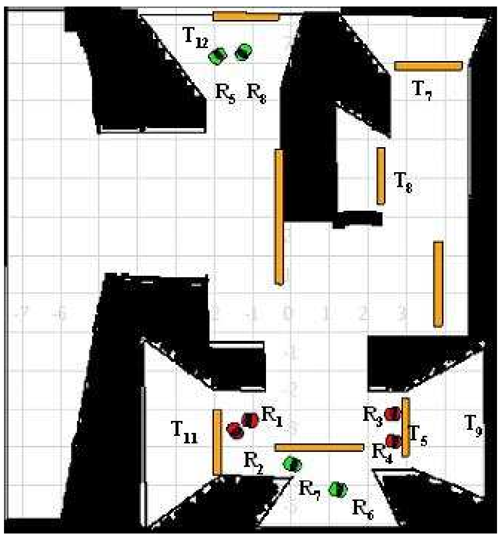
(d)


T12
T3
R1 R2
T7
T8
R3
R6
R7 R4 T5
T6
T10

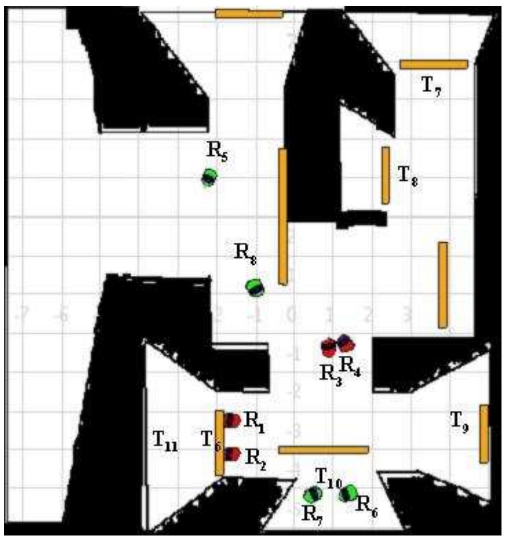
(e)


T12
T7
R1 R2
R5
T8
R8
R3
R6
R4
T6
R7

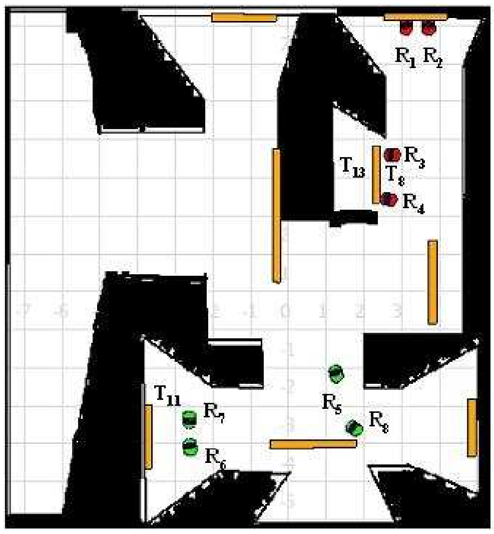
(f)


T12
R5 R8
T7
T8
T11
R1
R3
T5
T9
R4
R2
R7
R6

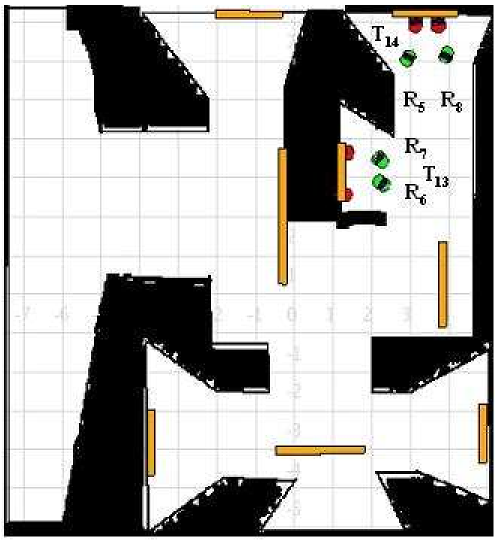
(g)


T7
R5
T8
R8
R3 R4
T9
R1
T11
T6
R2
T10
R7
R6

(h)



(i)



(j)

Figure IV.17: Coalitions being formed on the fly to perform box-pushing and patrol tasks as they are unblocked.

IV.17(a) shows Robots R_1 and R_2 performing task T_1. Completion of task T_1 unblocks tasks T_2, T_3 and T_4; however since only two box-pushing tasks can be performed at a time, tasks T_2 and T_4 are allocated coalitions $\{R_3,\ R_4\}$ and $\{R_1,\ R_2\}$ as shown in Figure IV.17(b) and IV.17(c). Once, T_2 and T_4 are executed, tasks T_5, T_6, T_7, T_8, and T_{10} become unblocked. Figure IV.17(d) shows the robot coalitions navigating to perform tasks T_3, T_5, and T_{10}. Task T_3 is being performed in Figure IV.17(e) by coalition $\{R_1,\ R_2\}$, thereby unblocking task T_{12}. Figure IV.17(f) shows robots navigating to perform T_{12} and Figure IV.17(g) shows T_{12} being performed by a coalition $\{R_5,\ R_8\}$ of patrol robots and T_5 being performed by a coalition of pushers $\{R_3,\ R_4\}$. Figure IV.17(h) shows tasks T_6 and T_{10} being performed by coalitions of box-pushers $\{R_1,\ R_2\}$ and patrol robots $\{R_6,\ R_7\}$ respectively. Tasks T_7, T_8 and T_{11} are shown performed in Figure IV.17(i) by coalitions $\{R_1,\ R_2\}$, $\{R_3,\ R_4\}$ and $\{R_6,\ R_7\}$ respectively. Finally, T_{13} and T_{14} are performed by coalitions $\{R_6,\ R_7\}$ and $\{R_5,\ R_8\}$ in Figure IV.17(j).

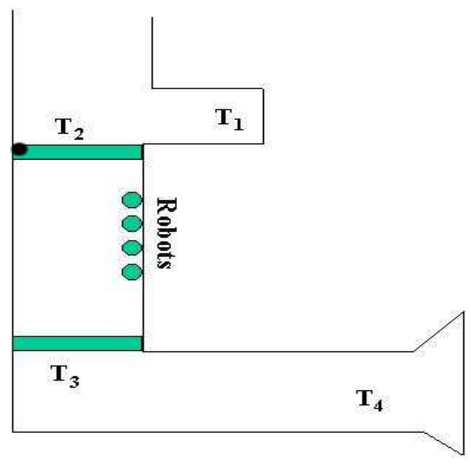


Figure IV.18: The urban indoor task environment (Real Robot Task).

Real Robot Experiments

Two real robot experiments were performed, in the first experiment a building corridor was mapped and boxes were placed, as in the simulation environment as shown in Figure IV.18. The environment had two rooms to be patrolled that were connected by a corridor and two boxes blocking each room. The task precedence graph is provided in Figure IV.19. Four robots, two patrol robots and two pusher robots were employed. The coalitions were formed to perform each task as it became eligible for execution, as demonstrated in Figures IV.20(a) - IV.20(f). Figure IV.20(a) shows the robots at their starting positions. Initially two pusher robots coalesce to push a box and unblock a room (task T_3 performed in Figure IV.20(b)), which prompts the patrol robots to coalesce and explore the unblocked room (task T_4 performed in Figures IV.20(c) and IV.20(d)). The pushers meanwhile unblock a second room (task T_2 performed in Figure IV.20(e)) and the patrol robots then visit and cover the second unblocked room (task T_1 performed in Figure IV.20(f)). All robots had a map of the environment and utilized a Monte-Carlo localization algorithm to determine their individual positions in the environment.

The second experiment involved forming coalitions to perform three box-pushing tasks. The objective was to manipulate the boxes so that they form a T-shape. Figures IV.21(a)-IV.21(b) show the initial and desired final configuration of the boxes. The third task was

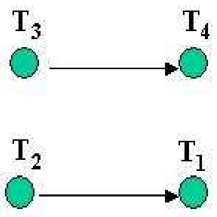
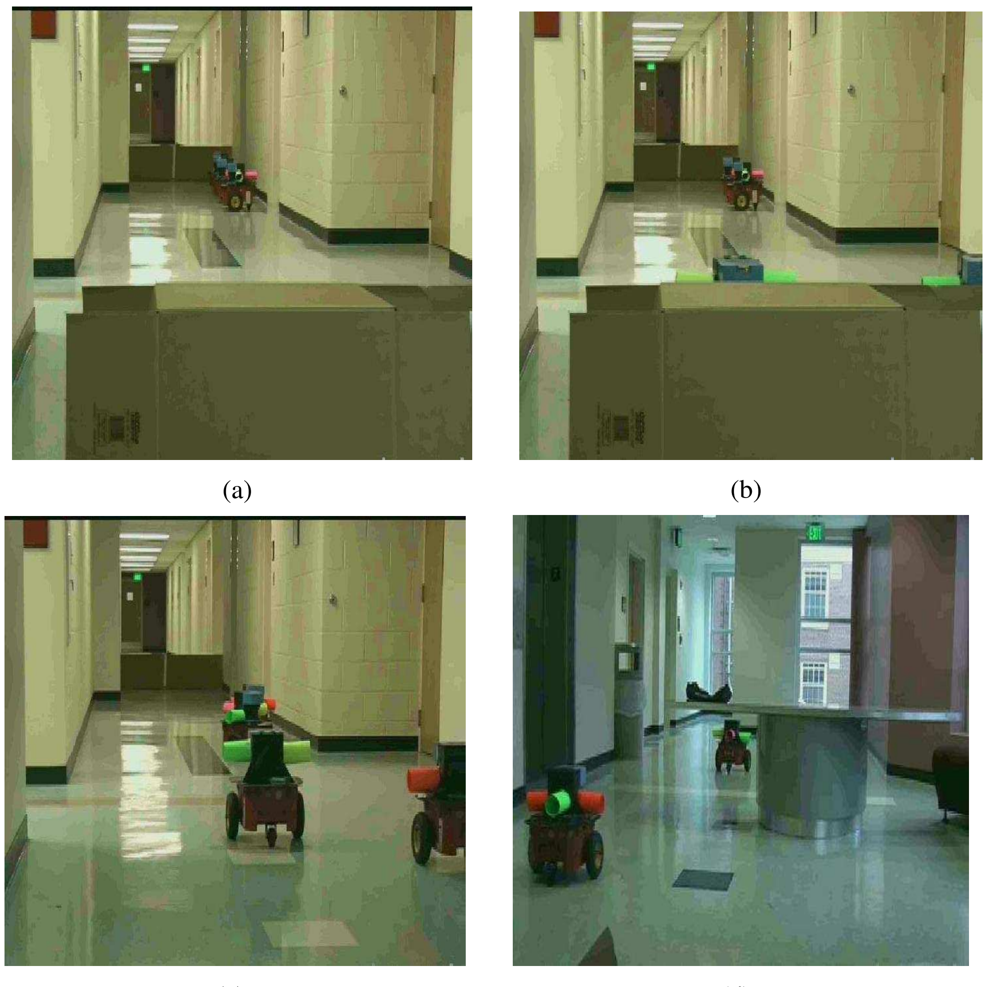


Figure IV.19: Precedence order graph for the task environment.

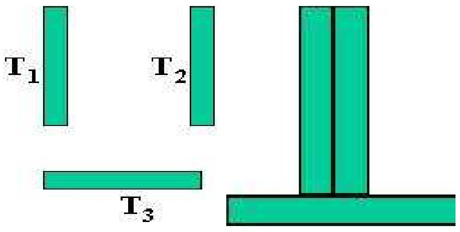


(e) (f)

Figure IV.20: (a) Robots at staring position. (b) Pushers coalesce to push a box. (c) Patroller robots coalesce to explore first unblocked room. (d) Pushers push a block to unblock second room. (e) Coalition of robots patrolling a room. (f) Patrollers visit second unblocked room.

dependent on the successful execution of the first two independent tasks as shown in Figure IV.22. Figure IV.23(a) shows the robots at their starting positions. Figure IV.23(b) shows the two tasks T_1 and T_2 being performed by two coalitions of pushers. Figures IV.23(c) shows a coalition navigating to perform T_3. Finally, T_3 is being performed by the same coalition that performed task T_1 in Figure IV.23(d).

Robotic domains frequently involve temporal ordering between tasks, the above experiments demonstrate that the algorithm can be applied to form overlapping coalitions to perform these tasks on the fly, as they become eligible for execution.



(a) (b)

Figure IV.21: Initial (a) and final (b) configuration of boxes to form a T-shape (Real Robot Task).

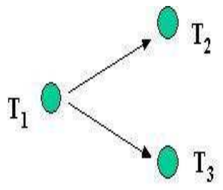


Figure IV.22: Precedence order graph for the task environment.

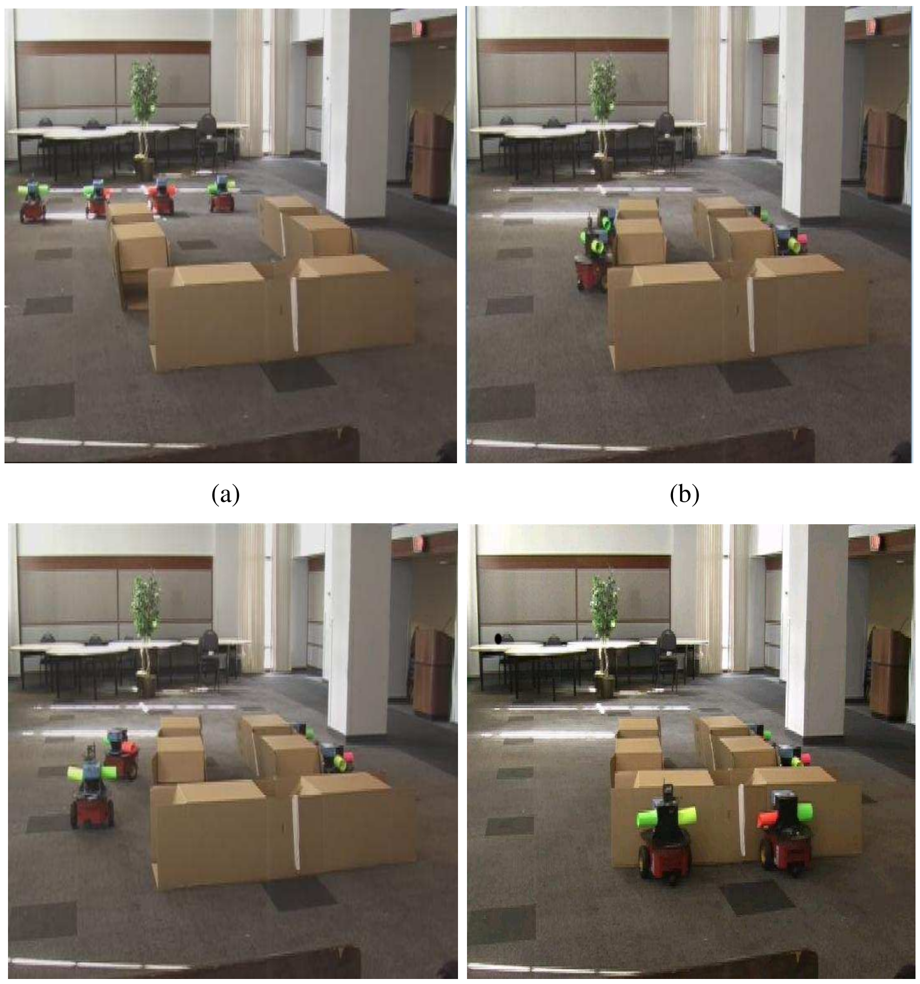


Figure IV.23: (a) Robots at starting positions. (b) Robots coalesce to perform the two independent box-pushing tasks. (c) Two robots then form a coalition to perform the dependent box-pushing task. (d) Dependent task performed.

IV.6 Discussion

The level of imbalance has important implications with regard to a coalition's level of fault tolerance. The effect of incorporating the FTC on the coalition formation was demonstrated in Sections IV.3 and IV.4. Three different tasks were defined and tested both in simulation and with real robots in order to validate the algorithm with a large number of robots. The tasks required different levels of coupling and each task required a different methodology for task execution. The results in Section IV.5.1 demonstrate that the algorithm operates independent of the nature of the tasks or task methodology. The experiments in Section IV.5.2 demonstrate that the algorithm is able to simultaneously allocate different types of tasks. Finally, IV.5.3 shows that the algorithm may be extended to form overlapping coalitions to perform precedence ordered tasks.

IV.7 Summary

Finding the optimal multi-robot coalition for a task is an intractable problem. This work shows that, with certain modifications, coalition formation algorithms provided in the multi-agent domain can be applied to the multi-robot domain. Initial experiments were conducted in simulation, however the effect of distributing the algorithm over a number of machines and its scalability could only be ascertained with real robot experiments. Real world issues like obstacle avoidance, battery power, localization accuracy only presented themselves in real world scenarios.

The coalition imbalance and its impact on the coalition's fault tolerance was demonstrated. Metrics for measuring balance and the fault tolerance of a coalition were evaluated. The algorithm was then demonstrated to work in simulation and on a set of Pioneer-DX robots with a diverse set of tasks. Finally, the extended algorithm that forms coalitions for precedence ordered task domains was demonstrated with a set of real world and simulated tasks.

