A Market Approach to Multirobot Coordination

M. Bernardine Dias
Anthony (Tony) Stentz

CMU-RI -TR-01-26

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

August 2001

Abstract

The problem of efficient multirobot coordination has risen to the forefront of robotics research in recent years. Interest in this problem is motivated by the wide range of application domains demanding multirobot solutions. In general, multirobot coordination strategies assume either a centralized approach, where a single robot/agent plans for the group, or a distributed approach, where each robot is responsible for its own planning. Inherent to many centralized approaches are difficulties such as intractable solutions for large groups, sluggish response to changes in the local environment, heavy communication requirements, and brittle systems with single points of failure. The key advantage of centralized approaches is that they can produce globally optimal plans. While most distributed approaches can overcome the obstacles inherent to centralized approaches, they can only produce suboptimal plans.

This work explores the development of a market-based architecture that will be inherently distributed, but will also opportunistically form centralized sub-groups to improve efficiency, and thus approach optimality. Robots will be self-interested agents, with the primary goal of maximizing individual profits. The revenue/cost models and rules of engagement will be designed so that maximizing individual profit has the benevolent effect of moving the team toward the globally optimal solution. This architecture will inherit the flexibility of market-based approaches in allowing cooperation and competition to emerge opportunistically. The outlined approach will address the multirobot control problem for autonomous robotic colonies carrying out complex tasks in dynamic environments where it is highly desirable to optimize to whatever extent possible. Future work will develop the core components of a market-based multirobot control-architecture, investigate the use of a negotiation protocol for task distribution, design and implement resource and role management schemes, and apply optimization techniques to improve system performance. The automated robot colonies domain is targeted for implementation and evaluation of the architecture. Portability of the architecture to other application domains will also be illustrated.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

In this digital age, the demand for technological solutions to increasingly complex problems is climbing rapidly. With this increase in demand, the tasks which robots are required to execute also rapidly grow in variety and difficulty. A single robot is no longer the best solution for many of these new application domains; instead, teams of robots are required to coordinate intelligently for successful task execution. For example, a single robot is not an efficient solution to automated construction, urban search and rescue, assembly-line automation, mapping/investigation of unknown/hazardous environments, and many other similar tasks. Multirobot solutions are paramount for several reasons:

1. A single robot cannot perform some tasks alone, a team is required for successful execution. While in many cases it may be possible to design a single robot capable of executing all tasks, many problems are better suited to team-execution. For example, a single robot can accomplish moving heavy objects if the robot is designed appropriately. However, in many cases, it is simpler to design a team of robots that cooperate to move the heavy objects efficiently. Other application domains such as robotic soccer require a team of robots and cannot be executed with a single robot.

2. A robot team can accomplish a given task more quickly than a single robot can by dividing the task into sub-tasks and executing them concurrently in application domains where the tasks can be decomposed. Application domains such as mapping of unknown areas and searching for landmines require careful coverage of a large area. Problems such as these can be easily decomposed into components such that a team of robots can divide the workload and execute sub-portions of the task concurrently, thus completing the overall task more efficiently.

3. A team can make effective use of specialists designed for a single purpose (for example, scouting an area, picking up objects, or hauling payload), rather than requiring that a single robot with versatile capabilities be a generalist, capable of performing all tasks, but expert at no tasks. This allows more flexibility in designing the robots since a robot that needs to haul heavy payloads can be built with a heavy base for stability and strength, while a robot required to provide visual feedback can be designed to be more agile and move around with greater speed.

4. A team of robots can localize themselves more efficiently if they exchange information about their position whenever they sense each other. This allows more robust localization capabilities. In an environment where a single robot would have to rely on landmarks of some sort for localization, a team could have the added advantage of being able to benefit from the localization information of their teammates.

5. A team of robots generally provides a more robust solution by introducing redundancy, and by eliminating any single point of failure as long as there is overlap between the robots' capabilities. For example, a team of robots equipped with cameras, will be a more reliable system for constructing vision-based maps of a dynamic environment because the failure of a single one of these robots will not jeopardize the entire mission.

6. A team of robots can produce a wider variety of solutions than a single robot can, and hence a team can opportunistically respond to dynamic conditions in more creative and efficient ways. Even if a team of robots does not overlap entirely in terms of specialization, the collective resources of the group can be used in creative ways to solve problems. For example, if a diagnostic robot loses its camera during operation, another robot with a camera could aid the diagnostic robot to complete its tasks by providing visual feedback. Similarly, if a rover gets stuck in the mud, one or more of its teammates can assist the stuck robot by pushing it out of the mud.

Thus, for many applications, a team of robots can be used more effectively. Introduced below are some of the more prominent application domains which would benefit from efficient coordination of multirobot systems:

- **Autonomous robot colonies for operations in remote locations**:

  Many applications in the future will require a colony of robots to autonomously execute complex tasks, while humans intervene remotely from time to time to alter the procedure of operations, remedy a situation beyond the capabilities of the robots, or coordinate with the robots to accomplish additional goals. Examples of such application domains are extra-planetary exploration and construction, operations in high-risk environments such as coal mining or scientific exploration of hazardous environments, and crop cultivation in underwater environments.

- **Robotic aids for urban reconnaissance**:

  Military Operations in Urban Terrain pose fierce constraints such as limited visibility, complex and expansive fortifications, limited intelligence, and the presence of native populations and other non-combatants that prohibit deployment of large forces. Moreover, the use of certain threats, e.g. biological and chemical agents, against both land forces and indigenous population in urban settings is an increasing likelihood. These conditions place both land forces and non-combatants in a highly non-deterministic, dangerous, confrontational, and volatile environment. The development of robotics technology will enable minimally invasive and precise operations that reduce risk to both ground forces and non-combatants by removing soldiers from dangerous and sometimes confrontational tasks. Potential tasks for robotic systems include mine sweeping, reconnaissance, security and monitoring presence, and communications infrastructure.

- **Robotic aids for urban search and rescue**:

  Urban Search and Rescue (USAR) workers have forty eight hours to find trapped survivors in a collapsed structure, otherwise the likelihood of finding victims still alive is nearly zero. Earthquake disaster mitigation requires rapid and efficient search and rescue of survivors. As recently seen in Turkey and Taiwan, the magnitude of the devastation of urban environments exceeds the available resources (USAR specialists, USAR dogs, and sensors) needed to rescue victims within the critical first 48 hours. Moreover, the mechanics of how large structures collapse often prevent rescue workers from searching buildings due to the unacceptable personal risk and the added risk to survivors from further collapse of the building. Furthermore, both people and dogs are frequently too big to enter voids, limiting the search to no more than a few feet from the perimeter. Robots can make a significant impact in this domain if made capable of aiding humans in USAR efforts.

- **Automated warehouse management**:

  Warehouse operators face the competing challenges of reducing costs while improving customer responsiveness. Order picking represents one of the costliest processes within Distribution Centers because of high labor content and equipment investment. Operators rely on humans to pick orders and either transport the material with manually driven industrial hand trucks (also known as pallet trucks or pallet jacks) or conveyor systems. An automated approach to case order picking has the potential to capture the benefits of manually driven pallet trucks and conveyor systems. Automated pallet jacks would roam the distribution center under the warehouse management system's global supervision and move to pickers stationed to serve one or two aisles. Pickers spend less time traveling and more time picking; the pallets are built as cases are picked, and no major infrastructure changes are required.

- **Intelligent environments**:

  Intelligent Environments are spaces in which computation is seamlessly used to enhance ordinary activity. They enable tasks historically outside the normal range of human-computer interaction by connecting computers to normal, everyday phenomena that have traditionally been outside the purview of contemporary user-interfaces. Their applications are intelligent rooms and personal assistants. Many familiar environments such as office buildings, supermarkets, schoolrooms, and restaurants are highly likely to incrementally evolve into intelligent environments within the next couple of decades. In these environments, agents who will oversee optimized utilization of the resources can represent different resources. These agents can also resolve any conflicts about resource utilization. Moreover, the agents can keep track of maintenance requirements for all resources in the environment. Finally, each human entering the environment can also have a personal representative agent whose goal is to optimize conditions in the environment for the user.

- **Automated construction**:

  The application domain of automated construction involves the assembly of large-scale structures, such as terrestrial buildings, planetary habitats, or in-space facilities. Such domains need heavy lifting capabilities, as well as precise, dexterous manipulation to connect parts together. A motivating scenario is that of assembling the steel structure of a large building. In such cases, a large crane is used to lift beams and move them near their destinations; a worker near the destination uses hand signals to guide the crane operator; when the beam is close enough, the worker grabs the end and moves it into place. The domain of automated construction, however, is not limited to terrestrial work. Future space facilities, characterized by their immense size and the difficulties of human construction in space will be assembled in part by groups of autonomous heterogeneous robots.

- **Robotic educational and entertainment systems**:

  Robotic toys, educational tools, and entertainment systems are rapidly gaining popularity. Many of these systems will require coordinated efforts by multiple robots. An example in this domain is robotic soccer.

- **Automated production plants**:

  A growing trend in production plants is automation. In order to increase production, decrease labor costs, improve efficiency, increase safety, and improve quality in general, more and more industries are seeking to automate their production facilities. This trend demands efficient and robust coordination of heterogeneous multirobot systems.

- **Robotic exploration of hazardous environments**:

  Exploration of hazardous environments has long been a problem demanding robotic solutions. Some examples of robotic hazardous environment exploration are exploration of extra-planetary regions, exploration of volcanic regions, exploration of disaster zones, and exploration of minefields.

- **Robotic cleanup of hazardous sites**:
  Robots continue to play an important role in cleanup of hazardous sites. Some examples in this domain are robotic minesweeping, robotic cleanup of nuclear waste, and robotic cleanup of disaster zones.

- **Agricultural Robots**

  Many groups involved with agricultural work are now seeking automated solutions to their labor problems. Due to the long hours, hard physical work in rough conditions, and tedious and repetitive nature of some of the tasks in this domain, a growing decline in the available pool of labor has become evident. Spraying fields, harvesting, moving containers, and sorting plants are some examples of tasks that can be automated in this domain. For many of these tasks, coordinated teams of robotic agricultural machines promise to provide efficient solutions.

These application domains demand high quality performance from multirobot systems. Hence, a general multirobot solution for these application domains must fulfill many requirements. The following characteristics are thought to be an exhaustive list of these requirements:

C1 (**Robustness**): *Robust to robot/agent failure, or no single point of failure for the system*. This is an important characteristic since many of the applications rely on continued progress even if some components in the system fail. Some application domains expect that several agents will malfunction or be destroyed during operation, and still require the overall mission to be completed in the best way possible given the remaining resources.

C2 (**Optimized**): *Optimized response to dynamic conditions*. This characteristic is desirable in general, and required in some domains. Since many of the application domains involve dynamic conditions, the ability to opportunistically optimize the system response to these conditions is necessary for success.

C3 (**Speed**): *Quick response to dynamic conditions*. Often in dynamic environments, a key to successful task execution is the capability to respond quickly to the dynamic conditions. If information always needs to be channeled to another location for

plan modification, conditions can change too rapidly for the planning to keep up.

C4 (**Extensibility**): *Easily extendable to accommodate new functionality*. A key characteristic to building a generalized system that can evolve with the needs of the different applications is the ability to easily add and remove functionality as needed. This is identified as extensibility.

C5 (**Comm**): *Ability to deal with limited and imperfect communication*. In general, many application domains cannot realistically guarantee perfect communication among all agents at all times. Hence, any generalized coordination architecture should be robust to communication failures and limits in range of communication.

C6 (**Resources**): *Ability to reason about limited resources.* The ability to reason about the limited resources available in a robotic system is very important for optimization purposes. This is because optimized resource utilization is a crucial component of optimized multirobot coordination.

C7 (**Allocation**): *Optimized allocation of tasks.* A key difficulty in coordinating multiple robots is deciding who does what. Thus, task allocation is an important factor in the architectural design.

C8 (**Heterogeneity**): *Ability to accommodate heterogeneous teams of robots.* Many architectures assume homogeneity for ease of planning. The coordination problem is more difficult if the robots are heterogeneous. A successful architecture will be able to accommodate any team regardless of its homogeneity or heterogeneity.

C9 (**Roles**): *Optimized adoption of roles*. In many architectures robots are restricted to being able to play only a single role at any given time. Yet, they possess the resources to be able to play more than a single role. Optimized role adoption will enable robots to play as many roles as required at any given time based on resource availability, and also allow robots to change in and out of different roles as conditions change.

C10 (**New Input**): *Ability to dynamically handle new tasks, resources, and roles*. In many dynamic application domains, the demands on the robotic system can change during operation. Hence, it may become necessary to assign new tasks, change existing tasks, add new resources, or introduce new roles. All of this should be supported by the architecture.