CHAPTER V

BALANCE AND TEAM PERFORMANCE

Chapter III, Section III.2.3 introduced the notion of coalition imbalance and its implications with regard to the formation of fault tolerant coalitions. This chapter provides a deeper exploration of the concept of coalition imbalance and identifies a relationship between the imbalance level of a multi-robot team and team performance. Experiments were conducted with simulated multi-robot soccer and foraging teams to demonstrate that teams lying at the extremities of the performance spectrum tend to exhibit a higher level of balance. Later sections of the chapter describe experiments that were conducted to demonstrate how balance information may be utilized to improve overall team performance.

V.1 Introduction

As the scope and complexity of modern task demands exceed the capability of individuals to perform them, teams are emerging to shoulder the burgeoning requirements. Accordingly, researchers have striven to understand and enhance agent performance in team settings. Teamwork with human teams is a well studied topic and the last half century has produced many theories that encompass different teamwork perspectives (Paris et al., 2000; Baker and Salas, 1992; Ilgen, 1999; Kleinman and Serfaty, 1989). Although team theories began as descriptive efforts, many have evolved over time to provide more normative guidelines for improving teams. It is now well accepted that to understand effective team performance or 'teamwork' one must understand how groups of individuals function to produce effectual synchronized output, rather than just summed or aggregated responses (Steiner, 1972; Hackman, 1983; Nivea et al., 1978; Fleishman and Zaccaro, 1992).

The same principles may also be extended to multi-robot teams, i.e. it is important to understand teamwork and team formation from an individual's perspective to generate effective robot teams. This chapter analyzes one such aspect of multi-robot teams, namely the notion of *balance*. The notion of *balance* in a multi-agent team refers to the variance of individual contributions by team members towards the completion of the joint team task. A higher balance implies that the team members are contributing more evenly toward the joint team task.

Although balance is a very recent concept in multi-robot coalition formation (Vig and Adams, 2005), balance between teams has been previously studied in sports economics (Fort and Maxcy, 2003) for the purpose of professional league formation. The motivation behind Fort and Maxcy's work was to preserve the competitive edge of a professional soccer or baseball league in order to retain spectator interest and maintain ticket sales. However, the idea of maintaining balance within a team is to the best of our knowledge a relatively unexplored domain. A question often asked of human sports teams is: *do teams that are cohesive and balanced perform better than teams that have a few outstanding players and is otherwise highly imbalanced*? Quantifying human player capabilities (height, stamina, strength, skill, etc.) is highly subjective and there are inherent difficulties associated with measuring individual contributions of human players. Therefore, conducting a thorough investigation of imbalance using human teams is impractical. Additionally variables such as playing conditions, injuries, and motivations make it difficult to acquire consistent, reliable data that is necessary in order to analyze the effects of imbalance.

Multiple robot teams in contrast provide an excellent platform for research in this area because robot teams offer a domain where these variables can be controlled to a greater degree. Robot soccer teams are generally comprised of players that are identical in their physical attributes, something impossible to attain with human teams. Also, robot teams can play each other repeatedly under identical conditions without the risk of injury or fatigue, which facilitates the acquisition of reliable data for statistical analysis. One contribution of this chapter is a technique for quantifying the importance of individual robots in domains where importance is not directly measurable such as multi-robot soccer.

It should be mentioned that multi-robot teams must deal with a variety of real world constraints that make their analysis more complex. Robots often encounter partial and complete robot failures, thus the procedure for multi-robot team formation must attempt to take probability of failure into account and must favor fault tolerant teams. This chapter discusses the implications of balance with regard to fault tolerance.

This chapter also investigates the impact of coalition imbalance on the performance of multi-robot soccer and foraging teams in an effort to better understand the relationship between coalition imbalance and performance. Experimental results indicate teams lying at the extremities of the performance spectrum tend to be more balanced relative to teams in the middle of the performance spectrum. In addition, this chapter investigates the possibility of improving team performance by utilizing balance information. Subsequently, it was found that by improving the contributions of under-performing agents, the overall team performance improved significantly, in most cases. Further experiments were conducted in the multi-robot foraging environment in order to study the effect of imbalance in a loosely coupled task domain.

The remainder of the chapter is organized as follows: Section V.2 investigates the relationship between imbalance and performance of a multi-robot team and describes the method used for balance quantification, Section V.3 outlines the experimental design, and Section V.4 discusses the obtained results. Section V.5 provides a discussion and concluding remarks.

V.2 Imbalance and Performance

Balch (1998) devised the simple social entropy metric for the measurement of diversity. Simple social entropy was designed by applying Shannon's (Shannnon, 1949) information entropy to the measurement of diversity in robot teams. Balch further analyzed the impact of diversity on performance for both multi-robot soccer and multi-robot foraging and obtained diverging results for both tasks. While homogeneous teams were found to yield the best performance for multi-robot foraging, heterogeneous teams exhibited relatively better performance in the multi-robot soccer domain.

This section investigates balance as a common factor in high performance teams for both the soccer and foraging domains. Thus far in this dissertation balance has been studied from a purely fault tolerance perspective. This section investigates the correlation between balance and the performance of a multi-robot team. The experiments supporting this work were designed to utilize Balch's experimental framework for both heterogeneous, tightly coupled soccer tasks and homogeneous, loosely coupled foraging tasks.

V.2.1 Multi-Robot Soccer Environment

Balch's multi-robot soccer experiments employed Q-learning (Sutton and Barto, 1998) in order to teach the robots to play soccer. The learning approach incorporated a touch-based reward function. According to this function, each robot was rewarded based on how recently the robot touched the ball prior to an event (goal scored for or against the robot's team). Formally, the reward function is given by:

$$R_{touch}(t) = \begin{cases} \gamma_d^{t_{touch}} \text{ if the team scores at } t\text{-}1. \\ -\gamma_d^{t_{touch}} \text{ if the opponent scores at } t\text{-}1. \\ 0 \text{ otherwise.} \end{cases}$$

Where t_{touch} is the time in milliseconds since the agent last touched the ball. γ_d is a parameter set to a value between 0 and 1 that indicates how quickly a potential reward should decay after the ball is touched. Note that if $\gamma_d = 1$, all robots in a team receive equal reinforcement (= 1 or -1) each time a goal is scored.

Utilizing this reward function, Balch ran experiments varying the values of γ_d and discovered a positive correlation between γ_d and performance. Balch also demonstrated that homogeneous soccer teams do not perform as well as heterogeneous soccer teams. However, Balch found that there was no correlation between γ_d and the level of heterogeneity. Balch was able to conclude from these results that while a heterogeneous team will outperform a homogeneous team, the degree of heterogeneity, as measured using social entropy does not correlate with team performance.

This Chapter includes results from experiments using a similar experimental setup, only instead of measuring the social entropy, balance was the quantity measured for each team as γ_d varied across a range of values. However, measuring balance entails the acquisition of knowledge regarding the individual contributions of all team members to the overall team objective. The procedure followed to collect the required data is described in the remainder of this Section.

Measuring Imbalance

Chapter I, Section III.2.3 discussed the role of balance in the formation of more fault tolerant coalitions, and introduced the FTC as a metric that subsumed elements of both balance and coalition size. Since the size of a multi-robot team is constant, this Section concerns itself only with balance and how to measure balance in a domain such as multi-robot soccer.

The balance coefficient metric works well when the $taskvalue$ can be reduced to a scalar value, however it does suffer from some limitations. Determining an exact $taskvalue$ is not possible for all task domains. Also, the balance coefficient does not account for negative contributions that correspond to cases when a robot detracts from the overall task performance. It is often possible to have negative contributions in multi-robot soccer. Since the objective is to compare imbalance levels across different teams, the balance coefficient is inappropriate for quantifying balance in this particular domain. Also, the robots have identical sensor and actuator capabilities and hence the differences in individual contributions are directly related to the policies that each robot follows. Taking these issues into consideration, a different technique for quantifying balance was devised. In order to measure the individual contribution of a particular agent towards the overall team performance, the team performance without the agent was measured. The drop (or potential gain) in performance when a robot was excluded from participation in a task provided a reasonable estimate of the relative contribution from that robot.

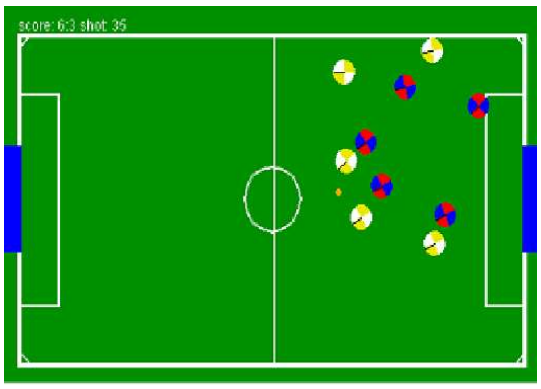


Figure V.1: A sample five against five Javabots simulation (Balch and Ram, 1998). The adaptive team (dark colored robots) played the control team (light colored robots).

Adapting this technique to the multi-robot soccer domain required the adaptive soccer team to converge to a policy via Q-learning. Then the adaptive team played against a fixed control team. Fig. V.1 provides a sample simulation of the full adaptive team playing the fixed control team.

After the performance of the adaptive team was recorded, it was important to obtain an estimate of individual player contributions to the overall performance. The individual contribution was determined by removing each robot team member one at a time and recording the resulting team performance. The relative contribution of a particular team member A_i was obtained by subtracting the performance of the four member team (without A_i) P_{full-A_i} from the performance of the full five member team P_{full}. Fig. V.2 shows a sample simulation where four members of the adaptive team play the full five member control team.

The contribution of an individual agent A_i was normalized across the range of the experiment by calculating the fractional change in performance when A_i was removed with respect to the maximum performance exhibited by all teams when any agent was removed. Hence the individual contribution $C(i)$ of member A_i was evaluated as follows:

$$C(i) = \frac{(P_{full} - P_{full-A_i})}{MAX(P_{full}, P_{full-A_1}, P_{full-A_2}, \ldots, P_{full-A_5})} \tag{V.1}$$

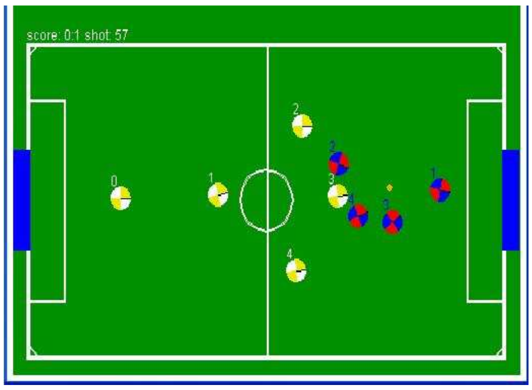


Figure V.2: A sample four against five simulation. The four robot adaptive team (dark colored robots) played the five robot control team (light colored robots).

Finally, imbalance is quantified as the standard deviation in the contributions for all five members.

$$Imbalance = \frac{\sqrt{\Sigma(C(i) - \overline{C})^2}}{5-1} \tag{V.2}$$

where $\overline{C}$ is the mean contribution of all five team members.

V.2.2 Multi-Robot Foraging

Balch utilized three different reward functions to enable the robots to learn to forage using Q-learning. The first was a Local reinforcement function that yielded a reward to a single robot upon delivery of a puck by that robot. The second was a Global reinforcement function that rewarded all robots whenever any robot delivered a puck. The final learning strategy utilized Shaped reinforcement (Matarić, 1997) which leverages domain knowledge in order to accelerate learning. This work utilizes the same reinforcement strategies in order to measure imbalance. Teams converged to policies using all three learning strategies and performance and balance levels were recorded over repeated trials while varying the number of robots.

Measuring Balance

Balch defined and formalized the notion of behavioral distance to measure the diversity of a multi-robot foraging team. However, the measure that was utilized suffered from a limitation in that it treated all different behaviors as being equally dissimilar. Since we are considering the notion of balance we can avoid measuring the behavioral distance and directly measure the individual contributions of the robots by recording the number of pucks each robot manages to collect.

The balance coefficient (Equation (III.3)) was utilized to measure balance in this set of experiments. The *taskvalue* was equal to the number of pucks collected by the entire team. This was also utilized as the net performance measure of the team. The individual contributions were the number of pucks collected by each robot. Thus the balance coefficient was calculated as follows:

$$BC = \frac{q_1 \times q_2 \times \dots q_n}{\left[\frac{\sum_{i=1}^{n} q_i}{n}\right]^n}. \tag{V.3}$$

where q_i represents the number of pucks collected by robot R_i.

Just as a variety of metrics exist for quantifying team diversity (Balch, 1998) and different metrics are appropriate for different domains, the above method should not be assumed to be a universal technique that is optimal for all possible scenarios. There may exist more appropriate techniques for quantifying balance in different multi-agent (robot) domains such as box-pushing, multi-target tracking, etc. However, no matter which technique is employed, the underlying objective is the same, that is to measure the level of balance and to quantify the disparity or lopsidedness of individual contributions towards the completion of a multi-agent task. Although balance appears similar to heterogeneity, it is in fact a different concept. While heterogeneity is a more static concept that deals with the diversity of individual agents within a team, balance is somewhat more dynamic and depends purely on what the agents contribute or how well they perform during the execution of a particular task.

V.3 The Experimental Setup

This section explains the experimental design that led to the results reported in Section V.4. The behavioral assemblage utilized for these experiments was similar to that outlined by Balch (1998).

V.3.1 Soccer Environment

Each robot could choose from amongst the following three behaviors in a given state:

1. **move_to_ball_behavior** (**mtb**) : The robot moves directly towards the ball. A collision with the ball will propel it away from the robot.

2. **get_behind_ball_behavior** (**gbb**) : The robot moves to a position between the ball and the defended goal while dodging the ball to avoid moving it in the wrong direction.

3. **move_to_backfield_behavior** (**mtbf**) : The robot moves to the back third of the field while simultaneously being attracted to the ball. The robot will kick the ball if it is within range.

Each robot in a team could be in one of two states:

1. **behind_ball** (**bb**) : Indicates that the robot was currently behind the ball.

2. **not_behind_ball** (**nbb**) : Indicates that the robot was in front of the ball.

As stated in Section V.2, the robots learnt to play soccer using a Q-learning approach based on the touch-based reward function devised by Balch (1998). This function rewarded each robot based on how recently the robot touched the ball prior to an event (goal scored for or against the robot's team).

The simulated robots learned to play soccer against a team with a fixed control policy. The same behavioral assemblage was utilized for both the fixed control team and the adaptive team to ensure that the teams were evenly matched.

Table V.1: Control Team Goalie Policy.

perceptual feature	assemblage		
	mtb	gbb	mtbf
not behind ball	0	1	0
behind ball	0	0	1

Table V.2: Control Team Forward Policy.

perceptual feature	assemblage		
	mtb	gbb	mtbf
not behind ball	0	1	0
behind ball	1	0	0

The fixed control team policy was designed to ensure at least one defensive robot (goalie) guarded the fixed control team goal and four forward robots attacked the opponent's goal. The strategy adopted by the goalie was to be positioned behind the ball if it found itself ahead of the ball and to move to the backfield if it was behind the ball. The policy for the fixed control team goalie is illustrated in Table V.1. The strategy for the team forwards was altered to move behind the ball if they are not behind the ball and to move to the ball if behind the ball. The policies for the fixed control team forwards is illustrated in Table V.2.

The JavaBots (Balch and Ram, 1998) robot soccer simulation software (as seen in Fig. V.1 and Fig. V.2) was utilized to perform all experiments. The soccer teams consisted of five simulated robots. The behaviors and motor schemas were designed using the Clay architecture (Balch, 1997). A game was terminated when one of the teams scored 20 goals. The performance metric was simply the difference in the number goals scored by the adaptive team and the number of goals scored by the fixed control team shifted by 20 units, as shown in Equation (V.4). This 20 unit shift ensured a positive range for performance between 0 and 40. This was necessary because it allowed for easy calculation of fractional standard deviation.

$$P_f = Score_{Adaptive} - Score_{Control} + 20 \tag{V.4}$$

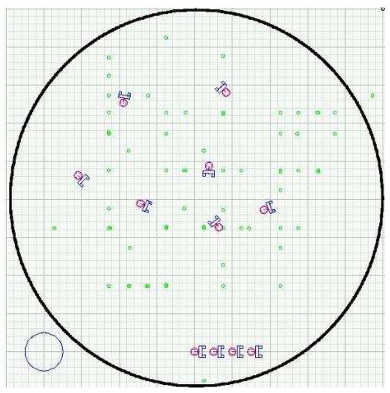


Figure V.3: A foraging simulation involving a team of four robots retrieving pucks to their starting position.

The experimental objective was to observe the relationship between the discount factor γ_d and performance. The experiments were conducted to note the change in performance as γ_d varied from 0.1 to 1.0. Ten trials were run for each γ_d value and the results for each trial were recorded. Each trial required a team to converge to a policy and then play 100 soccer games against the fixed control team. The average performance was recorded over all 100 games.

Information from this experiment was then further utilized to obtain multi-robot teams with performance measures within a certain range and a plot of performance vs. balance was derived (see Section V.4). Finally, a set of experiments were conducted where poorly performing team members were replaced with efficiently contributing members and the resulting improvement was recorded.

V.3.2 Foraging Environment

The foraging simulations used the Player/Stage environment (Gerkey et al., 2001) and were conducted to measure balance in a multi-robot foraging environment. Fig. V.3 shows the simulated foraging environment. The following behaviors were utilized for these experiments:

1. **Wander**: The robot navigates randomly in the environment while avoiding obstacles.

2. **Homing**: The robot returns to its starting location and if it has a puck, deposits the puck at the location.

3. **Dispersion**: The robot tries to move away from an intruder.

4. **Resting**: Stop moving and recharge battery if at home location.

The following conditions are utilized in order to define a state for an individual robot:

1. **Near_Intruder**: True if the robot is too close to another robot.

2. **Night_Time**: Periodically true for 20 seconds after every 3 minutes.

3. **At_Home**: True if the robot is at it's starting location.

4. **Have_Puck**: True if the robot currently has a puck in its gripper.

Multiple simulations with varying numbers of robots were conducted with a random distribution of pucks in each simulation. Simulations were stopped after fifteen minutes and the number of pucks collected by each robot was recorded. For each set of robots the balance was calculated as shown in Section V.4.2 and team performance was recorded and analyzed.

V.4 Experiments

This section provides the results of the experiments outlined in Section V.3. Section V.4.1 outlines the results depicting the relationship between balance and performance in the two domains of multi-robot foraging and multi-robot soccer. Section V.4.3 provides results that indicate the utility of balance information for a particular soccer team and depicts the correlation between imbalance and possible improvement in team performance.

V.4.1 Relationship between Balance and Performance: Multi-Robot Soccer

As mentioned in Section V.3, the adaptive team learned to play against the fixed control team utilizing different values of the discount factor γ_d. Each team then played 100 soccer games against the fixed control team and the average performance was recorded. Table V.3 shows the results from one such trial recording the number of wins for the adaptive team ($\#A_{wins}$) and the control team ($\#C_{wins}$), the average scores for both teams ($\overline{A}_{Score}$ and $\overline{C}_{Score}$), the average goal difference between the teams (GDiff), and the performance of the adaptive team (Perf). Similar data exists for the simulation run where four members of the adaptive team played against five members on the fixed control team.

Table V.3: Example results for a five on five simulation for varying values of γ_d.

γ_d	$\#A_{wins}$	$\#C_{wins}$	$\overline{A}_{Score}$	$\overline{C}_{Score}$	GDiff	Perf
0.1	0	100	8.8	20	-11.2	8.8
0.2	0	100	13.4	20	-6.6	13.4
0.3	0	100	16.4	20	-3.6	16.4
0.4	100	0	20	8.5	11.5	31.5
0.5	100	0	20	9.6	10.4	30.4
0.6	100	0	20	8.4	11.6	31.6
0.7	58	42	18.9	17.4	1.5	21.5
0.8	100	0	20	9.7	11.3	31.3
0.9	100	0	20	7.3	12.7	32.7
1.0	100	0	20	7.1	12.9	32.9

Fig. V.4 shows the variation in performance of the adaptive team as γ_d varied from 0.1 to 1.0. Ten trials were completed for each value of γ_d. By and large the results agree with those obtained by Balch (1998), i.e. the performance measure of a team increases with an increase in γ_d. There is a clear positive correlation between γ_d and performance as the Pearsons coefficient of correlation between γ_d and performance was found to be significant, $r(98) = .818,\ p < 0.01$.

The results in Fig. V.4 provided important information regarding the relationship between team performance and the value of the discount factor γ_d. Utilizing this information the range of the possible performance values was partitioned into bins, with each bin spanning five units on the performance scale. Each bin contained exactly ten teams in order to ensure a fair comparison across the bins. The experiments were repeated with appropriate γ_d values if there were fewer than ten teams in a bin, until ten teams were obtained. If

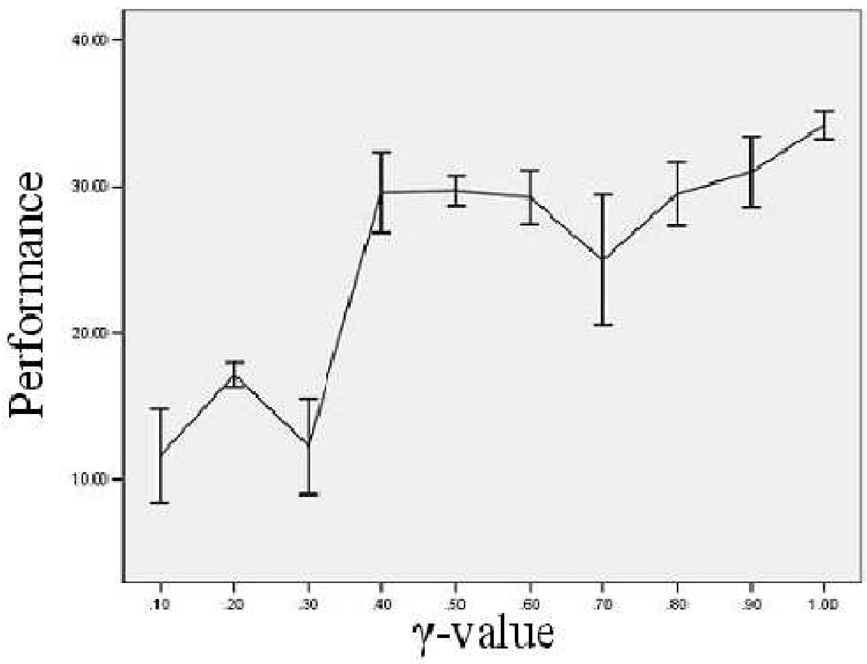


Figure V.4: Performance vs. Discount factor with 95% confidence intervals.

there were more than ten teams in a bin, then the average imbalance level of all the teams in the bin was calculated. Redundant teams with the highest deviation from the mean were discarded. Once each bin contained exactly ten teams, all teams played 100 games against the fixed control team and the results were recorded. Subsequently, each team's balance was recorded using the procedure outlined in Section V.3.

Fig. V.5 shows the scatter plot of Imbalance against Performance for all teams. Each circle represents the average performance of the adaptive team against the control team over a single trial comprised of 100 soccer games. The plot indicates a relatively high level of balance for teams whose performance lies at the extreme ends of the performance spectrum. The extremities of the performance spectrum represent teams with very high and very low performance levels but these teams tend to be more balanced. Finally, the bar graph depicting the average level of imbalance for teams lying in a particular performance range was calculated, Fig. V.6 provides the results. Each bar spans an equal performance range (5 units) and represents the average imbalance level of all ten teams lying within the

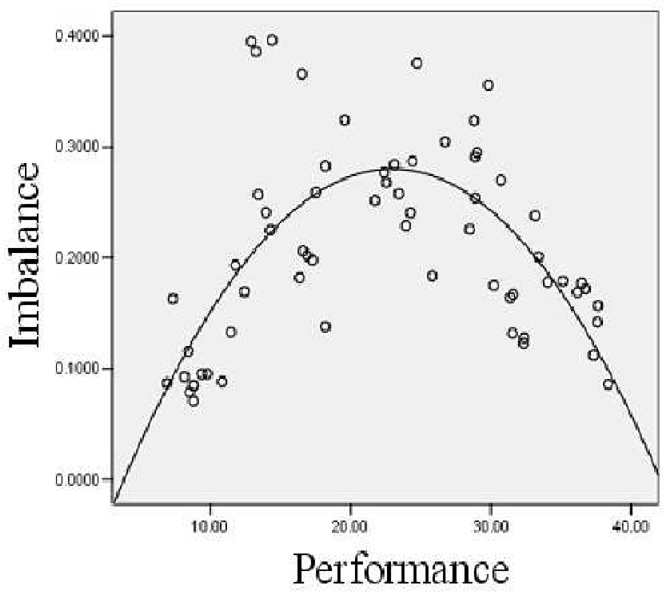


Figure V.5: Scatter plot of Imbalance vs. Performance.

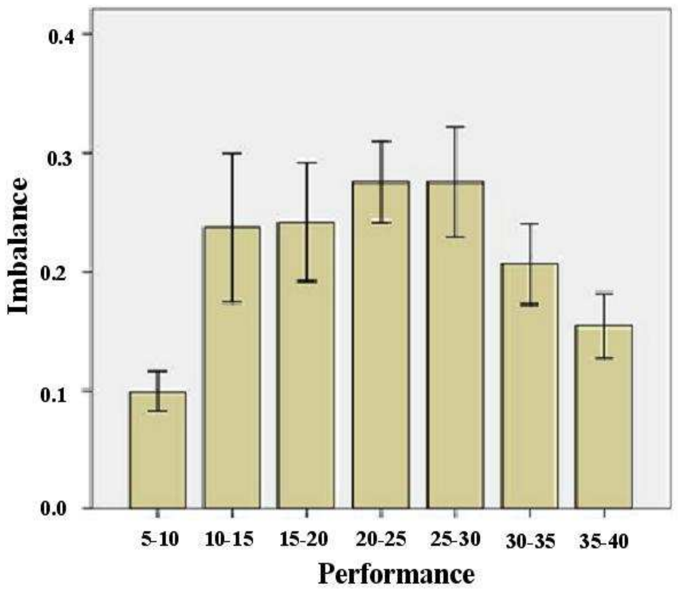


Figure V.6: Imbalance vs. Performance bar graph for adaptive team vs. control team with 95% confidence intervals.

Table V.4: Students t-test (df = 18) for mean difference between imbalance of 5-10 teams and other teams in different performance ranges (Control Team).

Performance Range	10-15	15-20	20-25	25-30	30-35	35-40
t-value (t)	-4.12	-6.06	-11	-9.75	-4.6	-3.63
Probability (p)	$< .01$	$< .01$	$< .01$	$< .01$	$< .01$	$< .01$

Table V.5: Students t-test (df = 18) for mean difference between imbalance of 35-40 teams and other teams in different performance ranges (Control Team).

Performance Range	5-10	10-15	15-20	20-25	25-30	30-35
t-value (t)	3.63	2.96	2.08	7.46	6.51	1.3
Probability (p)	$< .01$	.01	.05	$< .01$	$< .01$	.21

performance range.

Again, the bars in Fig. V.6 indicate a relatively high level of balance for teams whose performance lies in the 5-10 range and teams in the 35-40 range. Table V.4 demonstrates that the difference between the mean imbalance for the 0-5 teams and all the other team ranges is statistically significant as found using the Students t-test. Table V.5 similarly demonstrates that the difference between the imbalance of the 35-40 teams is also statistically lower than the other team ranges (except the 30-35 range teams).

The reason for the 0-5 teams' poor performance is that the teams with the worst performance levels are comprised of members that are equally ineffective at playing soccer. The resulting team is balanced but each member makes a very minor contribution towards task completion (playing soccer). As the performance improves, some members converge to more effective policies and begin contributing to the task in greater measure. The team has optimum performance when all the members contribute effectively. Thus the teams with the best performance also tend to have a higher level of balance on average than the intermediate teams.

After the experiments with the control team, it was necessary to study the reproducibility of the results against other teams. This was accomplished by repeating the experiments with two other, non-behavior based soccer teams, namely the DTeam and Kechze soccer

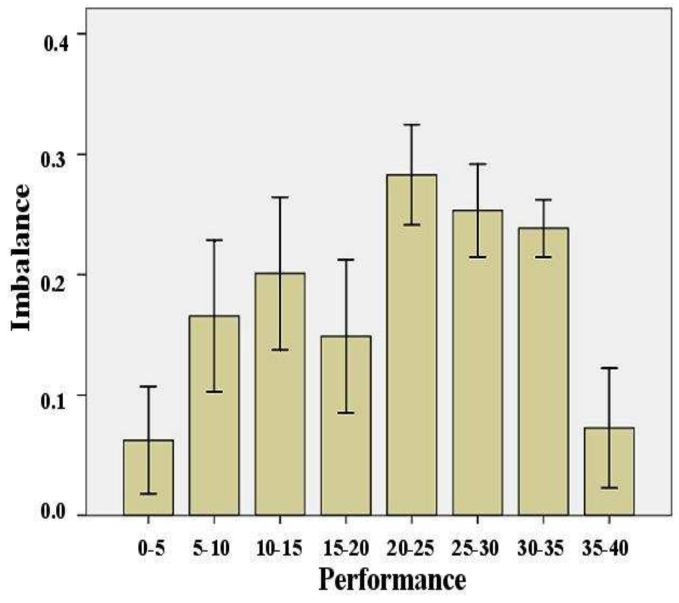


Figure V.7: Imbalance vs. Performance bar graph for adaptive team vs. DTeam with 95% confidence intervals.

Table V.6: Students t-test (df = 18) for mean difference between imbalance of 0-5 teams and other teams in different performance ranges for the DTeam experiment.

Performance Range	5-10	10-15	15-20	20-25	25-30	30-35	35-40
t-value (t)	-2.73	-3.17	-1.90	-6.99	-6.13	-6.33	-.359
Probability (p)	.01	.01	.07	$< .01$	$< .01$	$< .01$	.72

Table V.7: Students t-test (df = 18) for mean difference between imbalance of 35-40 teams and other teams in different performance ranges for the DTeam experiment.

Performance Range	0-5	5-10	10-15	15-20	20-25	25-30	30-35
t-value (t)	.359	2.62	2.96	2.08	7.46	6.51	6.97
Probability (p)	.72	.02	.01	.05	$< .01$	$< .01$	$< .01$

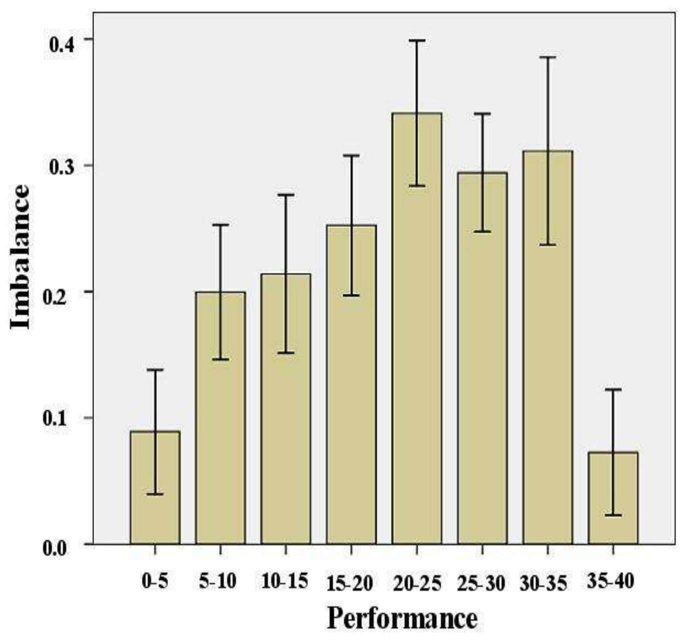


Figure V.8: Imbalance vs. Performance bar graph for adaptive team vs. Ketchze team with 95% confidence intervals.

teams that are part of the Javabots package. When selecting teams it was important to choose teams with comparable performance, if a team chosen was far superior or inferior to the learning teams then it would be difficult to span the entire performance range for the learning teams. Fig V.7 and V.8 shows the variation of imbalance with performance of the learned team when playing against the DTeam and Kechze soccer teams. In general the

Table V.8: Students t-test (df = 18) for mean difference between imbalance of 0-5 teams and other teams in different performance ranges for the Kechze team experiment.

Performance Range	5-10	10-15	15-20	20-25	25-30	30-35	35-40
t-value (t)	-3.56	-3.12	-3.58	-9.26	-7.26	-6.83	-.813
Probability (p)	$< .01$	.01	$< .01$	$< .01$	$< .01$	$< .01$	.43

Table V.9: Students t-test (df=18) for mean difference between imbalance of 35-40 teams and other teams in different performance ranges for the Kechze team experiment.

Performance Range	0-5	5-10	10-15	15-20	20-25	25-30	30-35
t-value (t)	.813	6.78	6.43	7.4	12.4	10.1	8.45
Probability (p)	.43	$< .01$	$< .01$	$< .01$	$< .01$	$< .01$	$< .01$

results agree with the results obtained against the control team, i.e. teams at the extremities of the performance spectrum tend to be more balanced than teams in the middle of the performance spectrum. Tables V.6 and V.7 show that when playing against the DTeam, the performance of the 0-5 and 35-40 range adaptive teams was statistically different from all other teams (except 15-20 range teams). Tables V.8 and V.9 show that when playing against the Kechze team, the performance of the 0-5 and 35-40 range adaptive teams was statistically different from all other teams.

V.4.2 Relationship between Balance and Performance: Foraging Experiments

As mentioned in Section V.3, three different reward functions were utilized to provide reinforcement to an adaptive multi-robot foraging team. The number of robots was varied and the mean performance and balance was recorded for each set of robots.