C11 (**Flexibility**): *Easily adaptable for different applications*. Since different applications will have different requirements, a general architecture will need the ability to be easily reconfigured for the different problems it proposes to solve. Instructions and advice on how to reconfigure the architecture for different applications will also be useful.

C12 (**Fluidity**): *Easily able to accommodate the addition/subtraction of robots during operation*. Several applications could require the ability to introduce new robots into the system during operation. Similarly, robots could be taken out or malfunction during task execution. A successful architecture will be able to support such events gracefully.

C13 (**Learning**): *On-line adaptation for specific applications*. While a generalized system is often more useful, its application to specific applications usually requires some tuning. The ability to tune relevant parameters automatically in an on-line fashion is thus a very attractive feature that can save a lot of effort in porting the architecture to different applications.

C14 (**Implementation**): *Implemented and proven on physical system*. As with any claim, a proven implementation is far more convincing. Moreover, successful implementation of an architecture on a robotic system requires discovering and solving many details that are not always apparent in simulation and software systems.

The matrix below illustrates the requirement characteristics of the multirobot applications described above:

Application	*Robustness*	*Optimized*	*Speed*	*Extensibility*	*Comm*	*Resources*	*Allocation*	*Heterogeneity*	*Roles*	*New Input*	*Flexibility*	*Fluidity*	*Learning*	*Implementation*
Autonomous colonies	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Urban reconnaissance	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Urban search and rescue	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Warehouse management	✓	✓	✓	✓		✓	✓	✓	✓	✓			✓	✓
Intelligent environments	✓	✓	✓	✓		✓		✓		✓	✓	✓	✓	✓
Automated construction	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓
Education and entertainment	✓	✓	✓	✓				✓			✓		✓	✓
Production plants	✓	✓	✓	✓		✓	✓	✓	✓		✓		✓	✓
Hazardous Exploration	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓		✓
Clean-up of hazardous sites	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓		✓
Agricultural Robots	✓	✓	✓	✓	✓	✓		✓			✓	✓	✓	✓

Table 1: Array of required characteristics for multirobot applications

It is clear that any multirobot coordination architecture needs to accommodate all of the above characteristics in order to be successful in a wide range of application domains. Therefore, designing a robust control-architecture for multirobot systems is a highly challenging task.

This work describes a novel market-based approach to multirobot coordination that aims to satisfy all of the above characteristics. The related work is explored next in section 2. Section 3 describes the problem to be solved and section 4 explains the proposed technical approach. Work to date is highlighted in section 5 and conclusions and future work are outlined in section 6. The document concludes with acknowledgements in section 7 and references in section 8.

2. Related Work

The past decade has witnessed a growing emphasis in research topics highlighting coordination of multirobot systems. This emphasis is generated by the increasing demand for automation in application domains where a single robot is no longer capable of performing the necessary tasks, and/or multiple robots can accomplish the same tasks more efficiently. Controlling a multirobot system is more complicated than controlling a single robot, not only because of the increased number of robots to be controlled, but also due to the added complication of requiring the robots to work together in a coordinated and intelligent manner to achieve assigned goals. Dynamic environments, malfunctioning robots, and multiple user requirements

add to the complexity of the multirobot coordination problem. Matarić [44] explores some of these issues, and presents a summary of some of the principal efforts in this field of research.

2.1. Centralized Approaches

Simply increasing the number of robots assigned to a task does not necessarily solve a problem more efficiently; multiple robots must cooperate to achieve efficiency. The difficulty arises in coordinating many robots to perform a single, global task. One approach is to consider the robot team to be a single robot "system" with many degrees of freedom. That is, a single robot or central computer is designated as the "leader" and is responsible for planning the actions of the entire group. This leader coordinates the group optimally to perform the specified task. The members of the group convey relevant information to the leader, and execute the plans generated by the leader. Some examples of such centralized approaches can be found in work done by Caloud et al. [14], Chaimowicz et al. [16], Jensen and Veloso [35], Brummit and Stentz [11], Simmons et al. [69], Švestka and Overmars [74], and Burgard et al. [12].

The principal advantage of such centralized approaches is that optimal plans can be produced. The leader can take into account all the relevant information conveyed by the members of the team and generate an optimal plan for the team. However, centralized approaches suffer from several disadvantages.

Optimal coordination is computationally difficult—the best known algorithms are exponential in complexity. Thus, an approach striving to compute globally optimal solutions becomes intractable for teams larger than a few. Additionally, the approach assumes that all relevant information about the robots and their environment can be transmitted to a single location for processing and that this information does not change during the time that an optimal plan is constructed. These assumptions are unrealistic for problems in which the environment is unknown and/or changing, communication is limited, and robots behave in unpredictable ways. Also, the system response to changes in the environment is sluggish since all relevant information must be conveyed to the leader before any action can be taken. Another weakness with this approach is that it produces a highly vulnerable system. That is, if the leader (the central planning unit) malfunctions, a new leader must be available or the entire team is disabled. Due to this reason, design of the architecture is made more complex because the designer must decide how many agents should be capable of being leaders. If all agents are able to be leaders, the potential for wasted resources is high since only a single leader is usually required at any given time. However, if only one agent is able to be a leader, the system is made highly vulnerable. Finally, the approach often has stringent and heavy communication requirements because information is required from all agents in order to compute optimal plans.

2.2. Distributed Approaches

Local and distributed approaches address the problems that arise with centralized, globally coordinated methods by distributing the planning responsibilities among all members of the team. Each robot operates largely independently, acting on information that is locally available through its sensors. Thus, each robot plans its course of action based on its local observations.

This allows fast response to dynamic conditions and decreases the communication requirements. A robot may coordinate with other robots in its vicinity, perhaps to divide a problem into multiple sub-problems or to work together on a sub-task that cannot be accomplished by a single robot. Typically, little computation is required, since each robot need only plan and execute its own activities. Also, less stringent constraints on communication are required, since the robots only communicate with others in their vicinity. The robots are better able to respond to unknown or changing environments, since they sense and respond to the environment locally. Moreover, the system is more robust since the entire team's performance no longer depends on the guidance of a single leader. No single point of failure exists for distributed systems and the approach scales easily to

accommodate large numbers of robots. The approach works best for problems that can be decomposed into largely unrelated sub-problems, or problems for which a desired group behavior results from the aggregate of individual behaviors and interactions.

Many research efforts have modeled distributed systems inspired by biology [for example, 3, 10, 19, 20, 46, and 66]. Others have designed systems based on fluidics and similar physics-based concepts [for example, 6, 18, 23, 82, and 88]. Some have chosen to pursue rule-based, heuristic-based and model-based approaches [for example, 22, 29, and 76]. Economy-based models have inspired still others [for example, 30, 56, 71, 72, and 85]. In general, cooperation in distributed approaches can be divided into two categories: fortuitous cooperation and planned cooperation. Work done by Arkin and Balch [3], Matarić [46], Brooks [10], Schneider-Fontán and Matarić [66, 67], Arkin [2], Parker [52, 53], Maio and Rizzi [42], Goldman and Rosenschein [29], Desai et al. [23], Böhringer et al. [6], Alur et al. [1], Osawa [49], and Pagello et al. [50] are examples of distributed systems in the fortuitous cooperation category. Some examples in the planned cooperation category are work done by Kaminka and Tambe [37], Decker and Lesser [22], Tambe et al. [78], Tambe [76], Weiß [84], Jennings and Kirkwood-Watts [34], Veloso et al. [81], Noreils [48], Bonasso et al. [7], Stentz and Dias [72], Golfarelli et al. [30], Sandholm [56], Smith [71], Gibney et al. [28], Collins et al. [21], and Jennings and Arvidsson [33]. All of these approaches aim to solve the multirobot/multiagent coordination problem in a distributed manner so that the difficulties inherent in a centralized system are circumvented.

However, the principal drawback of distributed approaches is that they often result in highly sub-optimal solutions because all plans are based solely on local information. Stentz and Dias [72] propose a market-based approach which aims to opportunistically introduce pockets of centralized optimal planning into a distributed system, thereby exploiting the desirable properties of both distributed and centralized approaches.

2.3. Economic Approaches

Smith [71] first introduced the concept of using an economic model to control multiagent systems as the Contract Net protocol. Many groups have since adopted similar strategies for controlling multiagent systems. Work done by Krovi et al. [39], Faratin et al. [26], Jung et al. [36], Brandt et al. [8], Wellman and Wurman [85], Smith [71], Gibney et al. [28], Collins et al. [21], Jennings and Arvidsson [33], Sandholm [56], and Sycara and Zeng [75] are examples of economy-based sofware-agent systems. In contrast, work done by Laengle et al. [40], Simmons et al. [70], Dias and Stentz [24], Gerkey and Matarić [27], and Golfarelli et al. [30] are examples of economy-based control-architectures applied to multirobot systems. Many characteristics differentiate software-agent domains from situated-agent (robotic) domains. Some principal differences between robotic systems and software systems are highlighted next.

Tasks assigned to robotic agents can vary significantly from tasks in software domains. Also, robotic agents often deal with more restricted resources and robotic systems often have to deal with more restricted communication. Failures occur with higher frequency and in a wider variety in robotic systems. 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 (for example, one robot pushing another robot that is stuck, two robots cooperating to lift a heavy obstacle, etc.). Thus, controlling multirobot systems can be a significantly different problem compared to controlling multiple software agents.

Economic approaches are not without their disadvantages. Negotiation protocols, mapping of task domains to appropriate cost functions, and introducing relevant de-commitment penalty schemes can quickly complicate the design of a control-architecture. Furthermore, some negotiation schemes can drastically increase communication requirements.

2.4. Optimization Techniques

Many research efforts have focused on developing optimization techniques applicable to distributed multiagent systems. Sandholm and Lesser [57], Sandholm et al. [62], and Excelente-toledo et al. [25] have proposed methods of allowing breaching of contracts as an optimization mechanism in economy-based approaches. Sandholm and Suri [59] examine the use of combinatorial auction schemes for optimizing negotiation. Many groups have investigated learning methods applied to multiagent systems. For example, Prasad et al. [54], Matarić [43, 45], Haynes et al. [31], Balch [5], Tan [79], Schmidhuber [64], Schaerf et al. [63], Zeng and Sycara [87, 86], Stone and Veloso [73], Schneider et al. [65], Weiß [83], and Brauer and Weiß [9].

Others have pursued methods of coalition formation to optimize multiagent coordination. Examples are work done by Sandholm and Lesser [58], Sandholm et al. [60], Zlotkin and Rosenschein [89], and Shehory and Kraus [68]. Tambe [77] proposes tracking behavior of other agents as an optimization technique. Rosin and Belew [55], and Matos and Sierra [47] adopt evolutionary methods for enhancing performance. Other optimization techniques have been proposed by Lux and Marchesi [41], Huber and Durfee [32], Castelfranchi and Conte [15], Panzarasa and Jennings [51], Sandholm and Lesser [61], Khuller et al. [38], Cheng et al. [17], and Andersson and Sandholm [2].

However, to date, no group has designed a control architecture that combines all necessary elements for coordinating multiple robots engaged in executing complex tasks in highly dynamic environments, in a robust and efficient manner.