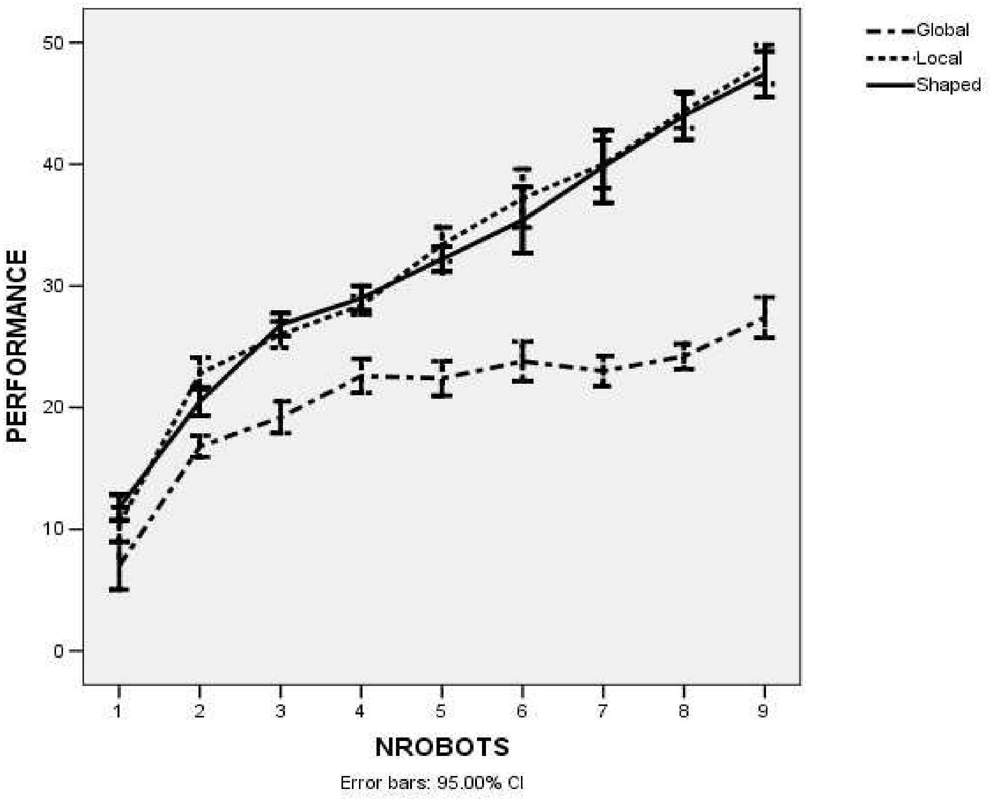


Figure V.9: Performance vs. Number of foraging robots with 95% confidence intervals.

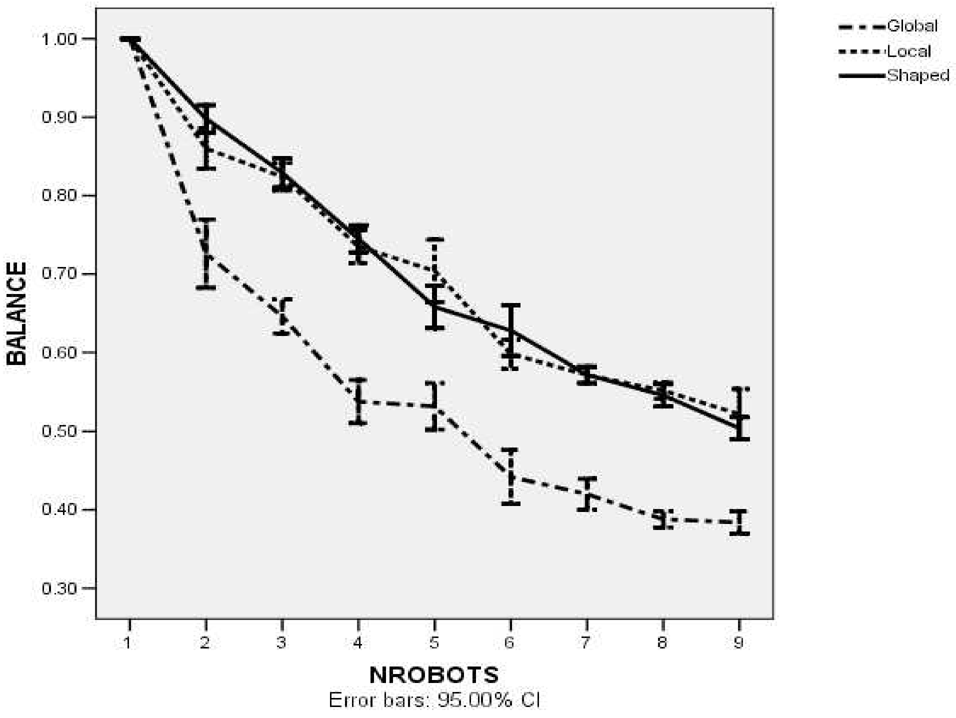


Figure V.10: Balance vs. Number of Robots with 95% confidence intervals.

Fig V.9 shows the variation in mean performance of the three learning strategies (Global, Shaped, Local) as the number of robots increased. Ten trials were conducted for each set of foraging robots. The Shaped and Local reinforcement strategies clearly outperform the Global reward strategy. This is because the Shaped and Local reinforcement policy rewarded the robots only when they accomplished a relevant task or subtask while the Global reinforcement rewarded a robot based on the potentially unrelated actions of other robots.

Fig V.10 shows the variation in balance of all teams with the different learning strategies. Again the Shaped and Local learning strategies result in teams that are more balanced than the Globally reinforced teams, even as the number of robots increased to nine.

It would be tempting to conclude from the above results that balanced teams always do better than imbalanced teams. Such an argument would be flawed because one can always construct a perfectly balanced team where the robots perform equally poorly. However, the results do suggest that teams that perform well tend to have a higher degree of balance.

V.4.3 Utilizing Balance Information to Improve Performance

If information regarding balance (i.e. the relative contributions made by individual members) is available, the question then becomes “can this information be utilized to improve team performance?” Human Factors researchers have previously examined the impact of individual differences in cognitive ability and personality characteristics on human team performance (Mohammed and Angell, 2003). A recent study by Brou et al. (2005) showed that a team’s weakest member had the most impact on the team’s performance as compared to the impact of any of the other team members. This result suggested that the performance of a multi-robot team could be most significantly improved if the performance of the weakest team member improved. If the weakest and strongest members of a multi-robot soccer team could be identified, then substituting the weakest robot with a copy of the strongest robot should result in a significant performance boost. Such a substitution may not necessarily create a more balanced team; however it should push the team towards a higher performance level.

In order to verify this hypothesis an additional analysis was performed based upon the various soccer teams developed for the experiment in Section V.4, that had converged to their respective policies. Once a team’s policy had stabilized, the agents that contributed most effectively (A_g) and least effectively (A_l) were identified. A new soccer team was created by substituting the policy of A_g into the policy for A_l. Intuitively, since a more efficient agent is taking the place of a less efficient one, the performance of the new team should be better than the original team. Figure V.11 provides the interpolated surface plot based on the data points (dots in the figure) representing the net improvement for a team with a given performance and imbalance level. The plot depicts how the improvement resulting from substitution varies with the dimensions of imbalance and initial performance. Figure V.11 shows the results obtained when the adaptive team played the Control team. Figures

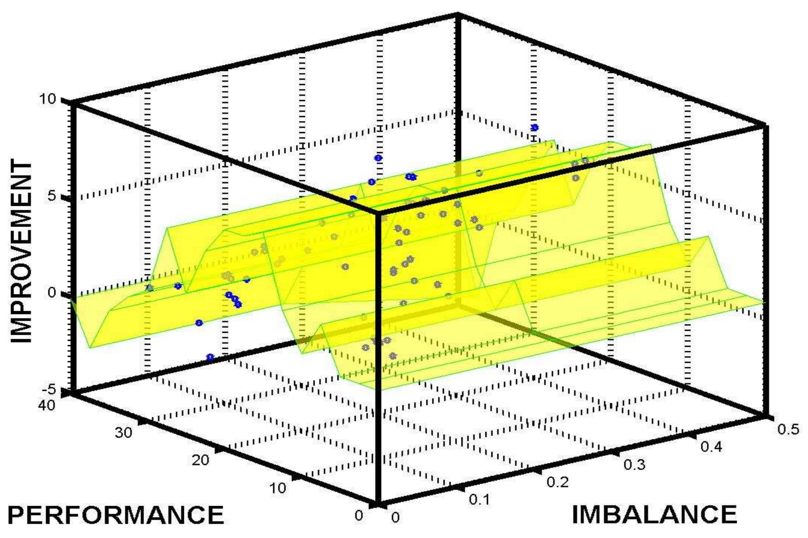


Figure V.11: Interpolated surface plot depicting the variation of the substitution induced performance improvement of the adaptive soccer team with initial performance and imbalance when the adaptive team played against the Control team.

V.12 and V.13 show similar plots for when the adaptive soccer team played the DTeam and Kechze teams.

The plots demonstrate that moderately performing soccer teams with a high level of imbalance exhibit a relatively higher level of improvement. Teams that are highly balanced (i.e. imbalance closer to zero) show a very minor improvement and occasionally even exhibit a drop in performance, especially at the higher end of the performance spectrum. This is because an imbalanced team implies a greater disparity between the contributions of the substituting agent, A_g and the substituted agent, A_l. Hence, for an imbalanced team, the potential for improvement due to the substitution is high. Conversely, for a balanced team, the potential for improvement is low because the both A_g and A_l are already providing a relatively equal contribution and the substitution is unlikely to yield a significant difference in performance.

A Pearsons coefficient of correlation was calculated between the imbalance and performance improvement to validate this hypothesis and the results are provided in Table V.10.

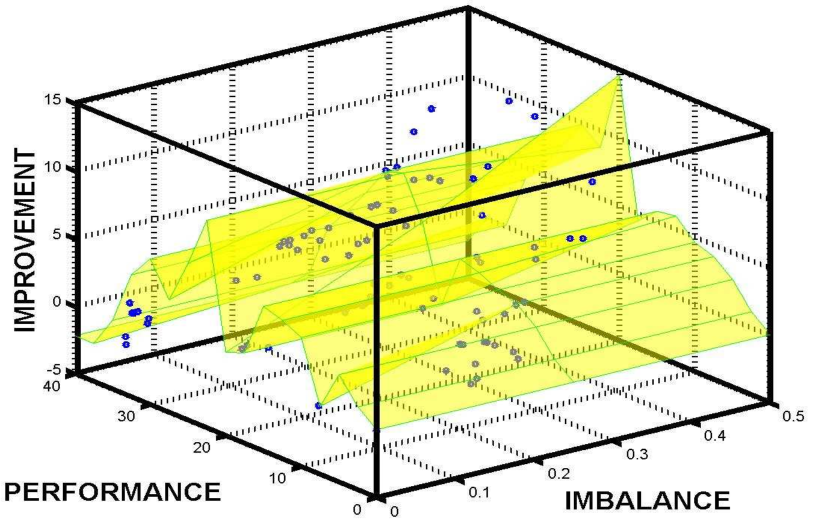


Figure V.12: Interpolated surface plot depicting the variation of the substitution induced performance improvement of the adaptive soccer team with initial performance and imbalance when the adaptive team played against the DTeam team.

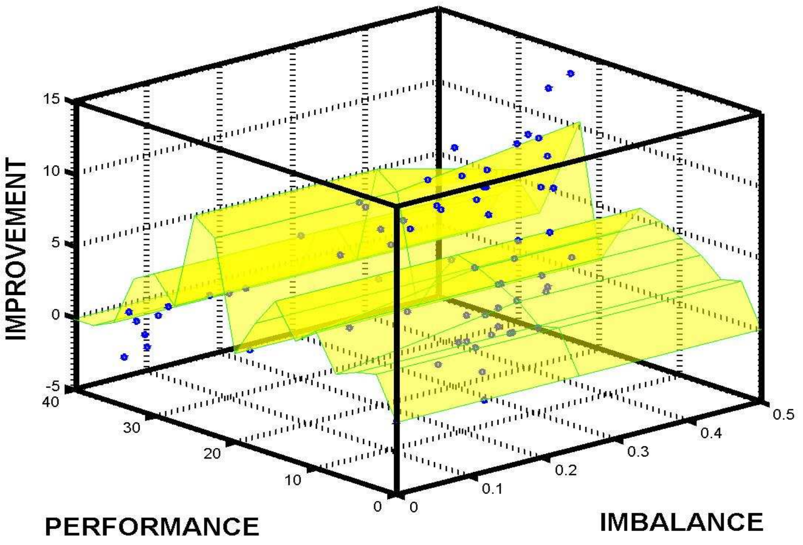


Figure V.13: Interpolated surface plot depicting the variation of the substitution induced performance improvement of the adaptive team with initial performance and imbalance when the adaptive soccer team played against the Kechze team.

All three teams exhibit a reasonably high degree of positive correlation between the two quantities. This result suggests that imbalanced teams may indeed exhibit higher performance improvement than balanced teams.

Table V.10: Pearsons Coefficient of Correlation between imbalance and improvement due to substitution.

Team	Pearsons Coefficient
Control	$r(68) = 0.616, p < 0.01$
DTeam	$r(78) = 0.686, p < 0.01$
Kechze	$r(78) = 0.567, p < 0.01$

V.5 Discussion

While the objective of Balch's experiments was to determine the effect of the different reward functions on team diversity, the endeavor of the above experiments was to ascertain a property that is common to high performance teams across different domains. The results indicate that balance holds promise for being such a property. The fact that balance information may be used to improve performance suggests the practical importance of determining balance.

Balance in multi-robot coalitions and its implications with respect to team performance is a relatively unexplored area of research. The results in this Chapter suggest a correlation between the level of balance in a multi-robot team and team performance where teams at the highest and lowest levels of the performance spectrum tend to be relatively balanced. While the relation between diversity and performance seems to be different for the loosely-coupled foraging and tightly-coupled soccer task, teams at the high end of the performance spectrum in both domains tend to exhibit high balance. Thus balance appears to have a more universal relationship with performance across different domains. Balancing a team may also be a useful concept for optimizing team performance, especially for highly imbalanced teams. Balance information enables us to determine which agents are performing poorly and replacing them generally results in a substantial improvement in team performance. However, this improvement is relatively more significant for imbalanced teams.

Observing that balance information may be utilized to improve performance of a team that has already learned to play soccer, the question then becomes, can this information be obtained in real-time? Can balance be directly incorporated into the reward function to ensure that the team members learn policies that result in balanced teams? This is an area that has been earmarked for future research. Another issue to be addressed is the effect of balancing in other domains, i.e. can the results of this Chapter be generalized to other multi-robot domains.

CHAPTER VI

MARKET-BASED COALITION FORMATION

Task allocation is an issue that every multi-robot system must address. Recent solutions to the task allocation problem propose an auction based approach wherein robots bid for tasks based on pre-defined cost functions for performing a task (Dias, 2004; Gerkey and Matarić, 2000). This Chapter presents RACHNA, a novel architecture for multi-robot task allocation based on a modified algorithm for the winner determination problem in multi-unit combinatorial auctions. A more generic utility based framework is proposed to accommodate different types of tasks and task environments. Experiments yield promising results demonstrating the system's superiority over simple task allocation techniques.

VI.1 The RACHNA System

A common feature of the market based systems discussed in Chapter I is that all these systems require the robots to bid on the tasks. The bidding process is central to determining the auction outcome. Therefore when dealing with complex tasks, the bidder should have a global view of the available resources. We propose a system, namely RACHNA[1], in which the bidding is reversed. The auction is performed by the tasks for the individual robot services. This allows for the bidding to be performed with the global information necessary for coalition formation.

As mentioned in Chapter I, there are some inherent differences between the multi-agent and multi-robot domains. One of the most prominent of these differences is the level of redundancy in multi-robot and software-agent capabilities. Whereas software-agents are simply code fragments programmed by individuals, robots are manufactured on a large scale. Therefore, robots are more likely to have greater redundancy in their sensor/actuator capabilities. Indeed, almost any modern day robotics facility would have a number of ro-

[1]Robot Allocation through Coalitions using Heterogeneous Negotiating Agents

bots with identical capabilities. To the best of our knowledge, RACHNA is the first system to leverage this redundancy in order to enable a more tractable formulation of the coalition formation problem. RACHNA achieves this through the formulation of the coalition formation as a multi-unit combinatorial auction. [2]

While the use of single good auctions allows the bidders to bid on only one good, combinatorial auctions permit bidding on combinations of goods. This work focuses on a particular type of combinatorial auction called multi-unit combinatorial auction.

Definition: The auctioneer has a set of items, *M = 1, 2,..., m* to sell. The auctioneer has some number of each item available: $U = \{u_1, u_2, ..., u_m\}, u_i \in Z+$. The buyers submit a set of bids, $B = \{B_1, B_2, ..., B_n\}$. A bid is a tuple $B_j = \langle (\gamma_j^1, ..., \gamma_j^m), p_j \rangle$, where $\gamma_j^k \geq 0$ is the number of units of item k that the bid requests, and p_j is the price. The **Binary Multi-Unit Combinatorial Auction Winner Determination Problem (BMUCAWDP)** is the problem of labeling the bids as winning or losing so as to maximize the auctioneers revenue under the constraint that each unit of an item can be allocated to at most one bidder:

$$max \ \sum p_j x_j \ s.t. \sum_{j=1}^{n} \gamma_j^i x_j \leq u_i, i = 1, 2, \ldots, m. \tag{VI.1}$$

The Multi-Robot Coalition Formation (MRCF) problem can be cast as a combinatorial auction with the bidders being represented by the tasks, the items as the different types of robots and the price the utility that each task has to offer. Unfortunately, the BMUCAWDP problem is inapproximable (Sandholm, 2002) however some empirically strong algorithms do exist (Leyton-Brown et al., 2000; Sandholm, 2002). It remains to be seen if such algorithms can be decentralized sufficiently to apply them beneficially to a multi-robot setting.

[2]There may be subtle unavoidable differences in robots with seemingly identical sensory capabilities due to wear and tear, loose wiring, sensor accuracy etc. These are ignored for the time being.

VI.1.1 The Architecture

We propose a system, namely RACHNA, in which the bidding is reversed. The auction is performed by the tasks for the individual robot services. This allows for the bidding to be performed with the global information necessary for coalition formation. There are two types of software agents that are involved in the task allocation:

1. **Service Agents:** The Service Agents are the mediator agents through which the tasks must bid for a service. RACHNA requires that each robot has a set of services or roles that it is capable of performing. The roles are determined by the individual sensor and behavioral capabilities resident on each robot. There is one service agent for each service type that a robot can provide. A service agent may communicate with any of the robots that provide the particular service to which the agent corresponds. For example, the foraging service agent may communicate with all robots that currently have sensor capabilities (i.e. camera and gripper) to perform the foraging service. Service agents reside on any one of the robots that are capable of providing the service. Thus, the global information concerning the task is acquired in a decentralized manner through the use of service agents.

2. **Task Agents:** Task Agents place offers on behalf of the tasks so as to acquire the necessary services. The task agents communicate only with the service agents during negotiations. Once the task has been allocated, the task agent may communicate directly with the robots that have been allocated to the task.

Figure VI.1 provides an overview of an example RACHNA architecture implementation with four service agents (Foraging, Pusher, Watcher, Mapper), N robots ($R_1, R_2, ..., R_N$) and a Task Agent bidding on the services. Each service has certain sensor requirements and only communicates with the relevant robots for purposes of allocation.

An economy is proposed where the tasks are represented by task-agents that are bidding for the services of the individual robots. The economy has a set of robots $R_1, R_2 ..., R_N$

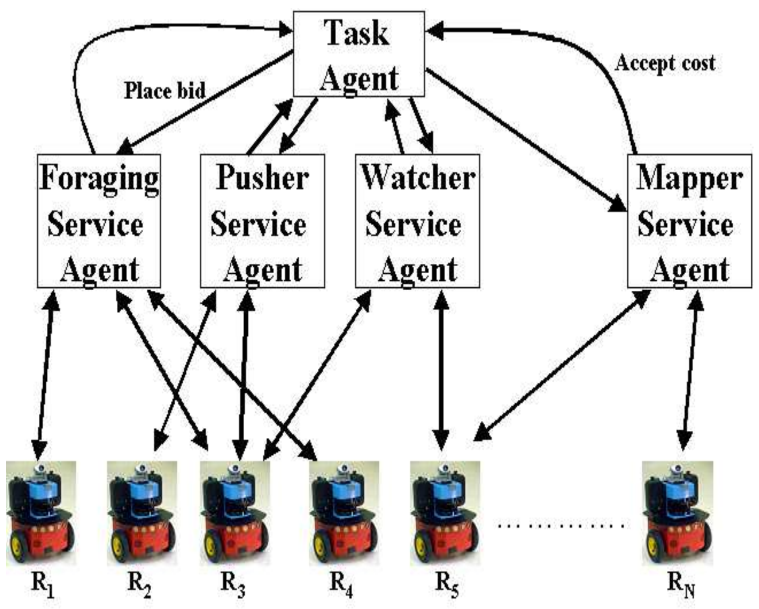


Figure VI.1: An example RACHNA Implementation.

where each robot is equipped with sensor capabilities that enable it to perform various services such as pushing, watching, foraging, etc. The tasks are assumed to be decomposable into the sub-task behaviors (roles) that they require. For example, in the box-pushing task as defined by Gerkey and Matarić (2002a), two pusher sub-task roles are required and one watcher sub-task role is required. Each role is represented by a service agent that is responsible for negotiating with the robots with the desired capability. The roles may be implemented through the use of behavior sets (Parker, 1998). The bids are relatively sparse compared to the overall space of coalitions and will yield a more tractable formulation of the MRCF. Also, unlike other heuristic based algorithms for coalition formation, no restriction is placed on the size of the desired coalitions.

VI.1.2 Utility vs. Cost

The notion of utility is a somewhat implicit albeit universal one. Most definitions of utility incorporate some sort of balance between quality and cost (Gerkey and Matarić, 2003; Tang and Parker, 2005b). However our view is that a predefined cost function may not be ideal for all situations. For instance, there might be a task that is extremely urgent but is relatively inexpensive. In such scenarios, it may be more beneficial to allow the task utility to be input by the user. Also while quantifying cost is relatively straightforward, quantifying quality of task execution prior to coalition formation for a fresh task is still not an exact science. Whatever measure of utility is used, for purposes of comparing coalitions, all that matters is that a mapping exists from each coalition-task pair to a scalar value.

VI.1.3 Multiple Decompositions

There exist many scenarios with multiple decompositions for a particular task exists and it may be advisable to consider many possible decompositions of each task whilst evaluating the potential coalitions. The ASyMTRe (Tang and Parker, 2005b) system is the first autonomous task decomposition system. It may be possible to allow for multiple decompositions by allowing a system such as ASyMTRe to provide different decompositions and introducing 'dummy' goods to incorporate these decompositions as described by Leyton-Brown et al. (2000). One dummy good is introduced per task and all the task agents representing the decompositions bid for that dummy good. Since only one of the decompositions can acquire the dummy good eventually, the final allocation will include at most one of the many task decompositions.

VI.1.4 Robot Failure

If a robot loses control of a sensor or actuator in RACHNA, the system allows for graceful performance degradation. Since there is a mapping from sensor capabilities to behavioral capabilities, if a sensor failure occurs a robot may still be capable of performing an alternative behavior. Consider a robot that is capable of performing the foraging and watcher behaviors. If the robot's gripper is damaged, it will be unable to execute the foraging behavior but it may still be able to perform the watcher behavior. The foraging service agent system simply deletes the robot from the list of foragers and in future auctions this robot will not receive offers relating to the foraging behavior. The robot will remiain in the list of watchers because the relevant sensors for watching (camera) are still intact. Thus, the system allows for graceful degradation in performance. It should be noted that the system makes no attempt at fault detection.

VI.1.5 The Allocation Environments

This work has incorporated three different types of tasks:

1. **Urgent:** These tasks are allowed to pre-empt an ongoing standard task. They generally have a higher average reward per robot and represent emergency tasks that require immediate attention such as fire extinguishing, rescue tasks, etc.

2. **Standard:** These tasks are allocated only when there are sufficient free resources and when the utility of these tasks is sufficient to merit allocation. These tasks may be pre-empted by urgent tasks. Loosely coupled tasks like foraging or tasks that may easily be resumed may fall into this category.

3. **Non pre-emtible:** These tasks are allocated similar to standard tasks but they cannot be pre-empted. Tightly coupled tasks fall under this category because once a tightly coupled task has been initiated, pre-emption would completely debilitate task performance.

We consider two different types of allocations in our system:

1. *Instantaneous Allocation:* This scenario involves the introduction of a number of tasks into the environment and the algorithm must allocate resources to the optimal set of tasks.

2. *Pre-emptive Allocation:* This scenario involves introduction of a single urgent task that requires immediate attention. The urgent task is allowed to tempt the robots into working for itself by offering a higher reward.

Instantaneous Assignment

Instantaneous assignement utilizes multiple round auctions with multiple tasks. The system is responsible for allocating resources to the tasks such that the overall utility is maximized. Recall that services correspond to goods, robots correspond to units of a particular good (service), and task offers correspond to bids. RACHNA adapts the MRCF problem to a distributed environment to allow the system to leverage the inherent redundancy of robots' sensory capabilities, thereby formulating a more tractable formulation of the coalition formation problem. The algorithm proceeds as follows, the auction begins with each task agent sending request messages to the individual service agents. The service agents attempt to obtain the minimum possible price for the requested services. This is achieved by evaluating the minimum salaries that the robots are currently earning and adding a minimum increment for luring the robot to the new task. The service agents then send this information to the task agents. The task agents determine if they have sufficient utility to purchase the required services. If this is the case, then the services (robots) are temporarily assigned to the task. This kind of offer-counteroffer proceeds in a round robin fashion with the salaries of the robots increasing at every step until no service (robot) changes hands during the round. At this point the final stable solution is reached. The formal algorithm is provided in the next section.

VI.1.6 The Algorithm

Initially, each Service Agent SA_i maintains a set of all possible robots. $Robots_i$ that are capable of performing that particular service. Additionally, SA_i is aware of all possible services a robot in $Robots_i$ can perform and keeps track of which service it is currently performing. All robot salaries are initialized to zero. In order to submit a successful bid, a task agent must place an offer so as to increase the current salaries of the robots by a minimum increment, ε_c. Each task agent has a fixed utility, U_c that is used for bidding on the services. Whenever a task discovers that it can no longer place a successful bid, it relinquishes its robots and decreases their salary by an amount ε_c.

Preprocessing

- Initially all task agents submit bids to all relevant service agents.
- Each service agent SA_i then evaluates the following heuristic:

$$score_i = \frac{numbids_i.q(i)}{avgunits_i} \tag{VI.2}$$

  where, $numbids_i$ is the total number of task agents bidding for service provided by SA_i, q_i is the number of robots capable of performing service SA_i, $avgunits_i$ is the average number of total units requested by these task agents.

Once, $score_i$ is evaluated for each service agent SA_i, this score is broadcast to all the service agents and the service agents order themselves according to this score.

Awarding of Services

Upon receipt of a bid b_j from Task Agent TA_k requesting m robots capable of performing SA_i's service, SA_i performs the following steps:

- If $m > |Robots_i|$ ignore the bid (not enough robots).
- Evaluate S_m, the sum of the current m lowest salaries of robots in $|Robots_i|$ that are not already awarded to TA_k by a higher ranked service agent (according to the last received broadcast).
- If $S_m + m\varepsilon_c \leq b_j$
  - Temporarily award the selected robots to the task.
  - Send a message to the purchased robots to increment their salaries by ε_c and to change their current task.

- Else continue.
- Upon receipt of a ‘disown’ signal from a task agent TA_k, a Service Agent SA_i does the following:
  - Search $Robots_i$ for any robots that are currently assigned to TA_k.
  - Decrement the salaries of the robots by ε_c and send a message to those robots.

Whenever a robot receives a message for a change in salary from a service agent, the robot sends a message to all connected service agents informing them of its new payoff, its new task, and the service it is currently required to provide.

Bidding by Task Agents

- Each task agent TA_k receives periodic broadcasts from each service agent SA_i informing it of the current salaries and tasks of the robots performing SA_i's service.
- If TA_k has already been awarded all the required services, continue.
- Compute the net utility, U_{Req}, required to place an acceptable offer to each service agent in order to purchase the required services (by increasing the salaries of the currently lowest paid robots by ε_c).
- If $U_{Req} < U_{Curr}$
  - Place the offers to the service agents.
- Else
  - Send ‘disown’ message to all service agents that have robots currently assigned to task TA_k.
- Upon receipt of the terminate message from all service agents, the task has been awarded the requested robots.

Table VI.1: RACHNA message types.

No.	Sender	Receiver	Type	Content
1	SA	All	Broadcast	Current Salaries, Tasks for all service robots
2	TA	SA	Unicast	Bid for service
3	TA	SA	Unicast	Disown
4	SA	Robot	Multicast	Increment, Decrement
5	Robot	SA	Multicast	Update Salary, task, service

If no message is broadcast by a service agent for a fixed period of time, T_{max}, the auction is deemed closed and the allocation process is stopped. The messaging protocol for the algorithm is depicted in Table VI.1.

VI.1.7 Payoff Stability

Coalition formation in game theory focuses on stable payoffs for agents in competitive scenarios. There are many stability criteria defined in cooperative game theory such as the core, the kernel (see Appendix), and the bargaining set. This section proves that the RACHNA system provides solutions that are stable by the criteria defined by the unconstrained bargaining set.

Definition 1: A **coalitional game with transferable utility** (a TU game) is a pair (N,v) where N is a coalition and v is a function that associates a real number, $v(S)$, with each subset S of N.

Definition 2: A **coalition structure** for N is a partition of N. If (N,v) is a game and R is a coalition structure for N, then the triple (N,v,R) is called a game with a coalition structure. Let (N,v,R) be a game with a coalition structure. Then

$$X(N,v,S) = \{x \in \Re^N \mid x(S) \leq v(S) \; for\, every\; S \in \Re\} \qquad \text{(VI.3)}$$

denotes the set of feasible payoff vectors for (N,v,R).

Definition 3: Let (N,v,R) be a game with coalition structure, $x \in X(N,v,R)$, and let $k,l \in S \in R, k \neq l$. An **objection** of k against l at x with respect to $(N,\ v,\ R)$ is a pair $(P,\ y)$

satisfying:

$$P \in T_{kl} \; and \; y \; \in \Re^{P}; \tag{VI.4}$$

$$y^{i} \geq x^{i} for \; all \; i \in P \; and \; y^{k} > x^{k}; \tag{VI.5}$$

$$y(P) \; \leq \; v(P) \tag{VI.6}$$

where $T_{kl}(N) = T_{kl} = \{S \subseteq N \setminus \{l\} \parallel k \; \in \; S\}$, or T_{kl} is the set of coalitions containing k and not containing l.

Thus, an objection (P, y) of k against l is a potential threat by a coalition P, which contains k but not l, to deviate from x. The purpose of presenting an objection is not to disrupt R, but to demand a transfer of money from l to k, that is, to modify x within $X(N, v, R)$. It is assumed that the players agreed upon the formation of R and only the problem of choosing a point x out of $X(N, v, R)$ has been left open.

Definition: Let (N, v, R) be a game with coalition structure. A vector $x \; \in \; X(N, v, R)$ is stable if for each objection at x there exists a counter-objection. The **unconstrained bargaining set**, $PM(N, v, R)$, is the set of all stable members of $X(N, v, R)$.

Result: The payoff configuration achieved via RACHNA must lie within the unconstrained bargaining set.

Proof: Assume that the solution $x \in (N, v, R)$ does not lie within the bargaining set, in that case there must exist at least one objection $(P, y), (k \in P, l \notin P, k, l \in S \in R)$ by a coalition P, where $P \; \notin \; R$, that does not have a counter objection. Hence, there must exist a coalition that contains a robot k but does not contain robot l, that has sufficient utility to pay a higher salary to all it's member robots than the current payoff x does. However, if such a coalition possibility (or task) existed, it would have bid on the required robots and successfully purchased them during the bidding process. Since the coalition P promises an equal or higher salary to all robots, the service agents would have allocated the robots to P during bidding. The fact that the coalition was not part of the final allocation is due to the fact that the coalition did not have the utility required to purchase the necessary service robots. Hence, no such coalition *P* can exist and we arrive at a contradiction.$\diamond$

Coalition formation in competitive agent environments is traditionally composed of two sub-problems, the first sub-problem is the formation of a coalition structure and the second is the arrival at a stable payoff configuration. Since robotic environments are cooperative, the payoff configuration is not pivotal for task allocation. However, it is still interesting to note that the payoff configuration lies within the unconstrained bargaining set.

Time Extended Assignment

Time extended assignment involves the random introduction of urgent tasks randomly into the environment and the tasks are allocated robot services according to a negotiation or bargaining process between tasks. The negotiation proceeds as follows, the new task submits a request to the required service agents for a certain number of services of that type. The service agents take into account the current salaries of the robots and allow for a bargaining process to ensue with tasks increasing robot salaries until either the new task can successfully purchase the resources or more resources are made available by other tasks releasing them.

VI.2 Experiments

Preliminary experiments were conducted by simulating the RACHNA system on a single computer. Experiment 1 recorded the variation in robot salaries with increasing competition. The second experiment recorded the sensitivity of the system to robot diversity. The sensitivity of the solution quality to the salary increment parameter was recorded in Experiment 3. A comparative analysis of the RACHNA system to simple task allocation techniques was provided in Experiment 4. A set of real world tasks were simulated in the Player/Stage environment to demonstrate task preemption in Experiment 5.

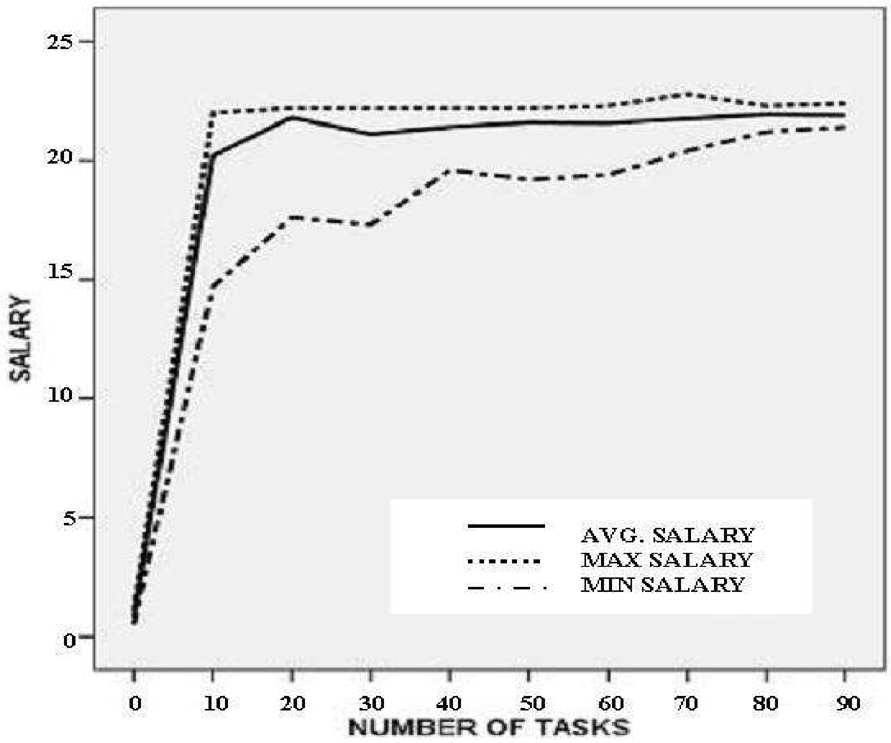


Figure VI.2: Average salary across all robots vs. tasks (bids).