2.5. Matrix of Technical Problems Solved in Related Work

The following matrix illustrates the accomplished characteristics of the best-known multirobot coordination architectures developed to date (Note that only publications pertaining to the design of a complete multirobot coordination architecture were evaluated in this matrix):

C1 (**Robustness**):	Robust to robot/agent failure, or no single point of failure
C2 (**Optimized**):	Optimized response to dynamic conditions
C3 (**Speed**):	Quick response to dynamic conditions
C4 (**Extensibility**):	Easily extendable to accommodate new functionality
C5 (**Comm**):	Ability to deal with limited and imperfect communication
C6 (**Resources**):	Ability to reason about limited resources
C7 (**Allocation**):	Optimized allocation of tasks
C8 (**Heterogeneity**):	Ability to accommodate heterogeneous teams of robots
C9 (**Roles**):	Optimized adoption of roles
C10 (**New Input**):	Ability to dynamically handle new tasks, resources, and roles
C11 (**Flexibility**):	Easily adaptable for different applications
C12 (**Fluidity**):	Easily able to accommodate the addition/subtraction of robots during operation
C13 (**Learning**):	On-line adaptation for specific applications
C14 (**Implementation**):	Implemented and proven on physical system (No - no implementation, PS - partially implemented in simulation, S - simulation or software agents, P - partially implemented on a robotic system, Yes - fully implemented on a robotic system)

Note: For C1 to C13:
No = Not accomplished, **P** = Partially accomplished, **Yes** = Fully accomplished

Ref #	*Robustness*	*Optimized*	*Speed*	*Extensibility*	*Comm*	*Resources*	*Allocation*	*Heterogeneity*	*Roles*	*New Input*	*Flexibility*	*Fluidity*	*Learning*	*Implementation*
1	**No**	**No**	**Yes**	**Yes**	**Yes**	**No**	**No**	**P**	**No**	**No**	**No**	**P**	**No**	**Yes**
3	**Yes**	**No**	**Yes**	**Yes**	**Yes**	**No**	**No**	**Yes**	**No**	**No**	**Yes**	**Yes**	**Yes**	**S**
4	**Yes**	**No**	**Yes**	**Yes**	**Yes**	**No**	**No**	**Yes**	**No**	**No**	**Yes**	**Yes**	**Yes**	**Yes**
6	**No**	**No**	**No**	**P**	**Yes**	**No**	**P**	**P**	**P**	**No**	**No**	**No**	**No**	**Yes**
11	**No**	**Yes**	**No**	**No**	**No**	**No**	**Yes**	**No**	**No**	**No**	**No**	**P**	**No**	**S**
12	**P**	**P**	**P**	**P**	**No**	**No**	**Yes**	**P**	**No**	**No**	**No**	**P**	**No**	**Yes**
13	**No**	**No**	**P**	**Yes**	**P**	**P**	**No**	**Yes**	**No**	**P**	**No**	**P**	**Yes**	**PS, P**
14	**P**	**P**	**P**	**Yes**	**Yes**	**P**	**P**	**Yes**	**No**	**P**	**Yes**	**Yes**	**No**	**P**
16	**P**	**P**	**Yes**	**P**	**P**	**No**	**P**	**Yes**	**P**	**No**	**P**	**P**	**No**	**Yes**
18	**P**	**P**	**P**	**No**	**Yes**	**No**	**P**	**No**	**No**	**No**	**No**	**No**	**No**	**No**
21	**No**	**P**	**P**	**P**	**No**	**P**	**P**	**Yes**	**No**	**P**	**Yes**	**P**	**No**	**No**
22	**No**	**No**	**No**	**Yes**	**No**	**Yes**	**No**	**Yes**	**No**	**No**	**Yes**	**No**	**No**	**S**
23	**No**	**P**	**Yes**	**No**	**Yes**	**No**	**No**	**No**	**No**	**No**	**No**	**No**	**No**	**S**
25	**P**	**No**	**P**	**P**	**No**	**No**	**No**	**No**	**No**	**P**	**No**	**P**	**No**	**S**
27	**Yes**	**No**	**Yes**	**Yes**	**P**	**Yes**	**No**	**Yes**	**No**	**P**	**Yes**	**Yes**	**No**	**Yes**
29	**No**	**No**	**No**	**P**	**Yes**	**No**	**No**	**Yes**	**No**	**No**	**P**	**P**	**No**	**S**
30	**No**	**No**	**P**	**Yes**	**Yes**	**No**	**No**	**Yes**	**No**	**P**	**P**	**No**	**No**	**S**
32	**P**	**No**	**No**	**No**	**Yes**	**No**	**No**	**P**	**No**	**No**	**P**	**P**	**No**	**S**
33	**P**	**P**	**P**	**P**	**P**	**P**	**Yes**	**P**	**No**	**P**	**No**	**No**	**No**	**No**
34	**Yes**	**P**	**Yes**	**Yes**	**Yes**	**No**	**No**	**Yes**	**No**	**P**	**P**	**Yes**	**No**	**Yes**
35	**No**	**P**	**P**	**Yes**	**No**	**No**	**P**	**Yes**	**No**	**No**	**Yes**	**P**	**No**	**S**
36	**P**	**P**	**P**	**Yes**	**P**	**P**	**P**	**Yes**	**No**	**No**	**P**	**P**	**No**	**S**
40	**P**	**Yes**	**Yes**	**Yes**	**Yes**	**No**	**Yes**	**Yes**	**No**	**P**	**Yes**	**Yes**	**No**	**Yes**
42	**No**	**No**	**Yes**	**No**	**Yes**	**No**	**No**	**No**	**No**	**No**	**No**	**P**	**No**	**S**
48	**P**	**P**	**Yes**	**Yes**	**Yes**	**P**	**P**	**Yes**	**No**	**P**	**Yes**	**P**	**No**	**Yes**
49	**No**	**No**	**P**	**P**	**Yes**	**No**	**P**	**P**	**No**	**No**	**P**	**No**	**No**	**S**
50	**P**	**No**	**Yes**	**P**	**Yes**	**No**	**No**	**P**	**No**	**No**	**P**	**No**	**No**	**S**
52	**Yes**	**P**	**P**	**Yes**	**Yes**	**No**	**P**	**Yes**	**P**	**P**	**Yes**	**Yes**	**Yes**	**Yes**
54	**No**	**P**	**P**	**P**	**No**	**No**	**No**	**Yes**	**P**	**No**	**No**	**No**	**Yes**	**S**
56	**No**	**P**	**Yes**	**Yes**	**Yes**	**P**	**Yes**	**Yes**	**No**	**P**	**Yes**	**P**	**No**	**P**
67	**Yes**	**P**	**P**	**P**	**Yes**	**No**	**P**	**P**	**No**	**P**	**No**	**Yes**	**No**	**Yes**
68	**No**	**P**	**No**	**Yes**	**No**	**Yes**	**P**	**Yes**	**No**	**No**	**Yes**	**No**	**No**	**No**
69	**No**	**No**	**No**	**P**	**No**	**No**	**Yes**	**Yes**	**No**	**P**	**P**	**No**	**No**	**Yes**
70	**Yes**	**P**	**Yes**	**Yes**	**Yes**	**P**	**Yes**	**P**	**No**	**No**	**P**	**Yes**	**No**	**P**
71	**P**	**P**	**Yes**	**Yes**	**Yes**	**P**	**Yes**	**Yes**	**No**	**P**	**Yes**	**Yes**	**No**	**No**
74	**No**	**P**	**P**	**No**	**Yes**	**No**	**Yes**	**No**	**No**	**No**	**No**	**No**	**No**	**S**
75	**Yes**	**Yes**	**Yes**	**Yes**	**P**	**No**	**Yes**	**Yes**	**No**	**P**	**P**	**Yes**	**Yes**	**S**
76	**Yes**	**Yes**	**Yes**	**Yes**	**P**	**No**	**Yes**	**Yes**	**P**	**P**	**Yes**	**Yes**	**P**	**P**
78	**Yes**	**Yes**	**Yes**	**Yes**	**P**	**No**	**Yes**	**Yes**	**P**	**P**	**Yes**	**Yes**	**Yes**	**P**
81	**Yes**	**P**	**Yes**	**No**	**Yes**	**No**	**P**	**P**	**No**	**No**	**No**	**No**	**No**	**Yes**
82	**No**	**No**	**No**	**No**	**Yes**	**No**	**P**	**No**	**No**	**No**	**No**	**No**	**No**	**Yes**
85	**No**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**No**	**P**	**Yes**	**P**	**No**	**No**
88	**No**	**P**	**P**	**No**	**Yes**	**No**	**P**	**No**	**No**	**No**	**No**	**No**	**Yes**	**Yes**

Table 2: The coverage of characteristics required by multirobot applications by multirobot coordination architectures designed to date

2.6. Summary of Prior Work

A closer examination of the architectures that satisfied many of the desired characteristics is useful. Arkin [3] provides a biologically inspired architecture where individual agents make reactive decisions based on motor-schema. Introducing cooperation through recruitment behaviors enhances the resulting fortuitous group-behavior. This eliminates the necessity for direct communication between agents. However, this system is incapable of optimizing its response to dynamic conditions. The agents also do not explicitly reason about optimal use of their limited resources, optimal adoption of roles, and optimal task distribution among the group. Furthermore, the group of agents cannot easily accommodate new tasks, resources, and roles assigned to the group. Finally, this work does not include implementation results from a robotic system – results are all in simulation.

Caloud et al. [14] introduce another successful architecture, "GOPHER". The architecture is broken down into four layers: task decomposition, task allocation, motion planning, and execution. The task allocation process is partially centralized by involving a central task processing system (CTPS) which announces tasks to all available robots within communication range. There is no apparent plan however to recover from failure or malfunction in the CTPS. Moreover, although robots make bids for different tasks offered by the CTPS, it is not clear how these bids are formed, or on what basis the costs are computed. Furthermore, any robot in the midst of executing a task will not participate in any other transactions until it has completed its current tasks. These characteristics detract from optimal resource utilization and optimized reaction to dynamic conditions. Also, no explicit reasoning about optimal utilization of limited resources and corresponding adoption of roles is evident. Finally, the implementation of the architecture is at preliminary stages, and no method of on-line adaptation for specific applications is discussed.

Chaimowicz et al. [16] present an architecture for tightly coupled multirobot coordination. The architecture is based on the concept of at least one leader being identified at any give time within a group, and the others being followers. Some of the followers can be leaders for other agents in the group – so a hierarchy could exist. (In the implementation only a single leader was present at any time). Leadership can be changed by request and by resignation. Conflicts are resolved using a priority based approach. If any deadlock is detected, command is relinquished to a human operator. No discussion is presented about what would happen if a leader is disabled before it can resign. Thus, the system is not fully robust. Also, since a robot could be assigned as a leader, even though it isn't optimally placed to become a leader, due to the current leader's resignation, the system will not always produce an optimal response to dynamic conditions. No consideration is provided about optimal management of resources. The roles are limited to leaders and followers. On-line optimization/adaptation schemes are not discussed, and extensibility, flexibility and fluidity of the system are limited.

Gerkey and Matarić [27] present "MURDOCH"; a completely distributed, resource-centric, publish/subscribe communication model. A key feature in this approach is that all communication is resource-centric, and never name-based. Thus, the claim is that all messages are addressed in terms of the resources required to do the task. All tasks are allocated based on a single-round auction scheme. The auctioneer determines a winner and notifies the bidders. The winner is awarded a time-limited contract to complete the task. The auctioneer is responsible for monitoring the progress of the task execution. To do this, the auctioneer periodically sends contract renewal messages to the winner, which sends back corresponding acknowledgements. These messages will have to be addressed by name and will increase the communication requirements of the system since the auctioneer and winner will have to remain within communication range (or periodically return to positions within communication range) to renew the contract. Furthermore, since the auctioneer assumes a fault if a renewal message is not acknowledged and reassigns the task, several robots could attempt to complete the same task if acknowledgements are not received on time or some acknowledgement messages are lost. The instantaneous greedy

task scheduling in MURDOCH does not allow for opportunistic optimization. Also, no discussion of on-line adaptation is presented.

Laengle et al. [40] introduce KAMARA, a distributed control-architecture for controlling the different components of a complex robot. This architecture could arguably be applied to the multirobot coordination problem. The architecture is based on the concept of generating an agent that represents each component of the robot. Each agent is composed of a communicator that handles communication, a head that is responsible for the planning and action selection, and a body which handles execution. Each body consists of one or more executive components which can be agents themselves. The agents can form teams to execute cooperative tasks when tight coordination is not necessary. If tight coordination is required, a special agent is generated which oversees the coordination of the agents involved by seizing control of these agent bodies from their heads. Task allocation occurs via negotiation where mediation is carried out by a selected agent in the candidate group for executing the task. Although disabled agents are compensated for because they cannot participate in the negotiation and hence cannot be assigned tasks, no method of ensuring completion of tasks assigned to the disabled agent is presented. Moreover, each agent is only able to execute a single task at a time. This can be highly sub-optimal in multirobot coordination because either resource utilization will be non-optimal, or an agent will be required to represent each resource on each robot which will drastically increase the required negotiation between agents. Also, the presented architecture has no method for on-line optimization.

Parker [52] designed ALLIANCE to be a fault-tolerant and adaptive multirobot coordination architecture. Essentially a behavior-based system, ALLIANCE has the added benefit of motivational behaviors such as impatience and acquiescence which prompt the robots to complete tasks when other robots fail to execute them, and to give up on tasks they cannot complete. While these motivational behaviors allow robot teams to be fault tolerant and adaptive, they do not enable the robots to respond to dynamic conditions in a quick and opportunistically optimal manner. Furthermore, the robots do not reason about the limited resources available to them and attempt to optimize utilization of these resources. No allocation of tasks is performed in this architecture. Instead, the high-level tasks are programmed into the behavior sets of the robots. This scheme does not promote optimized task allocation, and doesn't allow new types of tasks to be dynamically assigned. Parameterized learning methods are explored for on-line optimization. In Parker's discussion of other multirobot coordination schemes, her main argument against negotiation schemes is that they have not been proven on robotic systems. The proposed work for this Ph.D. thesis aims to overcome Parker's challenge.