VI.2.1 Wage Increase

The first set of experiments simulates a set of 68 robots and ten services such that each service had exactly ten possible robots capable of providing that particular service. A set of 100 tasks was generated with each task requiring a random vector of resources. The increment in salaries after each auction was recorded. Figure VI.2 shows the average, maximum, and minimum salary curves for all services. The results depict how with increasing competition (more tasks), the salaries increase as robots obtain better offers when there is a shortage of robots. Initially salaries are low (Number of tasks < 20), the salaries rise at different rates depending on demand for a particular service ($20 <$ Number of tasks < 40) and eventually if the demand for each service increases sufficiently, the salaries for all service agents approach high values (Number of tasks = 100).

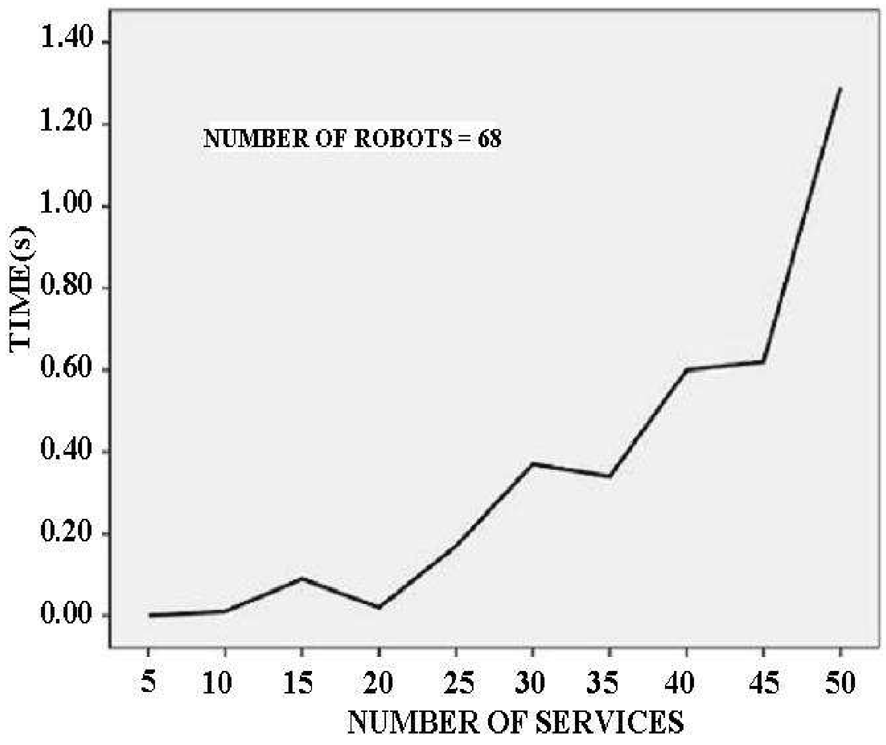


Figure VI.3: Execution Time vs. Number of Services.

VI.2.2 Effect of Diversity

RACHNA leverages the redundancy in the sensory capabilities of the entire set of robots to group robots and make the allocation problem more tractable. Note that RACHNA does not assume anything regarding the diversity of the resulting teams, just the diversity of the entire collection of robots. Figure VI.3 demonstrates that RACHNA's performance deteriorates as the number of services is increased and the number of robots remains constant. The higher the number of services, the lower the redundancy and hence, higher the execution time of the algorithm.

VI.2.3 Impact of Salary Increment

The variation in the number of auction rounds required to reach a stable solution was recorded as the minimum increment (ε_c) was varied from 0.2 to 2. Figure VI.4 shows the resulting graph. It is clear from Figure VI.4 that a small value of ε_c leads to a relatively larger number of auctions, thereby slowing down the allocation process. This is because

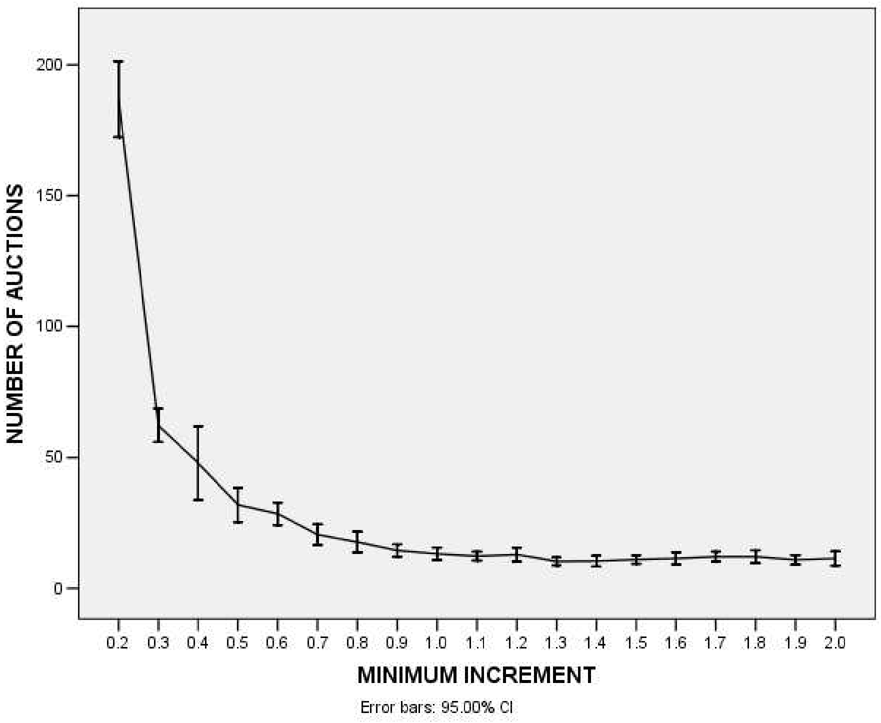


Figure VI.4: Minimum Increment (ε_c) vs. Number of Auction Rounds.

the salaries increase very slowly and the tasks exhaust their utilities after many increments.

Figure VI.5 shows the variation in the final utility obtained as the value of epsilon is varied. The figure demonstrates that the obtained utility steadily decreases as the minimum increment increases. This is because the higher the minimum increment, the lower the level of granularity in the search for better solutions.

Figures VI.4 and VI.5 suggest a tradeoff between running time (number of auctions) and solution quality. Very small increments in payoff yield higher quality solutions but are not efficient in terms of the number of auctions. Large increments in payoff arrive at a solution quicker but with a lower utility value. Ideally an increment value should be chosen so as to optimize this tradeoff.

VI.2.4 Utility Comparison

The solution quality produced by the RACHNA algorithm was compared to that obtained by simple task allocation schemes. Figure VI.6 provides the comparison between the solu-

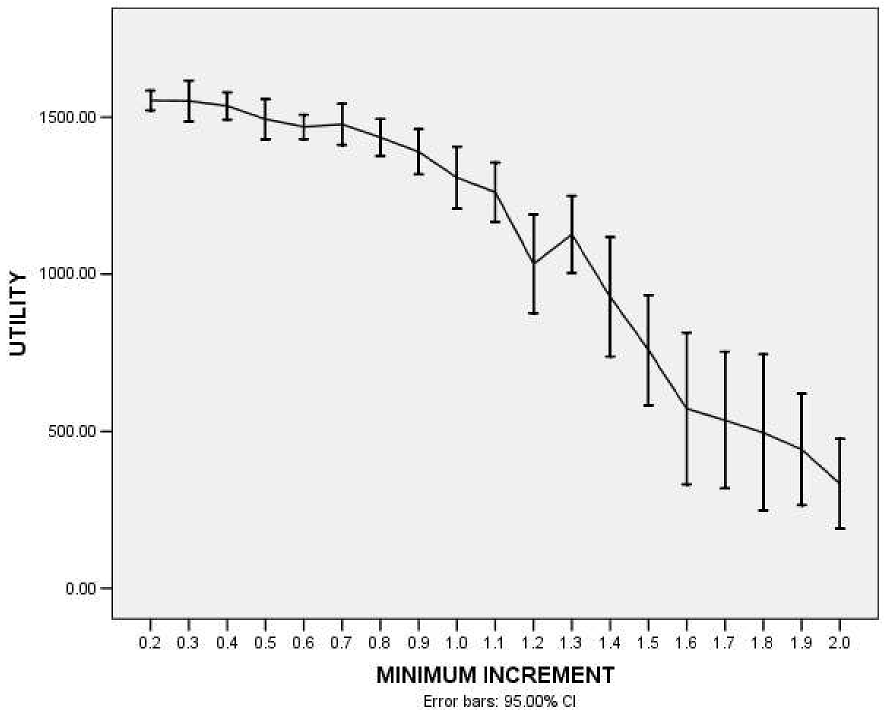


Figure VI.5: Minimum Increment (ε_c) vs. Final Utility.

tion quality produced by the RACHNA system to those produced by the global greedy and random alllocation algorithms. The allocation produced by an algorithm by Leyton-Brown et al. (2000) that utilizes a variant of A* search is also provided. This algorithm yields solutions that are either optimal or very close to optimal and therefore provides a reasonable upper bound on solution quality. The graph demonstrates that RACHNA's solution quality is still sub-optimal, but in a distributed approach such as the one suggested, this is inevitable. As is evident from the figure, RACHNA easily outperforms both the greedy and random allocation algorithms. The reason is that unlike greedy or random search, RACHNA refines the solution in each auction round to include better tasks (bids).

VI.2.5 Preemption Experiments

The player/stage environment was utilized for the set of experiments focused on task preemption in order to formulate a set of real world tasks. The experiments involved a set of five services and ten heterogeneous robots, as shown in Table VI.2. The sensor capabili-

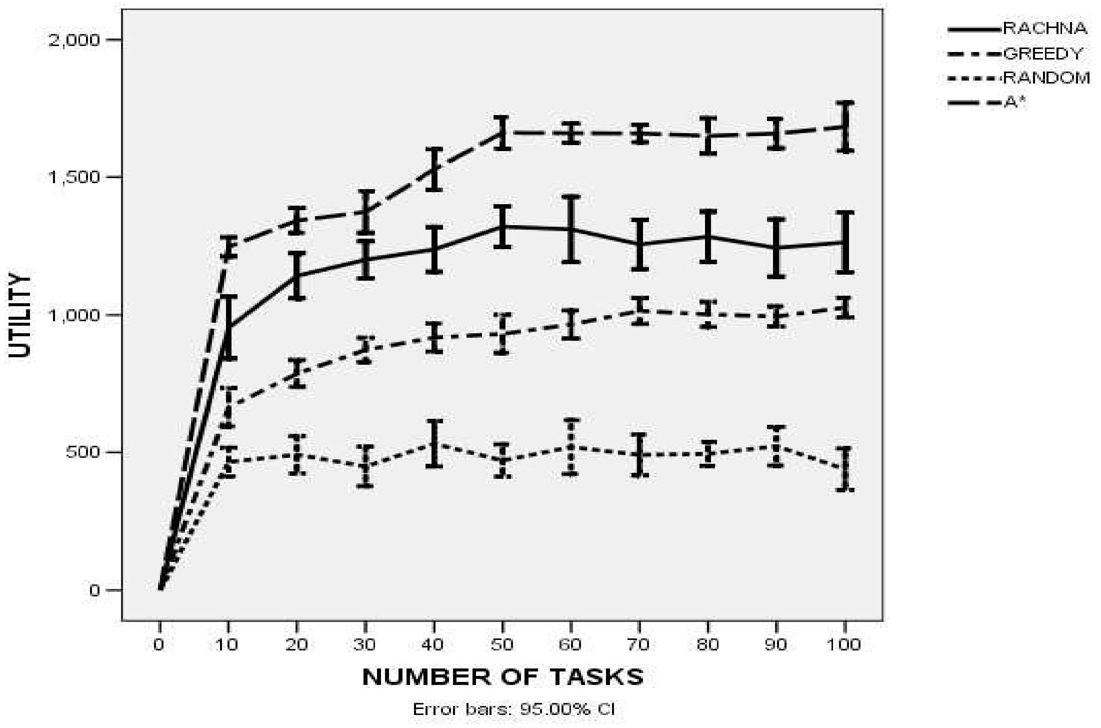


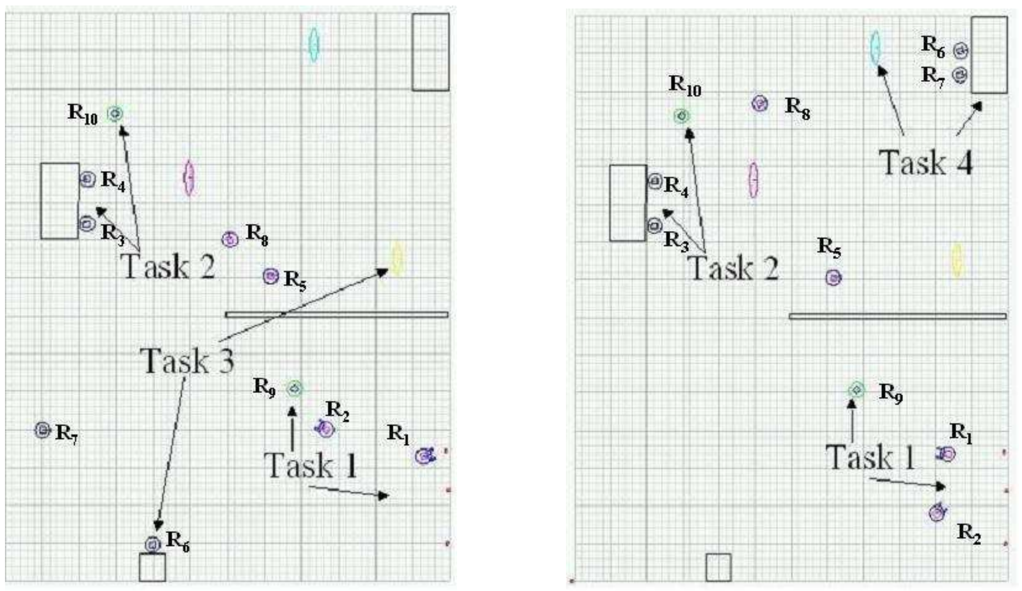
Figure VI.6: Comparison of greedy, random, and A* allocations to RACHNA.

Table VI.2: Services.

Services	Capabilities					Robots
	LRF	Camera	Bumper	Gripper	Sonar	
Foraging	0	1	0	1	1	R_1, R_2
Pushing	1	0	1	0	0	R_3, R_4, R_6, R_7
Object Tracking	0	1	0	0	1	R_1, R_2, R_5, R_8
Sentry-Duty	1	0	0	0	0	$R_3, R_4, R_6, R_7, R_9, R_{10}$

Table VI.3: Tasks.

Tasks	Services				Priority
	Foraging	Pushing	Object-Tracking	Sentry-Duty	
1	2	0	0	1	Standard
2	0	2	0	1	Non-premptible
3	0	1	2	0	Standard
4	0	2	1	0	Urgent



(a) Allocation of Tasks 1,2 and 3 prior to introduction of Task 4.

(b) Allocation of Task 4 and pre-emption of Task 3.

Figure VI.7: Pre-emption after allocation of standard Task 3 by urgent Task 4

Table VI.4: Salary increments for different robots.

ROBOT	AUCTION ROUND							
	1	2	3	4	5	6	7	8
$R_1(F,OT)$	5, T1	10, T4	15, T1	15, T1	15, T1	15, T1	15, T1	15, T1
$R_2(F,OT)$	5, T1	5, T1	10, T4	15, T1	15, T1	15, T1	15, T1	15, T1
$R_3(P,SD)$	5, T2	10, T4	10, T4	10, T4	15, T3	20, T2	20, T2	20, T2
$R_4(P,SD)$	5, T2	5, T2	10, T3	15, T4	15, T4	15, T4	20, T2	20, T2
$R_5(OT)$	5, T3	5, T3	5, T3	10, T4	15, T3	15, T3	20, T4	20, T4
$R_6(P,SD)$	5, T3	5, T3	10, T2	10, T2	15, T4	15, T4	15, T4	15, T4
$R_7(P,SD)$	-	5, T4	5, T4	10, T2	10, T2	10, T2	15, T3	15, T4
$R_8(OT)$	5, T2	5, T2	5, T3	10, T4	10, T4	15, T3	15, T3	10, -
$R_9(SD)$	5, T1	5, T1	5, T1	5, T1	5, T1	5, T1	5, T1	5, T1
$R_{10}(SD)$	5, T2	5, T2	5, T2	5, T1	5, T1	5, T1	5, T1	5, T1

ties of each robot determines which services it can perform. For example Robots 1 and 2 have Camera, Gripper, and Sonar sensors and can perform Foraging and Object Tracking tasks. The experiment involved four heterogeneous tasks as described in Table VI.3. For example, Task 1 required two foraging robots and one sentry-duty robot. While Tasks 1 and 3 were Standard tasks (Section VI.1.5), Task 2 was Non-premptible, and Task 4 was Urgent. Initially the first three tasks were introduced using the algorithm described in Section VI.1.6. Figure VI.7(a) shows the simulation of the first three tasks. Subsequently, Task 4 was introduced into the environment and the bargaining process resulted in Task 3 being preempted, as shown in Figure VI.7(b).