"Teamcore", developed by Tambe et al. [78] is the most successful architecture included in Table 2 in terms of satisfying the largest number of the identified criteria. The general idea of this approach is to provide the architecture with general-purpose teamwork coordination capabilities and to adapt these capabilities for specific applications via machine learning. The architecture is formed by generating Teamcore proxies to represent each participant in the group (the participants can be robots, humans, or software agents). Each proxy is equipped with a general teamwork model, STEAM [76], which automates its coordination with other proxies on its team. STEAM is based on a hierarchical reactive plan architecture, with teamwork capabilities inserted as reactive team plans which instantiate joint intentions for the team. The proxies can then adapt at four different levels: autonomy, execution, monitoring, and information requirements. While this system is highly adaptable and successful in satisfying many of the criteria in Table 2, it lacks the ability to reason about and optimize the use of limited resources available to the robots. Further, it has not been implemented and proven on a robotic system.

In summary, although many multirobot coordination architectures have been implemented on robotic systems with varying levels of success, no architecture that satisfies all of the criteria for efficient multirobot coordination in dynamic domains has been designed and implemented to date. Mainly lacking are architectures that optimize

resource utilization and role adoption in robotic systems. Furthermore, no in-depth analysis has been carried out to investigate the applicability of economic methods to the multirobot coordination problem.

3. Problem Statement

Clearly, the multirobot problem has not been completely solved for dynamic domains where highly sub-optimal solutions are very costly and single points of failure are unacceptable. That is, no control-architecture currently exists which can accomplish all of the characteristics outlined in the previous section. Mainly lacking in the prior work is a multirobot control-architecture that reasons in an optimized fashion about resource utilization and role adoption while maintaining the ability to respond quickly and efficiently to dynamic conditions. This work outlines the design and implementation of a novel market-based architecture to enable a detailed analysis of the effectiveness of applying economic strategies to solve the multirobot control problem in dynamic environments. The proposed architecture will be designed to satisfy all of the required characteristics highlighted in Table 1.

A principal advantage of the proposed market-based architecture is its ability to opportunistically optimize resource utilization. A simple example is illustrative. Consider a system where we want two items to be retrieved. Two robots are available to do the task.

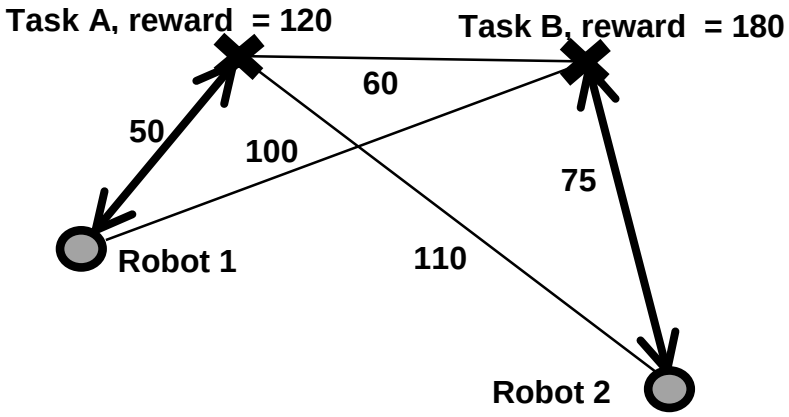

Figure 1: An illustration of simple reasoning

The robots incur costs in driving, as indicated by the numerical labels in Figure 1. The robots bid on each of the tasks. The robots are only smart enough to reason about single city tours. Robot 1 bids the cost plus a profit of 20 for task A (120). Robot 2 cannot compete for this task since its costs alone are 220. Robot 1 wins task A. For similar reasons, Robot 2 wins Task B. Both robots are happy, since each makes a profit. The user is happy, since the task is accomplished with a reasonable solution.

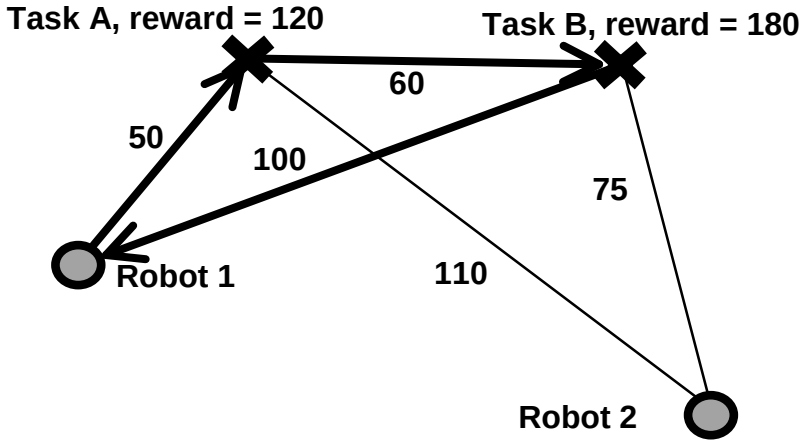

Figure 2: An illustration of more complex reasoning

Now assume that Robot 2 becomes smarter because it acquired more computer power or received a better algorithm. Robot 2 suggests to Robot 1 that it perform Task B immediately after Task A before returning to its base as illustrated in Figure 2. If Robot 2 can subcontract Task B, it will save 150 in cost. Therefore, it will do so if it pays out less than 150. Robot 1 considers the request and determines that the new route will add 110 in cost. Therefore, it will take the subcontract if it will receive more than 110. The two negotiate and agree on 130. Both robots are happy, since they increased their profits, and we are happy because the global task was accomplished at lower cost. Note that Robot 2 was a leader; it didn't do anything except generate and sell a plan. It didn't lead by coercion; instead, it used the additional profit to "buy off" the participant. Good plans generate more profits that can be used to gain participants. Note further that this plan could have been proposed to both robots by a third robot playing a "leader role".

The market-based architecture is not, however, the optimal solution for all multirobot application domains. One concern is that there may be difficulties designing appropriate cost and revenue functions to represent some problems accurately. Furthermore, centralized approaches will work best for multirobot applications in static environments where there are only a few robots required to execute a task. Similarly, reactive methods will provide less complicated control-architectures for multirobot applications where optimality is not a paramount concern. Thus, an important component of the proposed work will be to identify what multirobot problems are best solved by the proposed market-based architecture.

4. Technical Approach

It is now instructive to examine in detail the proposed approach to solving the multirobot coordination problem described above. A crucial component of the proposed solution is a control-architecture designed to inherit the efficacy and flexibility of the market. This market-based architecture is detailed next.

Consider an economic system for coordinating robots. An economy is essentially a population of agents (i.e., citizens) producing a global output. The agents coordinate with each other to produce an aggregate set of goods. Centralized economies suffer from an inability to gather all salient information, uncertainty in how to optimize with it, and unresponsiveness to changing conditions. Additionally, if economic output is divided equally amongst the entire population, individuals have little incentive to work harder or more efficiently than what is required to minimally comply with the economic plan. Individual input is de-coupled from individual output. The net effect is a sluggish, brittle, and inefficient economy.

Market economies are generally unencumbered by centralized planning; instead, individuals are free to exchange goods and services and enter into contracts as they see fit. Despite the fact that individuals in the economy act only to advance their own self-interests, the aggregate effect is a highly productive society. Individuals are often in the best position to understand their needs and the means to satisfy them. Thus, individuals reap the direct benefits of their own good decisions and suffer the direct consequences of their bad ones. At times they cooperate with other members of the society to achieve an outcome greater than that possible by each member alone. At times they compete with other members to provide goods or services at the lowest possible cost, thus eliminating waste and inefficiency. But at every turn, the individual members act solely to reap the greatest profit for themselves. This work proposes a method for applying these powerful mechanisms to the task of coordinating a team of robots. An important aspect of the market-based approach is that the robots will be self-interested. Note however that the robots will only be self-interested within the domain of capabilities allowed to them. For example robotic dishonesty will not be allowed in order to simplify architectural component design and overall system complexity. While other approaches have adopted cooperative models for robot coordination, it is not clear that cooperative motivations in general have any significant advantage over the proposed approach. Designing generic incentives to motivate cooperation in multirobot applications is not trivial, although such schemes would be better suited for some domains. Furthermore, it is not clear that cooperative incentives would not lead to interference between robots in some application domains. While designing appropriate cost and revenue models can also be non-trivial in some application domains, in many multirobot application domains the proposed approach provides a mechanism that inherits the many benefits of market mechanisms as described next.

4.1 The Market Mechanism[1]

Consider a team of robots assembled to perform a particular set of tasks. Consider further, that each robot in the team is modeled as a self-interested agent, and the team of robots as an economy. Thus, the goal of the team is to complete the tasks successfully while minimizing overall costs, since each robot will aim to minimize its individual cost and maximize its individual profit. A system such as this will inherit many desirable characteristics from the market mechanisms. Presented next is a brief examination of some of these characteristics.

4.1.1 Determining Revenues and Costs

Costs and revenues will dictate to a large extent the performance of a market-based approach. A function, **trev**, is needed to map possible task outcomes onto revenue values. Another function, **tcost**, is needed that maps possible schemes for performing the task onto cost values. As a team, the goal is to execute some plan **P** such that profit, **trev(P) – tcost(P)**, is maximized. Note that **trev** and especially **tcost** will not always be simple functions – they could potentially be complex functions involving statistical distributions and vectors of different components (for example, the cost function could be a combination of time spent, fuel expended, and CPU cycles utilized). The cost and revenue functions will be designed to reflect the nature of the application domain. Thus, these functions will need to reflect aspects such as priorities for task completion, hard deadlines for relevant tasks, and acceptable margins of error for different tasks.

But it is not sufficient to define just the revenue and cost functions for the team. These functions must provide a means for distributing the revenue and assessing costs to individual robots. Preferably, these individual revenues and costs are assigned based on factors over which the individuals have direct control. For example, if the task is to find and retrieve a set of objects, the team's revenue, **trev**, could be the number of objects retrieved (converted to a "cash" value), and the team's cost, **tcost**, could be the amount of energy consumed by the entire team to find the objects. The individual robot revenues and costs, **rrev** and **rcost**, could be the cash value of the number of objects turned in and the energy expended, respectively, by *that* individual.

Therefore, the sum of the individual revenues and costs equals the team's revenues and costs. However, the distribution is not even: individuals are compensated in accordance with their contribution to the overall task, based on factors that are within the control of the individual. An individual that maximizes its own personal production and minimizes its own personal cost receives a larger share of the overall profit. Note that the revenue/cost models and rules of engagement will be designed so that maximizing individual profit has the benevolent effect of moving the team toward the globally optimal solution. Therefore, by acting strictly in their own self-interests, individuals maximize not only their own profit but also the overall profit of the team. It is instructive to note that since costs and revenues drive its design, the architecture can be easily applied to many different applications. Another important feature of this architecture is its ability to accommodate several revenue providers. That is, a number of operators could connect to a single group of robots and offer the robots revenue in return for different services. This also prevents the revenue provider from becoming a single point of failure for the system. If for some reason the connection from the operator to the team of robots is disabled (a likely scenario in application domains such as urban reconnaissance), a different connection can be established and the mission can be completed.

4.1.2 The Role of Price and the Bidding Process

Robots receive revenue and incur costs for accomplishing a specific team task, but the team's revenue function is not the only source of income. A robot can also receive

[1] This approach was briefly introduced in 24, 72, and 80.

revenue from another robot in exchange for goods or services. For example, a robot may not be equipped to find objects for which the team function provides revenue, but it can transport the objects to the goal once they have been found. Therefore, this haulage robot provides a service to the robots that find the objects, and it receives payment for performing such a service.

In general, two robots have incentive to deal with each other if they can produce more aggregate profit together than apart—such outcomes are win-win rather than zero-sum. The *price* dictates the payment amount for the good or service. How is the price determined? Assume that robot **A** would like to purchase a service from robot **B**. Robot **B** incurs a cost **Y** for performing the service. Robot **A** can make an additional revenue of **X** if **B** performs the service for it. Therefore, if **X > Y**, then both parties have an incentive to execute the deal. But how should the composite profit, **X - Y**, be divided amongst the two parties? It may sound fair to split the winnings **(X - Y) / 2** by setting the price at **(X + Y) / 2**. But robots **A** and **B** may have other opportunities—they may be considering other deals that contend for the same money and resources. Since these factors may be hidden or complex, a common approach is to *bid* for a good or service until a mutually acceptable price is found. For example, robot **A** could start by bidding a price of **Y** (i.e., robot **A** receives the entire profit). Robot **B** could decline and counter with a bid of **X** (i.e., robot **B** receives the entire profit). The idea is to start by bidding a price that is personally most favorable, and then successively retreat from this position until a price is mutually agreed upon.

Note that a given robot can negotiate several potential deals at the same time. It begins by bidding the most favorable price for itself for all of the deals, successively retreats from this position with counter bids, and closes the first deal that is mutually acceptable. The deal will be mutually acceptable only if it results in increased profits for both robots. Thus, the deal will move the global solution towards its optimal. Note also that a deal can be multi-party, requiring that all parties agree before any part of the deal is binding. Since multiple rounds of negotiations could be time-consuming, time restrictions will have to be imposed on all bidding. These time restrictions can vary from bid to bid, and between application domains. Based on basic economic intuition, the negotiated price will tend toward the intersection of the supply and demand curves for a given service. If a service is in high demand or short supply, the price will be high. This information will prompt other suppliers to enter the fray, driving the price down. Likewise, if demand is low or supply high, the low price will drive suppliers into another line of business. Thus, price serves to match supply to demand. Note that in an efficient market with perfect information, the price will optimize the matching of supply and demand.

Finally, it is important to note that price and bidding are low bandwidth mechanisms for communicating aggregate information about costs. When consumers decide between purchasing apple juice or orange juice for breakfast, they do not analyze land acreage dedicated to both crops, the costs of producing each, the demand for each, and the impact of weather and pest infestations. Instead, they merely look at the price of each and weigh them against their own personal preferences. The price, however, *encodes* all of these factors in a concise fashion that enables them to make a locally optimal decision based on low-bandwidth information available at the point of sale.

4.1.3 Cooperation vs. Competition

As described in the previous section, robots interact with each other to exchange goods and services. Two robots are *cooperative* if they have complementary roles, that is, if both robots can make more profit by working together than by working individually. Generally, robot teams foster cooperation between members of different types (heterogeneous). For instance, a robot able to grasp and lift objects and a robot able to transport objects could team together to provide a pick-and-place service that neither one could offer independently.

Conversely, two robots are competitive if they have the same role; that is, if the amount of profit that one can make is negatively affected by the presence of the other robot. Generally, robot teams foster competition amongst members of the same type (homogeneous). For instance, two robots that are able to transport objects compete for the services of a given grasping robot, thus driving the price down. Either one could charge more money if the other were not present.

These delineations are not strict however. Subgroups of heterogeneous robots could form that provide a given service. These subgroups would compete with each other, thus providing an example where robots of different types compete rather than cooperate with each other. Heterogeneous robots could also compete if the same task can be accomplished in different ways. Conversely, two robots of the same type may cooperate by agreeing to segment the market. Homogeneous robots can also cooperate if accomplishing a specific task requires more than one robot. For example, several robots with grasping capability may need to cooperate in order to move a heavy object. The flexibility of the market-model allows the robots to cooperate and compete as necessary to accomplish a task, regardless of the homogeneity or heterogeneity of the team.

4.1.4 Self Organization

Conspicuously absent from the market is a rigid, top-down hierarchy. Instead, the robots organize themselves in a way that is mutually beneficial. Since the aggregate profit amassed by the individuals is directly tied to the success of the task, this self-organization yields the best results.

Consider a group of ten robots. An eleventh robot, **A**, offers its services as their leader. It does not become their leader by coercion or decree, but by convincing the group that they will make more money by following its advice than by acting individually or in subgroups. **A** does this by investigating "plans" for utilizing all ten robots. If **A** comes up with a truly good plan, it will maximize profit across the whole group. The prospective leader can use this large profit to bid for the services of the group members, and of course, retain a portion of the profit for itself. Note that all relevant robots will have to commit to the plan before it can be sold. The leader may be bidding not only against the individuals' plans, but also against group plans produced by other prospective leaders. Note that the leader acts both as a benevolent and a self-interested agent—it receives personal compensation for efforts benefiting the entire group.

But there is a limit to this organization. As the group becomes larger, the combinatorics become intractable and the process of gathering all of the relevant information to produce a good plan becomes increasingly difficult. A leader will realize this when it can no longer convince its subjects (via bidding for their services) to follow its plans. The leader role is discussed in more detail in section 4.4 . Centralized and distributed approaches are two extremes along a continuum. The introduction of leaders will allow the market-based approach to slide along this continuum in the direction of improved optimality in an opportunistic manner.

4.1.5 Learning and Adaptation

The robot economy is able to learn new behaviors and strategies as it executes its task. This learning applies to both individual behaviors and negotiations as well as to the entire team. Individual robots may learn that certain strategies are not profitable, or that certain robots are apt to break a contract by failing to deliver the goods or proper payment. Individuals may also learn successful bidding strategies or which deals to offer when. The robot team may learn that certain types of robots are in over-supply, indicated by widespread bankruptcy or an inability to make much money. Conversely, the robot team may learn that certain types of robots are in under-supply, evidenced by excessive profits captured by members of the type. Thus, the population can learn to exit members of one type and enter

members of another. Moreover, in this approach, successful agents are able to accumulate wealth and perpetuate their winning strategies because of their ability to offer higher payments to other agents.

One of the greatest strengths of the market economy is its ability to deal successfully with changing conditions. Since the economy does not rely on a hierarchical structure for coordination and task assignment, the system is highly robust to changes in the environment, including mal-functioning robots. Disabling any single robot should not jeopardize the system's performance. By adding escape clauses for "broken deals", any tasks undertaken by a robot that malfunctions can be re-bid to other robots, and the entire task can be accomplished. Also, escape clauses will provide the robots with further flexibility to adapt to dynamic conditions and accept more attractive offers that are generated, although some caution is necessary here to maintain stability and prevent cycles of broken deals. Thus, the market model allows the robots to deal with dynamic environments in an opportunistic and adaptive manner. Moreover, due to its distributed nature, the architecture can respond quickly (within the time constraints placed on bidding) to dynamic conditions. Finally, a key factor for efficiency and flexibility in the market-based architecture will be the agents' ability to efficiently manage their resource utilization and role adoption. This architecture will allow for efficient resource and role management, because each agent will be driven to make individual-rational decisions at any given time.

The strength and flexibility of the market-based architecture can be further illustrated by examining an example of its application in a multirobot task domain.

4.2 Example: Application of the proposed market approach to the automated warehouse domain

Controlling multiple automated pallet jacks in a warehouse is a challenging task. The underlying challenge is to provide optimal scheduling and coordination of the pallet jacks to maximize throughput and minimize labor costs. This involves maximizing the time spent by workers picking orders, the operation most difficult to automate, and minimizing their unproductive travel time. Thus, the workers will be dedicated to specific storage areas in the warehouse and the pallet jacks drive themselves from worker to worker in the necessary sequence to fill orders.

If the market architecture was applied to the automated warehouse domain, agents representing the pallet jacks will negotiate to win order filling, delivery, and other tasks the pallet jacks are capable of performing. Note that even though these are software agents, they represent situated agents and hence this problem will fall within the multirobot application domain. They will have to negotiate with agents representing restrictive areas on their routes so that they are able to make cost-effective bids and reach their goals with minimal delays. Agents representing the restrictive areas will be responsible for controlling traffic in these areas and thereby preventing collisions and delays. Agents representing human workers will negotiate on their behalf for loading, sorting, and other such tasks that will maximize the efficiency and utility of the laborers. Finally, agents representing the Warehouse Management System (WMS) will be responsible for announcing orders issued by the WMS and orchestrating efficient execution of other instructions from the WMS.

Each agent can choose to play one or more complementary roles at any point in time if the agent is managing the set of resources required by those roles. Maximizing individual profit will be each agent's motivation for choosing whether or not to adopt a particular role. The following roles will correspond to the duties of the different agents specified above:

- Loader – loading pallets
- Transporter – transporting pallets

- Order Filler – efficiently filling an order
- Coordinator – computing efficient plans for a group of agents
- Computer – utilizing computing resources for evaluation of a problem
- Traffic Cop – monitoring a given area to prevent collisions and traffic congestion

Let's assume that only pallet-jack agents and WMS agents will manage sufficient computing resources to adopt multiple roles. The agents representing laborers and restrictive areas will probably not have sufficient computing resources to manage more than a single role. In general, an agent will be able to assume more roles if it has access to more resources. Agents orchestrate optimal planning and scheduling by switching between these roles as necessary. For example, an agent assuming the Order Filler role could plan a sequence of routes through the warehouse assuming that there is no contention for resources (i.e., human loaders, intersections, one-way routes etc.). If it can't "buy" everything it needs, it will need to modify its plans. On the other hand, if it can get what it needs but the prices are high, then that is an opportunity to assume the role of a Coordinator. As a Coordinator it can coordinate Order Filler plans with Loader and Transporter plans to produce a globally more efficient solution and sell the results to the participants. The more computing time/resources available, the more complicated that reasoning could be, involving many agents and resulting in even greater global efficiency. The system is thus an "anytime" system, producing good results that get better if time permits.

Consider the case of two pallet jacks (**J1** and **J2**) employed in a warehouse. The warehouse also employs two human laborers (**H1** and **H2**) for picking orders, and a Warehouse Management System (**W**) that oversees order picking. **H1** is assigned to pick orders from a storage location (**S1**) that contains tomatoes and cabbages, while **H2** is assigned to a storage area (**S2**) containing tomatoes, eggs, and potatoes. Two intersections (**I1** and **I2**) and one passageway (**P**) in the warehouse only allow for a single pallet jack to pass though at a time. In this scenario the above warehouse components would be represented by the agents **j1**, **j2**, **h1**, **h2**, **w**, **i1**, **i2**, and **p** respectively. Assume all costs are time-based. For example, the cost for travelling from a start position to a goal is directly proportional to the time it takes to move from start to goal. For simplicity, let as assume the WMS designates sufficiently high, equal rewards for all assignments/orders. Note that in reality, the reward offered for each assignment can vary depending on the urgency of the order. Also assume that the agents determine the bid price as a percentage of the difference between the offered reward and the cost. Let the cost of loading one unit of any of the produce also be fixed. Figure 3 below shows the layout of a simplistic warehouse, the costs of all possible routes, the locations of the key components in the warehouse, and the different items stored in the storage areas. The most likely scenario would be that agents **j1** and **j2** reside on the corresponding pallet jacks, while all other agents reside on the central computing system which supports **W**. It is assumed that communication is possible at all times between all agents, either point-to-point or routed through a central computer.

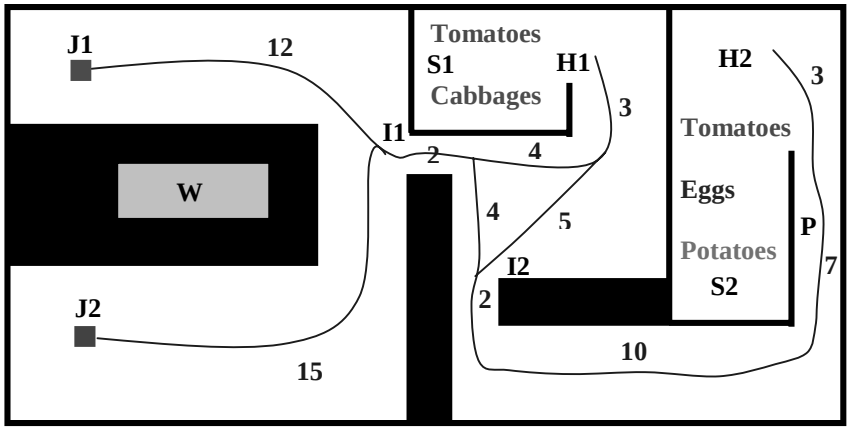

Figure 3: Illustration of warehouse setting

Now consider the case where **W** receives an order for cabbages and tomatoes. **W** will communicate this order to **w** who will announce the order to **j1** and **j2**. Agents **j1** and **j2** will estimate their costs to carry out the order and choose the route which allows them to fill the order with the lowest cost. Thus, **j1** and **j2** will both choose the route to **S1** and back which will cost each of them **42** and **48** respectively. Based on this estimate, plus the estimate payment to get the pallets loaded, **j1** is able to make a lower bid and hence will win the assignment from **w**. Once the assignment is won, **j1** will negotiate with **i1** for the right of passage through **I1** during the estimated required time intervals, and also with **h1** for the services of **H1** to pick and load the appropriate units of produce into the pallet. Assuming no conflicts, **i1** will charge the standard cost, **2**, for each pass through **I1**, and **h1** will charge the standard cost per unit of produce loaded into the pallet. Then **i1** and **h1** will reserve the relevant time slots in their schedules. Thus, the optimal solution will be produced.