The salary increments and the associated task for each robot after each auction round is shown in Table VI.4. An auction round involves bidding by all tasks at least once. Prior to introduction of Task 4 in round 1, the robots are allocated to Tasks 1,2 and 3 for minimum salaries. After round 2, Task 4 has temporarily acquired R_1, R_3 and R_7 by offering a higher salary to each. As the competition increases for the pusher and object tracker services, all robots posessing these resources draw higher salaries with each successive auction round. Auction round 8 involves Task 3 exhausting its utility and being forced to relinquish R_7 and R_8. The salaries of these robots are lowered by the minimum increment (here 5). Eventually, Task 4 acquires robots R_6 and R_8 from Task 3 at the end of round 8.

Currently, the RACHNA system has been implemented in simulation, and pending implementation with real world robots, the system cannot be completely validated. In particular, foreseeable issues could include communication load, message synchronization, and sensor failure. However, the presented experiments do provide insight into the system's performance with real robots.

VI.3 Summary

This Chapter presented RACHNA, a market-based distributed task allocation scheme based on the multi-unit combinatorial auction problem. RACHNA reverses the auction scheme found in other market based coordination schemes by allowing tasks to bid on robot services rather than the other way around. RACHNA is a utility based system, allowing the user to specify the priority of a task. As mentioned in Chapter I, care was taken while designing the system to ensure that it would be able to accommodate different task types and task environments. RACHNA also allows for multiple decompositions of tasks, the payoff configuration resulting from RACHNA's allocation scheme was proven to lie within the unconstrained bargaining set.

Experiments were conducted to demonstrate the sensitivity of the system to parameters of robot diversity and wage increments. The experiments further demonstrate that the system produces solutions of sub-optimal quality and enables a far more tractable formulation of the coalition formation problem by leveraging the redundancy in robot sensor capabilities. Finally, the idea of preemption of a complex multi-robot task was investigated, and an example of task preemption was demonstrated. Future work involves real world task experiments with the RACHNA system to demonstrate dynamic pre-emption of tasks. Another aspect that needs to be studied is the robustness to communication, and partial or complete robot failure.

CHAPTER VII

SUMMARY OF CONTRIBUTIONS AND FUTURE DIRECTIONS

This chapter consolidates the work presented in previous chapters and summarizes this dissertation's contributions to the field. This chapter also outlines potential avenues for future research.

VII.1 Summary of Contributions

Contributions of this dissertation include:

1. While coalition formation is a well studied area in both distributed artificial intelligence and theoretical computer science, none of the algorithms proposed in these areas have been previously demonstrated in the multi-robot domain. This dissertation idetifies the reasons behind this divide between the software agent and robotic coalition formation algorithms. The issues involved in translating coalition formation algorithms from the software agent setting to the multi-robot domain are identified and solutions are provided.

2. A popular heuristic-based software agent coalition formation algorithm is modified, extended, and validated with a large set of both simulated and real world robots and tasks. To the best of our knowledge, this is the first instance of a multi-robot coalition formation algorithm in the field. The algorithm was further extended and applied to a set of precedence ordered tasks.

3. This dissertation introduces the notion of coalition imbalance and highlights its relevance to the formation of fault tolerant coalitions. The notion of balance is further explored to test for correlation between balance and performance of a multi-robot team. Results suggest that both multi-robot soccer and foraging teams at the upper extremities of the performance spectrum tend to exhibit a higher degree of balance

than mediocre teams. This dissertation also demonstrates how balance information may be further utilized to improve team performance in domains similar to multi-robot soccer where individual contributions may be hard to perceive directly.

4. The dissertation presents RACHNA, a novel market-based coalition formation scheme that leverages the inherent redundancy in the sensory capabilities of robots to enable a more tractable formulation of the coalition formation problem. Most prior market based solutions are based on a contract-net based auction protocol, with robots bidding for tasks. RACHNA reverses the bidding process and allows tasks to bid on robots by incrementing robot payoffs. Also prior task allocation systems appear to be tightly bound to the task domain and do not generalize well to different tasks. RACHNA is more generic and is allows for coalition formation for different tasks. The notion of preemption of complex multi-robot tasks is also explored.

5. This dissertation presents the multi-robot coalition formation problem in the light of prior work done on coalition formation in economics, distributed artificial intelligence, theoretical computer science, and robotics. Problems that share a similar structure with the coalition formation problem are identified in an effort to outline possible avenues for further research into the problem.

Instead of the current task allocation schemes that focus mainly on task specific allocation for single robot tasks, the emphasis in this dissertation was on generic, task independent allocation schemes for multi-robot tasks. While domain knowledge is important and should be utilized to improve performance wherever possible, we believe that it should not be at the expense of the generalizability of the allocation scheme to different tasks.

VII.2 Future Directions

This dissertation provides the first inroads into the coalition formation problem for multi-robot systems. However, much work needs to be done to improve the task allocation schemes presented in this dissertation. The successful migration of the heuristic based algorithm in Chapter III from multi-agent to multi-robot domains suggests that other heuristic based techniques from parallel problems like job shop scheduling, weighted bin-packing, set-covering, and maximal clique may potentially be decentralized for application to the coalition formation problem.

The RACHNA system has still only been implemented in simulation and future experiments involving real world robots need to be performed. The RACHNA system also needs to be extended to include schedules of tasks to allow for a planning component in the system. The long term goal is to integrate the RACHNA system with a suitable interface to allow for complex human machine teaming studies.

Another potential area of research is the handling of communication failures by coalition formation systems and the formulation of techniques to make the coalition formation process more robust to these failures. Currently, the parameters for these systems were adjusted based on trial and error. Future work involves learning and dynamically adjusting these parameters with experience.

So far, the emphasis has been on generic, task independent allocation methods. However, the attempt is not to undermine the role of domain knowledge but to develop systems that would allow for allocation of many different types of tasks. The idea was not to preclude the incorporation of domain knowledge but retain sufficient flexibility to allow for the suitable incorporation of domain knowledge wherever possible. Thus, a future direction of research would be to examine methods to flexibly utilize domain knowledge in an effort to improve performance.

The utility functions used in this dissertation were fairly simple (Chapter III) or random (Chapter VI). Ideally a perfect utility function should encompass and reflect all possible elements influencing the performance of a coalition-task pair such as communication costs, probability of sensor failure, sensory costs, and the benefits of performing the task. Although obtaining the utopian utility function is unattainable in a noisy, dynamic environment, it may be worthwhile to explore different utility functions incorporating these elements.

The dissertation provides an investigative study into the notion of balance and examines its impact on both team performance and fault tolerance. However, it remains to be seen how well the results generalize outside of the domains of multi-robot soccer and foraging. Another important area of research is examination techniques for obtaining balance information in real-time to enable online improvement in performance.

A limitation of the systems described in this dissertation is that multi-robot tasks are assumed to be decomposed into pre-defined roles, to be performed by individual robots. Thus, for true autonomy the task allocation systems require integration with a task decomposition system. Another useful addition would be the flexibility to take into consideration multiple task decompositions. A further limitation of the coalition formation systems is the number of parameters that need to be adjusted for the allocation process to work. A learning mechanism may be worthwhile to implement in order to arrive at optimal values for these parameters.

The difficulty of the coalition formation problem is compounded by the various nuances associated with a robotic environment. Physical constraints, communication costs, and environmental factors have a huge impact on the performance of a multi-robot task allocation algorithm. Thus far, the coalition formation problem has been somewhat neglected by the multi-robot research community. However, as the field matures and task complexity increases, solutions to the coalition formation problem are assuming greater significance. This dissertation provides the initial inroads into the multi-robot coalition formation problem in an effort to both bridge the divide between agent and robotic environments while also directing the attention of the multi-robot research community towards this very important problem.

APPENDIX: DEFINITIONS

The theories of coalition formation are generally with regard to cooperative "*n*-person games in characteristic function form with side payments." The following defines some concepts that are relevant to the coalition formation process as defined in game-theory literature.

COALITION: A coalition is defined as a subset of the set of *N* players and is denoted by S. To say that a coalition *S* is formed, it is required that agreements take place involving approval by every player in S and by no players not in *S* (set *N*-*S*).

COALITION STRUCTURE: A coalition structure is a means of describing how the players in *N* divide themselves into mutually exclusive and exhaustive coalitions. Any proposed or actual partition of the players can be described by a set

$$\rho = S_1, S_2, ..., S_m \tag{1}$$

of the *m* coalitions that formed. The set *P* is a partition of *N* that satisfies three conditions:

$$\left\{ \begin{array}{l} S_j \neq \phi, j = 1, ..., m \\ S_i \cap S_j = \phi foralli \neq j, and \\ \bigcup_{S_j \in \rho} = N \end{array} \right\}$$

COOPERATIVE VS NON-COOPERATIVE GAMES: The defining quality of a cooperative game is that players may enter into mutually binding agreements. Specifically it is assumed that negotiations are compulsory.

CHARACTERISTIC FUNCTION: A cooperative *n* person game in characteristic function form with sidepayments is a pair *(N,v)*, where $N = (A_1, A_2, \ldots, A_N)$ is a set of players and *v* is a real valued function defined on the subsets of *N* called a characteristic function, which assigns a real number *v(S)* to each subset *S* of players. The number *v(S)*, known as the value of *S*, is the money which the coalition can obtain when its members act together. The value of the empty set, is always $v(\phi) = 0$.

PAYOFF CONFIGURATION: The payoff configuration is the means by which any proposed or actual outcome of a game may be expressed. Formally, a payoff configuration (PC) for a set of n agents $A_1, A_2, ..., A_N$ is defined as a pair

$$(x;\rho) = (x_{A_1}, x_{A_2}, \ldots, x_{A_N}; S_1, S_2, \ldots, S_m) \tag{2}$$

where *x* is a payoff vector and *S* is a coalition structure, as defined previously. These definitions require that there cannot be more coalitions than players, and that

$$x(S_j) \equiv \sum_{i \in S_j} x_i = v(S_j), \forall j = 1, 2, \ldots, m. \tag{3}$$

Equation (3) expresses the requirement that each proposed or formed coalition will disburse neither less or more than its value to its members.

SUPER-ADDITIVE VS NON-SUPER-ADDITIVE ENVIRONMENTS: Superadditivity is a property of characteristic functions that says that any two disjoint coalitions can earn at least as much profit by joint effort as they can individually. In characteristic function notation:

$$v(S \cup T) \geq v(S) + v(T) \forall S,T \subseteq N \text{ such that } S \cap T = \phi \tag{4}$$

Superadditivity means that any pair of coalitions is better off by merging into one. Classically it is argued that almost all games are superadditive because in the worst case, the agents in the composite coalition can use solutions that they had when they were in separate coalitions. However, many games are not superadditive because there is some cost to the coalition formation process itself. For example, there might be coordination overhead like communication cost. The multi-robot domain falls into the non-super-additive category because the addition of more robots to a coalition leads to increased interference between the robots and increased computational costs. Also when allocating the coalitions to individual tasks we cannot have a single grand coalition.

SUPERADDITIVE COVER: Aumann and Dreze (1974) defined the superadditive cover of a coalition as the maximum obtainable joint output from any partition of subsets of the coalition. Formally the superadditive cover is given by:

$$\hat{v}(S) = \max_{P} \sum_{j=1}^{P} v(S_j) \tag{5}$$

such that $P = (S_1, \ldots, S_P)$ is a partition of S.

That is the coalition divides into subcoalitioins such that the sum of the values of the subcoalitions is a maximum.

INDIVIDUAL RATIONALITY: An agent joins a coalition only if it can benefit at least as much within the coalition as it could benefit by itself. An agent benefits if it fulfills tasks, or receives a payoff that compensates for the loss of resources or non fulfillment of some of its tasks. Thus individual rationality requires that:

$$x_i \geq \nu(T) \text{ for every } T \subseteq N. \tag{6}$$

GROUP RATIONALITY: Group rationality states that a player A should refuse any PC that yields a lower payoff for the player A than the payoff player A would receive when the group as a whole receives optimal payoff. For superadditive games, the coalition structure that forms will satisfy:

$$\sum_{S\in\rho} \nu(S) = \nu(N) \tag{7}$$

for non-superadditive games,$\nu(N)$ is replaced by its superadditive cover $\hat{\nu}(N)$.

COALITIONAL RATIONALITY: Coalitional rationality extends the principle of group rationality to a subset of players. That is, no combination of players should settle for less than what it can collectively obtain by forming a coalition. Formally, this coalitional rationality constraint can be expressed as the constraint on $(x;\rho)$ where:

$$x(T) \geq \nu(T) \text{ for every } T \subseteq N. \tag{8}$$

CORE: Gillies (1953) introduced the core for superadditive games. The concept was generalized by Aumann and Dreze (1974) to non-superadditive games. Formally, the Core of a game $(N;\nu)$ is the set of all PCs $(x;\rho)$, if any, such that $x(T) \geq \nu(T) \forall T \subseteq N$. Informally, it is the set of all PCs that satisfy coalitional, group and individual rationality.

PARETO OPTIMALITY: A payoff vector is Pareto-optimal if no other payoff vector dominates it, i.e., no other payoff vector is better for some of the agents and no worse for the others. A specific Pareto-optimal payoff vector is not necessarily the best for all the agents. There may be multiple Pareto-optimal payoff vectors where different agents prefer different payoff vectors. Therefore, Pareto optimality is insufficient for the evaluation of possible coalitions.

EXCESS: The excess (Davis and Maschler, 1965) of a coalition C with respect to the coalitional configuration PC is defined by :

$$e(C) = v(C) - \sum_{A_i \in C} x_i \tag{9}$$

where x_i is the payoff of agent A_i in PC. C is not necessarily a coalition in PC, and it can be in any other coalitional configuration. $v(C)$ is the coalitional value of coalitional C.

SURPLUS: The maximum surplus S_{ij} of agent A_i over agent A_j with respect to a PC is defined by , $S_{ij} = \max_{C|A_i \in C, A_j \notin C} e(C)$ where *e(C)* represents the excesses of all the coalitions that include A_i and exclude A_j, and the coalitions C are not in PC, the current coalitional configuration. Agent A_i outweighs agent A_j if $S_{ij} > S_{ji}$ and $x_j > (A_j)$, where $v(A_j)$ is the coalitional value of agent A_j in a single agent coalition.

If one agent A_i has a higher maximum surplus than an agent A_j, then A_i is stronger than A_j and can claim part of $A_j{}'$ s payoff in the same coalitional configuration, but this claim is limited by individual rationality which requires that $x_j > (A_j)$. This means that in any suggested coalition, agent A_j must receive more payoff than it obtains by itself in a single member coalition. Two agents A_i and A_j cannot outweigh one another and are in equilibrium if one of the following conditions is satisfied:

$$\left\{ \begin{array}{l} S_{ij} = S_{ji} \\ S_{ij} > S_{ji} and x_j = v(A_j) \\ S_{ij} < S_{ji} and x_j = v(A_j) \end{array} \right\}$$

.

KERNEL: The definition of the kernel as provided by Davis and Maschler (1965) is the set K of all PCs $(x;\rho)$ such that every pair of players A_i and A_j are in equilibrium as defined above.

BIBLIOGRAPHY

Abdallah, S. and Lesser, V. (2004). Organization-Based Cooperative Coalition Formation. In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Techonology, IAT*, pages 162–168.

Aumann, R. J. and Dreze, J. H. (1974). Cooperative games with coalition structures. *International Journal of Game Theory*, 3:217–237.

Baker, D. P. and Salas, E. (1992). Principles for measuring teamwork skills. *Human Factors*, 34(6):469–475.

Balas, E. and Padberg, M. (1976). Set partitioning: A survey. *SIAM Review*, 18:710–760.

Balch, T. (1997). Clay: Integrating motor schemas and reinforcement learning. Technical Report GIT-CC-97-11, College of Computing, Georgia Institute of Technology.

Balch, T. (1998). *Behavioral Diversity in Learning Robot Team*. PhD thesis, Georgia Institute of Technology, Dept. of Computer Science.

Balch, T. (2002). Taxonomies of multirobot task and reward. In Balch, T. and Parker, L. E., editors, *Robot Teams: From Diversity to Polymorphism*, pages 323–335.

Balch, T. and Arkin, R. C. (1994). Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):1–25.

Balch, T. and Ram, A. (1998). Integrating robotics research with javabots. In *Working Notes of the American Association for Artificial Intelligence*.

Bandura, A. (1986). *Social Foundations of Thought and Action*. Prentice Hall, Englewood Cliffs.

Bar-Yehuda, R. and Even, S. (1981). A linear time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2:198–203.

Bass, B. M. (1980). Individual capability, team performance and team productivity. *Human Performance and Productivity*, pages 179–232.

Blum, A. L. and Furst, M. L. (1997). Fast planning through planning graph analysis. *Artificial Intelligence*, (90):281–300.

Boicu, M., Tecuci, G., Stanescu, B., Marcu, D., Barbulescu, M., and Boicu, C. (2004). Design principles for learning agents. In *American Association of Artificial Intelligence Workshop on Intelligent Agent Architectures: Combining the Strengths of Software Engineering and Cognitive Systems, Techical Report WS-04-07*, pages 26–33.

Borenstein, J. and Koren, Y. (1991). The vector field histogram: A fast obstacle-avoidance for mobile robots. *IEEE Journal of Robotics and Automation*, 7(3):278–288.

Borgwardt, K.-H. (1982). Some distribution-independent results about the asymptotic order of the average number of pivot steps of the simplex algorithm. *Mathematics of Operations Research*, 7:441–461.

Botelho, S. C. and Alami, R. (1999). M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1234 – 1238.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23.

Brou, R., Doane, S., Bradshaw, G., Giesen, J. M., and Jodlowski, M. (2005). The role of individual differences in dynamic team performance. In *Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting*, pages 1238–1242.

Caloud, P., Choi, W., Latombe, J.-C., Pape, C. L., and Yim, M. (1990). Indoor automation with many mobile robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 67–72.

Chen, Q., Zhu, K., and McCalley, J. D. (2001). Dynamic decision-event trees for rapid response to unfolding events in bulk transmission systems. In *IEEE Porto Power Tech. Proceedings*, pages SSR5–399.

Choset, H. (2001). Coverage for robotics - A survey on recent results. *Annals of Mathematics and Artificial Intelligence*, 31:113–126.

Chu, P. C. and Beasley, J. E. (1996). A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94:396–404.

Chvatal, V. (1979). A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235.

Cohen, P. R. and Levesque, H. J. (1991). Teamwork. *Nous, Special Issue on Cognitive Science and Artificial Intelligence*, 25(4):487–512.

Collins, J., Jamison, S., Mobasher, B., and Gini, M. (1997). A market architecture for multi-agent contracting. Technical Report 97-15, University of Minnesota, Dept. of Computer Science.

Dahl, T. S., Matarić, M. J., and Sukhatme, G. S. (2003). Multi-robot task-allocation through vacancy chains. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 14–19.

Dang, V. D. and Jennings, N. (2004). Generating coalition structures with finite bound from the optimal guarantees. In *Proceedings of the Third International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 564–571.

Dantzig, G. B. (1972). *Linear Programming and Extensions*. Princeton University Press, Princeton, New Jersey.

Davis, M. and Maschler, M. (1965). The kernel of a cooperative game. *Naval Research Logistics Quarterly*, 12:223–259.

DeVries, S. and Vohra, R. (2003). Cominatorial auctions: A survey. *INFORMS Journal on Computing*, 135:284–309.

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

Doorenbos, B., Etzioni, O., and Weld, D. S. (1993). A scalable comparison-shopping agent for the world wide web. In *International Joint Conference on Artificial Intelligence*, pages 39–48.

Dudek, G., Jenkin, M., and Milios, E. (2002). Taxonomies of multirobot systems. In Balch, T. and Parker, L. E., editors, *Robot Teams: From Diversity to Polymorphism*, pages 1–26, Natick, MA. A. K. Peters, Ltd.

Dyer, M. E. and Wolsey, L. A. (1990). Formulating the single machine sequencing problem with release dates as a mixed integer programming problem. *Discrete Applied Mathematics*, 26:255–270.

Farenelli, A., Iocchi, L., and Nardi, D. (2004). Multirobot systems: A classification focussed on coordination. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 34(5):2015–2028.

Fass, L. F. (2004). Automatic-theoretic view of agent coalitions. In *American Association Of Artificial Intelligence Workshop on Forming and Maintaining Coalitions and Teams in Adaptive Multiagent Systems, Technical Report WS-04-06*, pages 18–21.

Fenwick, J. W., Newman, P. M., and Leonard, J. J. (2002). Cooperative concurrent mapping and localization. In *IEEE International Conference on Robotics and Automation*, pages 1810–1817.

Fisher, M. and Kedia, P. (1990). Optimal solution of set covering / partitioning problems using dual heuristics. *Management Science*, 36(6):674–688.

Fleishman, E. A. and Zaccaro, S. J. (1992). Toward a taxonomy of team performance functions. In Swezey, R. W. and Salas, E., editors, *Teams: Their Training and Performance*. 31–56.

Fort, R. and Maxcy, J. (2003). Competitive balance in sports leagues: An introduction. *Journal of Sports Economics*, 4:154–160.

Fox, M. S., Barbuceanu, M., and Teignen, R. (2000). Agent oriented supply chain management. *International Journal of Flexible Manufacturing Systems*, 12:165–188.

Garrido, A., Salido, M., Barber, F., and López, M. (2000). Heuristic methods for solving job-shop scheduling problems.

Gasser, L. (1993). Social knowledge and social action. In *International Joint Conference on Artificial Intelligence*, pages 751–757.

Gerkey, B. and Matarić, M. J. (2003). A framework for studying multi-robot task allocation. In *Proceedings of the Multi-Robot Systems: From Swarms to Intelligent Automata*, volume 2, pages 15–26.

Gerkey, B. and Matarić, M. J. (2004a). Are (explicit) multi-robot coordination and multi-agent coordination really so different? In *Proceedings of the AAAI Spring Symposium on Bridging the Multi-Agent and Multi-Robotic Research Gap*, pages 1–3.

Gerkey, B., Thrun, S., and Gordon, G. (2005). Parallel stochastic hill-climbing with small teams. In *Proceedings of the Multi-Robot System Workshop*, pages 65–77.

Gerkey, B. P. and Matarić, M. J. (2000). MURDOCH: Publish/Subscribe task allocation for heterogeneous agents. In *Proceedings of Autonomous Agents*, pages 203 – 204.

Gerkey, B. P. and Matarić, M. J. (2002a). Pusher-watcher: An approach to fault-tolerant tightly-coupled robot coordination. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 464 – 469.

Gerkey, B. P. and Matarić, M. J. (2002b). 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. P. and Matarić, M. J. (2004b). A formal analysis and taxonomy of task allocation in multi-robot systems. *International Journal of Robotics Research*, 23(9):939–954.

Gerkey, B. P., Vaughan, R. T., Stoy, K., Howard, A., Sukhatme, G. S., and Matarić, M. J. (2001). Most valuable player: A robot device server for distributed control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1226–1231.

Gillies, D. (1953). *Some Theorems on N-Person Games*. PhD thesis, Department of Mathematics, Princeton University.

Green, S., Hurst, L., Nangle, B., and Cunningham, P. (1997). Software agents: A review. Technical report, Department of Computer Science, Trinity College, Dublin, Ireland.

Hackman, J. R. (1983). A normative model of work team effectiveness. Technical Report 2, Yale University, New Haven.

Hoffman, K. and Padberg, M. (1993). Solving airline crew-scheduling problems by branch-and-cut. *Management Science*, 39(6):667–682.

Ilgen, D. R. (1999). Teams embedded in organizations: Some implications. *American Psychologist*, 54:129–139.

Jennings, A. and Higuchi, H. (1992). A personal news service based on a user model neural network. *IEICE Transactions on Information Systems*, 75(2):198–209.

Jung, D. and Zelinsky, A. (2000). Grounded symbolic communication between heterogeneous cooperating robots. *Autonomous Robots*, 8(3):269–293.

Khoo, L. P., Tar, S. B., and Lee, S. S. G. (1998). The potential of intelligent software agents in the world wide web in automaing part procurement. *International Journal of Purchasing and Materials Management*, 34(1):46–53.

Kitano, H., Tambe, M., Stone, P., Veloso, M., Coradeschi, S., Osawa, E., Matsubara, H., Noda, I., and Asada, M. (1997). The Robocup synthetic agent challenge 97. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 24–29.

Klee, V. and Minty, G. (1972). How good is the simplex algorithm? In Shisha, O., editor, *Qualities*, volume III, pages 159–175.

Kleinman, D. L. and Serfaty, D. (1989). Team performance assessment in distributed decision-making. In *Proceedings of the Symposium on Interactive Networked Simulation for Training*, pages 22–27.

Klusch, M. and Shehory, O. (1996). A polynomial kernel-oriented coalition formation algorithm for rational information agents. In *Proceedings of International Conference On MultiAgent Systems*, pages 157–164.

Korte, B. and Vygen, J. (2000). *Combinatorial Optimization: Theory and Optimization*. Springer-Verlag, Berlin.

Laengle, T., Lueth, T. C., Rembold, U., and Woern, H. (1998). A distributed control architecture for autonomous mobile robots. *Advanced Robotics*, 12(4):411–431.

Leyton-Brown, K., Shoham, Y., and Tennenholtz, M. (2000). An algorithm for multi-unit combinatorial auctions. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 56–61.

Li, X. and Soh, L.-K. (2004). Investigating reinforcement learning in multiagent coalition formation. In *American Association of Artificial Intelligence Workshop on Forming and Maintaining Coalitions and Teams in Adaptive Multiagent Systems, Technical Report WS-04-06*, pages 22–28.

Lin, L. and Zheng, Z. (2005). Combinatorial bids based multi-robot task allocation. In *Proceedings of the International Conference on Robotics and Automation*, pages 1145–1150.

Low, K. H., Leow, W. K., and M. H. Ang, J. (2004). Task allocation via self-organizing swarm coalitions in distributed mobile sensor network. In *Proceedings of the American Association of Artificial Intelligence*, pages 28–33.

Matarić, M. J. (1997). Reinforcement learning in the multi-robot domain. *Autonomous Robots*, 4(1):73–83.

Mohammed, S. and Angell, L. (2003). Personality heterogeneity in teams: Which differences make a difference for team performance? *Small Group Research*, 34(6):651–677.

Murphy, R. R., Lisetti, C., Tardiff, R., Irish, L., and Gage, A. (2002). Emotion based control of cooperating heterogeneous mobile robots. *IEEE Transactions on Robotics and Automation*, 18(5):744–757.

Nivea, F., Fleishman, E. A., and Reick, A. (1978). Team dimensions: Their identity, their measurement and relationships. Technical Report 1, Advanced Resources Research Organization, Washington D.C.

Oser, R. L., McCallum, G. A., Salas, E., and Morgan, B. J. (1999). Toward a definition of teamwork: An analysis of critical team behaviour. Technical Report NTSC technical report no. 90-009, Orlando: Naval Training Systems Centre.

Paris, C. R., Salas, E., and Canon-Bowers, J. A. (2000). Teamwork in multi-person systems: A review and analysis. *Ergonomics*, 43(8):1052–1075.

Parker, L. E. (1995). The effect of action recognition and robot awareness in cooperative robotic teams. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 212–219.

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. (1999). Cooperative robotics for multi-target observation. *Intelligent Automation and Soft Computing, Special Issue on Robotics Research at Oak Ridge National Laboratory*, 5(1):5–19.

Prince, C. and Salas, E. (1993). Training and research for teamwork in the military aircrew. In Wiener, E. L., Kanki, B. G., and Heinreich, R. L., editors, *Cockpit Resource Management*, pages 337–366.

Pynadath, D. V. and Tambe, M. (2002). The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423.

Rapoport, A. (1970). *N-Person Game Theory*. University of Michigan Press, University of Michigan.

Rappaport, J. P. and Kahan, A. (1984). *Theories of Coalition Formation*. Lawrence Erlbaum Associates, London.

Ruffell-Smith, H. M. (1979). A simulator study of the interaction of pilot workload with errors. Technical Report TM-78482, Moffet Field: National Aeronautics and Space Adminstration, Ames Research Centre.

Sadeh, N. M. and Fox, M. S. (1996). Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, 86:1–41.

Sadeh, N. M., Sycara, K., and Xiong, Y. (1995). Backtracking techniques for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, 76:455–480.

Sandholm, T. (1993). An implementation of the contract net protocol based on marginal cost calculations. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 256–262.

Sandholm, T. (2002). Algorithm for optimal winner determination in cominatorial auctions. *Artificial Intelligence*, 135:1–54.

Sandholm, T. and Lesser, V. (1996). Advantages of a leveled commitment contracting protocol. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 126–133.

Sandholm, T. W., Larson, K., Andersson, M., Shehory, O., and Tomhe, F. (1999). Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1-2):209–238.

Sandholm, T. W. and Lesser, V. R. (1995). Coalition formation among bounded rational agents. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 662–669.

Schneider, J., Apfelbaum, D., Bagnell, D., and Simmons, R. (2005). Learning opportunity costs in multi-robot market based planners. In *Proceedings of the International Conference on Robotics and Automation*, pages 1151–1156.

Schramm, C., Bieszczad, A., and Pagurek, B. (1998). Application-oriented network modeling with mobile agents. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*, pages 696–700.

Schrijver, A. (1986). *Theory of Linear and Integer Programming*. Wiley, Amsterdam.

Shannnon, C. E. (1949). *The Mathematical Theory of Communication*. University of Illinois Press, Illinois.

Shapely, L. S. and Shubik, M. (1973). *Game Theory in Economics*. Rand Corporation, Santa Monica, California.

Shehory, O. and Kraus, S. (1995). Task allocation via coalition formation among autonomous agents. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 655–661.

Shehory, O. and Kraus, S. (1996a). Cooperative goal-satisfaction without communication in large-scale agent-systems. In *Proceedings of the European Conference on Artificial Intelligence*, pages 544–548.

Shehory, O. and Kraus, S. (1996b). Formation of overlapping coalitions for precedence ordered task-execution among autonomous agents. In *Proceedings of International Conference on MultiAgent Systems*, pages 330–337.

Shehory, O. and Kraus, S. (1998). Methods for task allocation via agent coalition formation. *Artificial Intelligence Journal*, 101(1-2):165–200.

Shoham, Y. (1997). An overview of agent-oriented programming. In Bradshaw, J. M., editor, *Software Agents*, Menlo Park, California. AAAI press.

Simmons, R., Singh, S., Hershberger, D., Ramos, J., and Smith, T. (2000). First results in the coordination of heterogeneous robots for large-scale assembly. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*, Honolulu Hawaii.

Smith, R. 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.

Sorbella, R., Chella, A., and Arkin, R. (2004). Metaphor of politics: A mechanism of coalition formation. In *American Association of Artificial Intelligence Workshop on Forming and Maintaining Coalitions and Teams in Adaptive Multiagent Systems, Technical Report WS-04-06*, pages 45–53.

Spielman, D. A. and Teng, S.-H. (2001). Smoothed analysis: Why the simplex algorithm usually takes polynomial time. In *Proceedings of ACM symposium on Theory of Computing*, pages 296–305.

Srinivasan, V. (1971). A hybrid algorithm for the one machine sequencing problem to minimize total tardiness. *Naval Research Quarterly*, 18:317–327.

Stearns, R. E. (1968). Convergent transfer schemes for n-person games. *Transactions of the American Mathematical Society*, 134:449–459.

Steiner, I. D. (1972). *Group Processes and Productivity*. Academic Press, New York.

Stentz, A. and Dias, M. B. (1999). A free market architecture for coordinating multiple robots. Technical Report CMU-RI-TR-01-26, The Robotics Institute, Carnegie Mellon University.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Sycara, K. (1995). Intelligent agents and information revolution. In *UNICOM Seminar on Intelligent Agents and their Business Applications*.

Sycara, K. and Zeng, D. (1996). Coordination of multiple intelligent software agents. *International Journal of Intelligent and Cooperative Information Systems*, 5(2-3):181–211.

Tang, F. and Parker, L. E. (2005a). Coalescing multi-robot teams through ASyMTRe: A formal analysis. In *Proceedings of the IEEE International Conference on Advanced Robotics (ICAR)*, pages 817 – 824.

Tang, F. and Parker, L. E. (2005b). ASyMTRe: Automated synthesis of multi-robot task solutions through software reconfiguration. In *Proceedings of the IEEE International Conference on Robotics and Automation*. 1770-1777.

Tripathi, A., Koka, M., Karanth, S., Osipkov, I., Talkad, H., Ahmed, T., Johnson, D., and Dier, S. (2004). Robustness and security in a mobile-agent based network monitoring system. Technical Report TR 04-003, University of Minnesota, Department of Computer Science and Engineering.

Ulam, P. and Arkin, R. (2004). When good comms go bad: Communications recovery for multi-robot teams. In *Proceedings of the IEEE International Conference of Robotics and Automation*, volume 4, pages 3727– 3734.

Vig, L. and Adams, J. A. (2005). Issues in multi-robot coalition formation. In Parker, L. E., Schneider, F. E., and Schultz, A. C., editors, *Proceedings of Multi-Robot Systems. From Swarms to Intelligent Automata*, volume 3, pages 15–26, Washington D. C. Springer Verlag.

Vig, L. and Adams, J. A. (2006a). Market-based multi-robot coalition formation. In Gini, M. and Voyles, R., editors, *Distributed Autonmous Robotics Systems*, volume 7, pages 227–236, Minneapolis. Springer Verlag.

Vig, L. and Adams, J. A. (2006b). Multi-robot coalition formation. *IEEE Transactions on Robotics*, 22(4):637–649.

Vohra, R. (1995). Coalitional non-cooperative approaches to cooperation. Technical report, Brown University, Dept. of Economics.

Vuurpijl, L. G. and Schomaker, L. R. B. (1998). A framework for using multiple classifiers in a multiple-agent architecture. In *Third IEE European Workshop on Handwriting Analysis and Recognition*, pages 8/1–8/6.

Wagner, H. M. (1959). An integer programming model for machine scheduling. *Naval Research Quarterly*, 6:131–140.

Werger, B. and Matarić, M. J. (2000). Broadcast of local eligibility: Behavior based control for strongly cooperative multi-robot teams. In *Proceedings of Autonomous Agents*, pages 21–22.

White, T. and Pagurek, B. (1998). Towards multi-swarm problem solving in networks. In *Proceedings of the 3rd International Conference on Multi-Agent Systems*, pages 333–340.

Wooldridge, M. and Jennings, N. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152.

Wu, L. S. (1977). A dynamic theory for the class of games with nonempty cores. *SIAM Journal of Applied Mathematics*, 32:328–338.

Zlot, R. and Stentz, A. (2005). Complex task allocation for multiple robots. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 67–72.