However, this is a simplistic scenario. Consider the slightly more complex scenario where the order is for eggs and tomatoes. Once again, **j1** will win the assignment, but this time will choose to deal entirely with **S2**. But what happens if **H2** is currently occupied and hence causes a delay? This will translate to a greater cost for **J1** to complete the order. If the increase in cost is sufficiently high (the delay is sufficiently long), **j1** will choose the alternate route of first heading to **S1** for the tomatoes and then to **S2** for the eggs. Thus, the added cost of travelling to **S1** is offset by the lower delay at **S2** and hence the overall cost of completing the order is minimized. That is, optimality is achieved. Note that in a more sophisticated system, an idle agent assuming the role of Coordinator could perceive a large number of orders requiring stops at **S2** and produce a plan where **H1** moves to **S2** temporarily. If this plan results in greater profits for all involved parties, the coordinating agent will be able to easily sell the plan to the participating agents thus improving efficiency in the warehouse operations.

4.3 Proposed Implementation Scenario[2]

Domains in which complex tasks must be performed in environments that are largely inaccessible to humans motivate the proposed implementation. The tasks are thus best performed autonomously by robots. The claim is that for many of these domains (e.g., long-term planetary exploration, habitat construction, space facility construction and maintenance), the tasks can be performed better, faster, and cheaper using dynamically formed teams of heterogeneous robots. This is similar to the manner in which our society utilizes workforces consisting of specialists in diverse fields to perform similar tasks.

As an illustration, consider the problem of robotic exploration of Mars. For the foreseeable future, mobile robots will serve as the remote sensors and data collectors for scientists. To create an outpost for such long-term exploration, the robots need to assemble solar power generation stations, map sites and collect science data, and communicate with Earth on a regular basis. In this scenario on the order of ten robots are sent many with different capabilities. Some of the robots specialize in heavy moving and lifting, some in science data collection, some in drilling and coring, and some in communication. The rovers have different, but overlapping, capabilities – different sensors, different resolutions and fields of view, even different mobility, such as wheeled and aerial vehicles. Figure 4 is a NASA artist's depiction of a portion of such a scenario.

[2] Motivation for this scenario was obtained from the FIRE team's proposal to NASA for the "Heterogeneous Multi-Rover Coordination for Planetary Exploration" program.

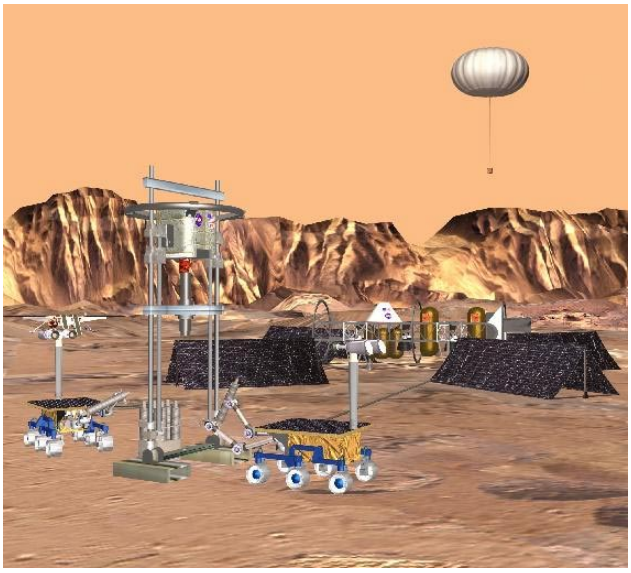

Figure 4: Conceptual Illustration of a Multirobot Martian Outpost (Illustration produced by courtesy of Jet Propulsion Laboratory)

The rovers cooperatively search for a location suitable in size and terrain for a base station. Once such a location is found, rovers with appropriate capabilities form several teams to construct the base station capable of housing supplies and generating energy. Two rovers carry parts, such as solar panels, that are too large for a single rover. Complementary capabilities are exploited – for example, to align and fasten trusses rovers with manipulators receive assistance from camera-bearing rovers that position themselves for advantageous viewing angles.

Meanwhile, other rovers begin general exploration of the region. To start, several scouting robots (perhaps joined by aerial vehicles) quickly survey the region. Scientists on Earth (and perhaps the rovers themselves) identify sites within the region that have high likelihood to contain interesting science data. Rovers with specialized sensing instruments are sent to investigate. If a particular subtask requires more intensive scrutiny, additional rovers with appropriate capabilities are brought in. For instance, to perform a seismographic test, one rover could transmit pulses into the ground while other rovers further away receive the reflected signals. Meanwhile, several rovers move to high ground to provide an inter-robot communications network. The exploration teams dynamically reconfigure depending on what is found and which rovers are available to help.

Rover failures are addressed by dispatching a rover with diagnostic capabilities. The diagnostic rover can use its cameras to view the failed robot to see if it can be aided in the field (e.g., if it has a stuck wheel or is high-centered), or it may drag the rover back to the base station to be repaired by replacement of failed modules. In the meantime, another robot with the same (or similar) capabilities can be substituted, so as to complete the original task with minimal interruptions.

At any given time, different teams of rovers may be involved in exploration, base-station construction/maintenance, and rover diagnosis/repair. Many tasks will be time critical, requiring execution within hard deadlines (e.g., repair of a failed power generation station) or synchronization with external events (communication satellite visibility, periods of sunlight). The teams form dynamically, depending on the task, environment, and capabilities and availability of the various robots to best meet mission requirements over time. The rovers negotiate their individual roles, ensure safety of the group and themselves, and coordinate their precise actions, attempting as a group to avoid unnecessary travel time, to minimize reconfiguration and wait time, and to prefer more reliable alternatives in cases of overlapping capabilities. The challenge is to

keep all the robots healthy and busy in appropriate tasks in order to maximize the scientific data collected.

Similar scenarios exist for domains such as habitat construction and in-space facility construction and maintenance. For instance, consider an inspection robot that has identified a failed component on the Space Station. It tries to assemble a team of robots to replace the failed component. After negotiation, a courier robot (capable of retrieving the necessary replacement part) and a repair robot (capable of swapping-out the failed device) take responsibility for the repair task, leaving the inspection robot free to continue inspection. While the courier collects the replacement part, the repair robot evaluates the problem and plans its course of action, possibly seeking additional aid if it encounters unexpected difficulties it is unable to resolve. Upon arrival with the replacement part, the courier and repair robot coordinate to complete the task.

While these scenarios are meant to motivate the types of capabilities the proposed work intends to develop, the focus will be on an experiment designed to demonstrate successful control of an autonomous robotic colony over a long duration (several hours).

The proposed implementation will be on the simulator and robotic system of the NASA funded project FIRE (Federation of Intelligent Robotic Explorers). The scenario will be that of a heterogeneous group of robots stationed at an extra-planetary outpost for scientific exploration of the region. The scenario will unfold in four stages:

1. The robots able to climb slopes well will disperse to the highest points within sight and scout out the area at high level
2. Based on the findings of the first stage, small teams of scouts will deploy to explore designated areas in greater detail
3. Based on the findings of the second stage, designated sites will be further characterized by the performance of science experiments
4. Based on the findings of the third stage, sample returns to the processing facility at the base station will be recommended

Furthermore, communication with the earth will only be available within a specific window of time each day. During this window, the robot with the most powerful antenna available will be required to locate to a high point in order to be able to successfully communicate with operators on earth. The federation of robots will only have a fixed data storage capacity, and hence have high incentive to offload the gathered data to earth each day. All robots will be able to communicate only within a limited range. Included in the federation will be the following robots:

Robots:	Resources:
R1, R2, R3	Radio, 2 cameras
R4, R5, R6	Radio, 1 camera, 1 manipulator, science instrument, fast computer
R7	Radio, tool-kit, 2 manipulators, 1 camera

These robots will be able to adopt the following roles if they have the necessary resources (Note that some of these resources will be simulated – for example diagnosing and fixing a robot will be implemented by a qualified robot spending a fixed time in close proximity to a disabled robot):

Candidate Roles:	Required Resources:
Communicator	Radio
Mapper	2 cameras + laser
Leader	Fast computer
Diagnoser/Fixer	1 camera, 2 manipulators, tool kit
Sampler/Scientist	1 camera, science instrument, 1 manipulator, fast computer

The initial task assigned to the federation is as follows:

Map a designated region (large-scale) and characterize any interesting sites as follows: If at any site "X1", “X2”, … or “Xn” is observed, perform the corresponding science experiment and if the experiment results in "Y1", “Y2”, … or “Yn”, analyze a sample and record the analysis.

The execution of the scenario will include the following activities:

- Exploration tasks are distributed among robots, and robots deploy, keeping in mind their maximum range for communication.
- Optimization by leaders can happen with respect to positioning communicators and distribution of exploration tasks to robots.
- New robots can enter the scene at any point during operations.
- Whenever an "X" is detected, a capable robot has to perform the required science experiment and collect a sample if the result is a "Y".
- Some robots can be disabled and its assignments will have to be re-distributed.
- The disabled robots will also have to enlist the aid of a diagnoser/fixer robot.
- Some disabilities will not be fixable, and creative solutions will emerge to complete the task. For example, one of the scientists could lose a camera and have to cooperate with a robot with a camera to do its experiments

The costs for this scenario will be based on fuel consumption and time. The commodity of value will be information (or scientific data).

The following features of the architecture will be highlighted through the scenario:

- Robust to robot death
- Opportunistic optimization
- Reasoning about limited resources
- Ability to dynamically accommodate new robots
- Ability to dynamically accommodate new tasks
- Ability to deal with limited communication

An important factor for this experiment will be identifying metrics for evaluation. A clearly applicable metric will be to compute the globally optimal solution when possible (where the optimality of the solution is measured by the total value of information gathered) and compare the results produced to the optimal result. However, the globally optimal solution will not always be computable. In such cases, a useful metric will be to generate and compare results with and without the opportunistic optimization component of the architecture.

4.4 Current Conceptions of Architectural Development and Optimization

Consider a team of agents assembled to perform a particular task. A software agent, representing the user, negotiates with the team to perform the task. The user desires the team to perform the task well while minimizing costs. To accomplish this, the user defines a revenue function that maps possible task outcomes onto revenue values, and a cost function that maps possible schemes for performing the task onto cost values. With the aim of maximizing their individual profits, the agents bid on parts of the task. They receive revenue for performing their subtasks, and they are charged for costs incurred while executing their subtasks. Once they have received their assignments, the agents can continue to negotiate amongst themselves, buying and selling subtasks to minimize costs. If the task changes during execution, perhaps because new information has been discovered, this creates new subtasks, which are bid out to the agents. If an agent malfunctions or dies during execution, this agent’s subtasks are bid out to the surviving agents. Agent resources, such as sensing, communication, and computing, can be bought and sold in the marketplace to concentrate resources where they are most productively employed.

The proposed structure for the architecture is shown below:

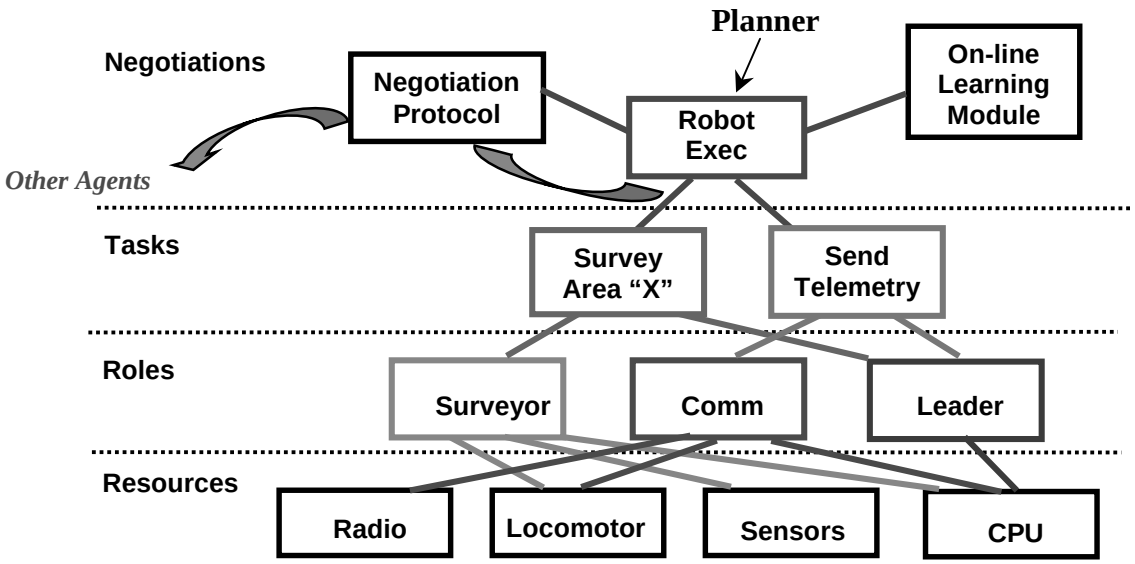

Figure 5: Current conception of architectural structure

Figure 5, which is tailored to the distributed mapping application, illustrates the architectural software that will run on each robot/agent in the team. It is organized in layers. In the bottom layer are the resources under the robot's control, such as sensors, computers, and communication devices. These resources are available to the robot to perform its tasks—some unused resources can be leased to other robots in the colony if there is a demand for them. For example, if a robot is not using its entire computing capacity, it can do another robot's data processing for a fee.

The next layer consists of the agent's roles for accomplishing tasks. Roles are application-specific software modules that implement particular agent capabilities or skills, such as acting as a communication router or generating optimal plans for a group of agents. The roles utilize resources in the layer below to execute their tasks. Roles execute tasks that match their specific capabilities. They receive assignments by bidding on tasks offered by other agents. As they execute their tasks, they may generate other tasks or subtasks to be bid out to the other agents. These new tasks will be communicated to the executive.

At the top layer in the architecture, the agent executive coordinates the activities of the agent and its interactions with other agents. All of the planning is carried out at this top layer. The executive bids on tasks for the agent to perform and offers tasks for sale. It matches tasks to roles, schedules the roles to run, and resolves any contention for resources using some form of combinatorial exchange scheme. Finally, it offers unused resources for sale to other agents. The executive is equipped with an on-line learning module that enables it to perform better over time by adapting to the specific application and environment. Some candidates for adaptation via on-line learning are negotiation strategies, role and resource management, cost and revenue models, navigation of the environment, and task decomposition. While the proposed work does not aim to produce novel learning methods, it does intend to provide the capability to apply existing learning techniques to improve architectural performance.

When a robot receives an announcement for a task, the task definition will indicate which role(s) are required to execute that task. Each role definition will include necessary resource requirements. Thus the robot executive will dynamically reason about task, role, and resource management in a manner which optimizes its profit. But the architecture for a single robot does not complete the picture. It is now instructive to examine how several robots will interact.

Figure 6 below illustrates the high-level interaction between a group of robots and two users (via user-interface agents or Operator Executives):

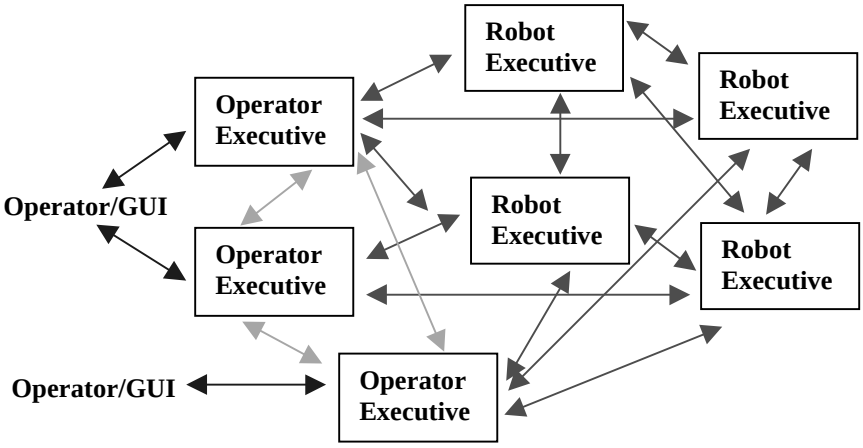

Figure 6: Interaction between operators and robots

As shown in Figure 6 above, the operators communicate high-level tasks to the interface agents known as the "Operator Executives" (or the OpExecs). The OpExecs then interpret the operator's commands and translate them into tasks that the robots can recognize. Next these tasks are bid out to the robots within communication range. The OpExecs are also able to negotiate amongst themselves to optimize usage of the robots and task allocation.

The architectural components interact with each other as illustrated in Figure 5. A crucial component of the architectural design will be investigating different designs for cost and revenue models that reflect the requirements of different application domains. They could potentially be complex functions involving statistical distributions and vectors of different components (for example, the cost function could be a combination of time spent, fuel expended, and CPU cycles utilized). These functions will need to reflect aspects such as priorities for task completion, hard deadlines for relevant tasks, and acceptable margins of error for different tasks. Other factors to be considered here are rewards for partially completed tasks, compensation for imperfect or incomplete state information, and errors in task execution. The negotiation protocol will require bids, calls for bids, and contracts to allow negotiation. Also, bidding mechanisms are necessary to compute bids and determine when to place a bid, and when to wait. Finally, penalty mechanisms are necessary to allow agents to break deals when profitable.

The OpExec will need to communicate with the operator(s), the robots, and any other OpExec(s). Through these communications, the OpExec will receive assignments that will be decomposed into tasks that will be bid out to the robots via the negotiation protocol. An OpExec may also subcontract assignments it receives to other OpExecs. Thus, each OpExec will maintain a portfolio of assignments, tasks, and subcontracts. A combinatorial exchange mechanism and some form of scheduler will be necessary to determine how best to match bids, tasks, and assignments. Finally, each OpExec will keep track of its wealth by updating its personal account after each relevant transaction.

The core components of the executive are illustrated below in Figure 7:

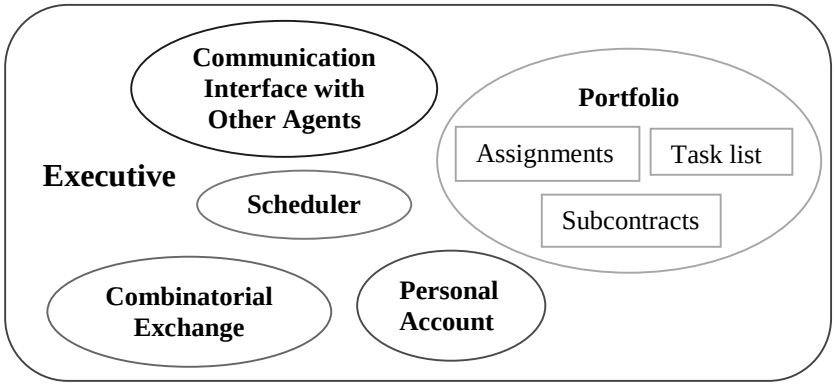

Figure 7: Core components of executive

The robot executives will need to communicate with the other robots and the OpExec(s). Through these communications, the robots will receive assignments (that may be further decomposed into sub-tasks) that will either be subcontracted out to the other robots via the negotiation protocol, or executed. Thus, each robot executive will maintain a portfolio of assignments, tasks to be executed, and subcontracts. A combinatorial exchange mechanism and some form of scheduler will be necessary to determine how best to match bids, tasks, and assignments. Finally, each robot executive will keep track of its wealth by updating its personal account after each relevant transaction. Thus, the operator-executives and the robot-executives will be identical except for the operator-executives' ability to translate commands from the operator into tasks that are recognized by the robots. Note that robots or operator-executives could then break down these tasks into sub-tasks. The method of task decomposition is currently an open research topic.

An important contribution of this work will be the development of a "leader" role which allows an agent with the necessary resources to assess the current plans of a group of robots and provide more optimal plans for the group. The leader can gain knowledge of the groups' current situation through communication or some form of observation. This allows introducing pockets of optimal centralized planning into the distributed system. Consider the following scenario: Robots R1, R2, and R3 have received composite offers of y1, y2, and y3 respectively for the tasks they are offering, while robots R4, R5, and R6 have been able to negotiate composite rewards of x1, x2, and x3 respectively for executing a collection tasks. A 7^{th} robot R7 has the ability to be a "leader" and thus has to negotiate to re-distribute all these tasks in some fashion so that it still maintains some personal profit. This involves deciding which tasks to buy from which robots such that R7 can pay a price that will increase the sellers profit margin, and then off-load the tasks to other robots at a price which will allow R7 to improve its personal profit margin. Furthermore, there may be other leaders in competition with R7 to optimize resource utilization among other subsets of robots that may overlap in varying degrees with R7's group. Hence, this research problem is different from existing work in combinatorial exchange schemes because there will be competition amongst the exchanges.

A primary candidate for suitable optimization techniques for solving this problem is the combinatorial exchange algorithm developed by Sandholm et al [59]. This algorithm and other optimization techniques will be investigated to ascertain what works best for the type of optimization problems encountered in the multirobot coordination domain. Other potential methods for solving this problem are depth first search, breadth first search, genetic/evolutionary algorithms, and simulated annealing. Brief descriptions of these approaches are presented below.

- **Depth First Search**:
  In depth first search the node's descendants are searched before its unvisited siblings. For tree search at least, depth first search has low memory requirements, since only the nodes on the

'current' path need to be recorded. One criticism is that depth first search may get stuck exploring long (and potentially infinite) blind alleys, when there is in fact a solution path of only one or two steps. However, this can be prevented by setting a depth limit to the search (but then it wouldn't be exhaustive.). Thus, for depth first search, infinite search paths are generally not a problem, since trying all possibilities imposes a maximum search depth. The problem is finding a practical search depth cutoff. An artificial cut off results in a loss in optimality and completeness. Depth first searches are great for their "any time" property – a feasible solution is produced in a short time.

- **Breadth First Search**
  In breadth first search the siblings are searched before its descendants. It is very systematic and is guaranteed to find the solution if it exists. It explores the whole tree and will therefore locate all possible solutions (as well as all the non-solutions). The algorithm is, however, costly; as it calculates all possible solutions it will take a long time and need a lot of memory to do so. Breadth first search may use more memory, but will not get stuck in blind alleys, and will always find the shortest path first (or at least, the path that involves the least number of steps). It may be more appropriate when exploring very large search spaces where there is an expected solution which takes a relatively small number of steps, or when you are interested in all the solutions (perhaps up to some depth limit). In general, breadth first search algorithms are good for searching graphs and other structures where the search will take you to the same states repeatedly, since such search algorithms save and re-use partial results. This makes them good for re-planning. But it is also why they require more memory. Moreover, since they explore all possible branches at each state before continuing, they are slower to come up with a feasible solution and are not "any time".

- **Simulated Annealing**:
  In order to avoid the metastable states produced by quenching, metals are often cooled very slowly, which allows them time to order themselves into stable, structurally strong, low energy configurations. This is called annealing. Annealing gives the system the opportunity to jump out of local minima with a reasonable probability while the temperature is still relatively high. This iterative improvement algorithm is effective in overcoming local maxima. With a slowly reduced temperature, finding the global maximum is guaranteed. Time complexity however can rapidly grow in certain cases requiring low temperature.
  The advantages to this method are that it can deal with arbitrary systems and cost functions, it statistically guarantees finding an optimal solution, it is relatively easy to code, and it generally produces a "good" solution even for complex problems. This makes simulated annealing an attractive option for optimization problems where heuristic (specialized or problem specific) methods are not available.
  Disadvantages of simulated annealing are as follows. This method can be very slow, especially if the cost function is expensive to compute. For problems where the energy landscape is smooth, or there are few local minima, it is overkill – simpler, faster methods (e.g., gradient descent) will work better. But usually the energy landscape is not known. Heuristic methods, which are problem-specific or take advantage of extra information about the system, will often be better than general methods. But simulated annealing is often comparable to heuristics. The method cannot tell whether it has found an optimal solution. Some other method (e.g. branch and bound) is required to do this.

- **Genetic Algorithms**:
  Genetic algorithms (GAs) are optimization techniques based on the concepts of natural selection and genetics. In this approach, the variables are represented as genes on a chromosome. GAs feature a group of candidate solutions (population) on the response surface. Through natural selection and the genetic operators, mutation and recombination, chromosomes with better fitness are found. Natural selection guarantees that chromosomes with the best fitness will propagate in future populations. Using the recombination operator, the GA combines genes from two parent chromosomes to form two new chromosomes (children) that have a high probability of having better fitness than their parents have. Mutation allows new areas of the response surface to be explored. GAs offer a generational

improvement in the fitness of the chromosomes and after many generations will create chromosomes containing the optimized variable settings.
There are many advantages to using genetic algorithms. They require no knowledge or gradient information about the response surface. Discontinuities present on the response surface have little effect on overall optimization performance. They are resistant to becoming trapped in local optima. They perform very well for large-scale optimization problems. They can be employed for a wide variety of optimization problems. GA can search spaces of hypotheses containing complex interacting parts, are easily parallelized and can take advantage of additional computational power. There are also disadvantages to using genetic algorithms. They generally have trouble finding the exact global optimum. They require a large number of response (fitness) function evaluations and their configuration is not straightforward.

Another aspect to be considered in the proposed work is the bidding language. Optimal communication of the requirements of different agents will require an efficiently designed bidding language. For example an agent may require an end result which can be achieved by several combinations of items (like the choice between several paths to reach a goal in a TSP). One method to communicate the agent's requirements is to communicate an exponentially long list of bids (concatenated as OR statements). This is however not desirable. An alternative is to communicate the agent's requirements via a suitable data structure (for example a graph with prices attached to each node and connection). Hence, it may be advantageous to develop a suite of data structures supported by the bidding language that will be useful for efficient communication across many multirobot application domains. The goal will be to design a bidding language that takes advantage of any inherent structure common to multirobot optimization problems.

As a final result, this work aims to produce a "plug-and-play" architecture that can be applied to a wide variety of multirobot applications. While the basic structure of the architecture will be available, users will have to program in relevant resources, roles, cost and revenue models, bidding strategies, and on-line learning schemes. Protocols and examples will be provided for all of these components. Also, advice on how to design cost and revenue functions for different applications will be available. Finally, suites of certain components such as bidding strategies, learning schemes, scheduling algorithms, common resources, useful problem representations, and common roles will be made available.

5. Work To Date

5.1 Step I:
Initial Implementation of Market-Based Architecture in Simulation

An initial version of the market-based architecture was developed and tested for a distributed sensing task in a simulated interior environment[3]. A group of robots, located at different starting positions in a known simulated world, were assigned the task of visiting a set of pre-selected observation points. This problem is equivalent to the distributed traveling salesman problem, where the observation points are the cities to visit. The robot colony was structured as illustrated below (the illustration shows a snapshot of the robot colony at a given time and the double-headed arrows indicate communication channels):

[3] This work was published in IAS-6 (Dias and Stentz 24).

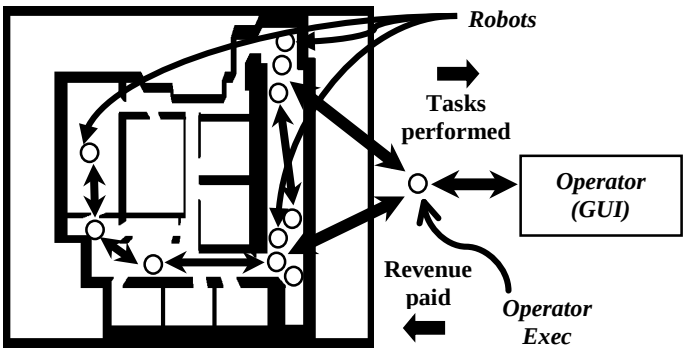

Figure 8: Organizational structure for a colony of robots engaged in distributed mapping

Each robot was equipped with a map of the world, which enabled it to calculate the cost associated with visiting each of these cities. The costs were the lengths of the shortest paths between cities in an eight-connected grid, interpreted as money. Let $\mathbf{c_{ij}}$ be the cost for the $\mathbf{j}^{th}$ robot to visit the $\mathbf{i}^{th}$ city from the ($\mathbf{i\text{-}1}^{th}$) city in its tour (where the $\mathbf{0}^{th}$ city is the starting location).

The robot cost function for the $\mathbf{j}^{th}$ robot was computed as follows:

$$\mathbf{rcost(j)} = \sum_{\mathbf{i}=1}^{\mathbf{n_j}} \mathbf{c_{ij}}$$

where $\mathbf{n_j}$ is the number of cities in the tour for robot **j**. The team cost function was:

$$\mathbf{tcost} = \sum_{\mathbf{j}=1}^{\mathbf{m}} \mathbf{rcost(j)}$$

where **m** is the number of robots. The team revenue and robot revenue functions, **trev** and **rrev**, were determined by the negotiated prices. The maximum available team revenue was chosen to exceed team costs for reasonable solutions to the problem.

All robots (bidders) adopted the same simplistic strategy of bidding a fixed percentage of the maximum profit they could obtain. According to this strategy, if a task were on offer for a maximum price of **r**, and the cost to carry out the task were **c**, a robot computed its bid **b** as follows:

$$\mathbf{b = 0.9*(r-c) + c}$$

Thus, the robots bid for each city based on their estimated costs to visit that city.

The interface between the human operator and the team of robots was a software agent, the *operator executive* (*exec*). The *exec* conveyed the operator's commands to the members of the team, managed the team revenue, monitored the team cost, and carried out the initial city assignments. Being a self-interested agent, the *exec* aimed to assign cities quickly while minimizing revenue flow to the team. In our initial implementation, the *exec* adopted the following greedy algorithm for assigning tasks:

- Announce all cities to all robots and wait for all incoming bids
- Insert each incoming bid in a priority queue with the lowest bid claiming the highest priority
- Assign **m** cities (one to each robot) starting with the highest priority bid. (Note, once a city is assigned from the priority queue, all other bids for that city and all other bids submitted by that robot are removed from the queue before making the next assignment)

- Delete all bids, and call for re-bids for all remaining cities
- Repeat procedure until all **n** cities are assigned

Once the *exec* had completed the initial city assignments, the robots negotiated amongst themselves to subcontract city assignments. Unlike a bartering system where robots can only make "city-for-city" deals, our architecture allows robots to make "city-for-revenue" deals, thereby enabling transactions that reduce team costs but increase a robot's individual costs. Each of the robots, in turn (the initial implementation was fully synchronous), offered all the cities on its tour (individually) to all the other robots for a maximum price equal to the offerer's cost reduction by removing that city from its tour. Each bidder then submitted a bid for that city greater than the cost for adding the city to its tour. Note that a bidder only submitted a bid for a city if it could make at least a fixed minimum profit by inserting that city into its tour. In order to estimate the additional cost of inserting a city into its tour, the bidder evaluated the cost of inserting that city at each point of the tour and picked the point of insertion that resulted in the lowest cost increase. In this initial implementation, only single-city deals were considered, and the robots continued to negotiate amongst themselves until no new, mutually profitable deals were possible. Thus, negotiations ceased once the system settled into a local minimum of the global cost.

Some preliminary results are illustrated in the figures below:

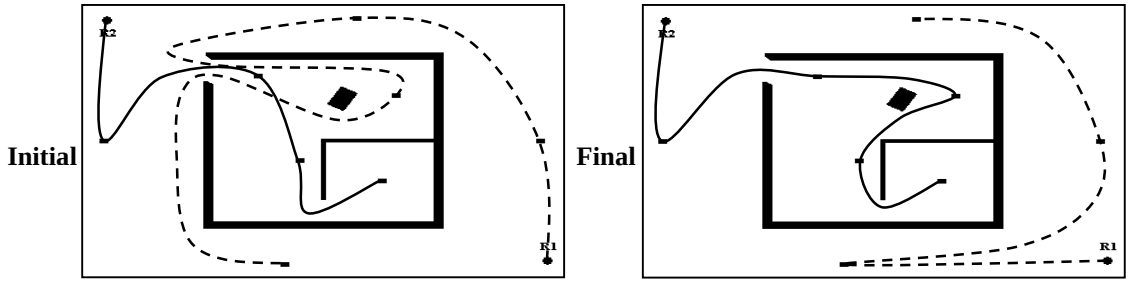

Figure 9: Initial assignments and final tours for 2 robots and 8 cities (14.7% decrease in team cost)

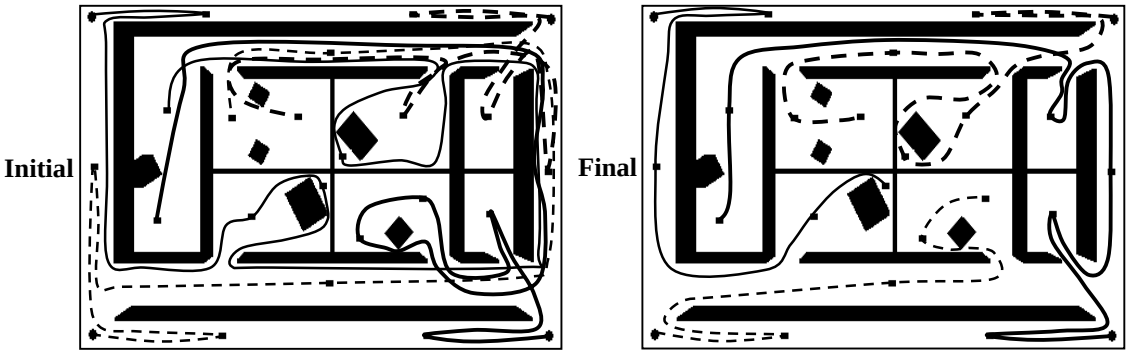

Figure 10: Initial assignments and final tours for 4 robots and 20 cities

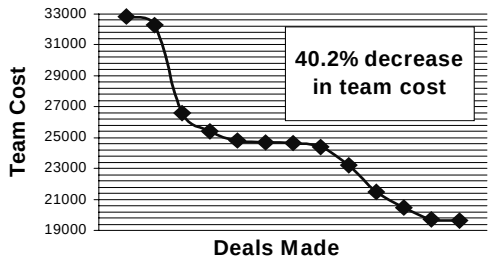

Figure 11: Team cost reduction during inter-robot negotiation for the example in Figure 10

Note that the reported decrease in team costs was calculated on the operator executive's initial greedy assignment and not on a random assignment that would have resulted in a significantly higher initial team cost on average. Further comparisons to globally optimal solutions and other greedy solutions will be carried out in the near future. Although many features of the market-based architecture were not implemented in this preliminary version, initial experiments clearly show effective global plans with low team costs.

5.2 Step II: Extension of Architectural Capability to Respond to Dynamic Conditions in Simulation

Preliminary testing was also carried out to evaluate the system response to dynamic conditions in a scenario similar to exploration of a partially known world[4]. The operator designated a set of observation points to be visited in a simulated world. Robots began their exploration based on the tours they won after negotiations had ceased. Whenever a robot approached a doorway, a new observation point was dynamically triggered inside the "newly observed" room. When a new goal was triggered, the robots ceased their current tours and began negotiations to determine new tours that were most profitable in light of the new information. Thus, they were able to adapt to the dynamically triggered conditions.

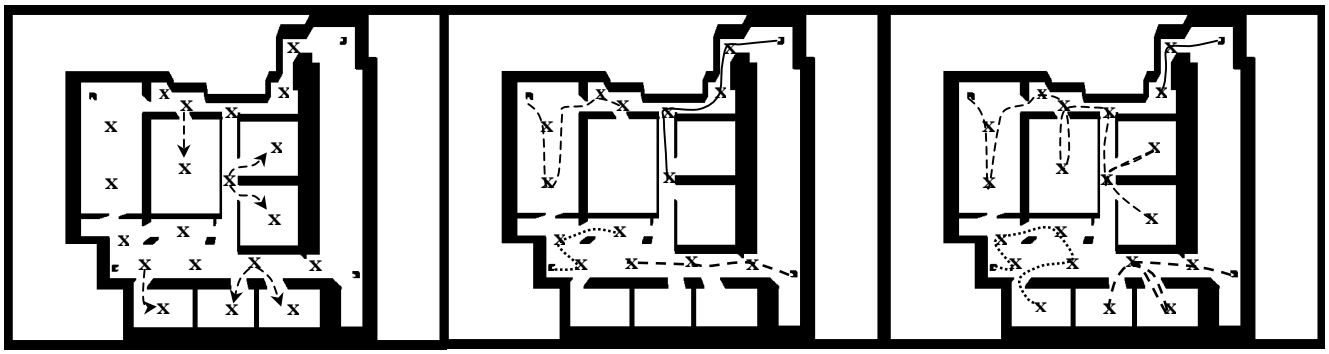

Figure 12: Results from Dynamic re-negotiations

The illustration above shows the evolution of four robots' tours in a 20-city exploration scenario.[5] Initially the operator designated 14 cities. Whenever a robot reached a doorway, a city was triggered in the "newly discovered" room. Thus, six additional cities were triggered and the robots adapted their tours to allow visiting all the cities.

[4] This work was published in SPIE 2000 (Thayer et al. [80])
[5] A movie of this evolution is available at http://www.frc.ri.cmu.edu/projects/colony/frcworld.shtml

5.3 Step III: Initial Experimentation with the Market-Based Architecture applied to a Multirobot System

An initial version of the architecture was implemented on the Cognitive Colonies robotic system. The aim of the Colonies project is to build a group of robotic aids for urban reconnaissance. Ten PioneerII-DX robots, built by Activmedia Incorporated, (shown in Figure 13) were purchased for the project.

Figure 13: Robot team used in Cognitive Colonies project

The robots are each equipped with onboard computing as well as 16 sonar sensors for obstacle detection and a forward pointing camera for map construction. The distributed TSP scenario was tested using 4 of these robots in a cluttered environment. The robots only negotiated with an OpExec in this implementation. Inter-robot negotiation was not implemented due to some communication problems that are currently being sorted out. The operator was able to specify goals to be visited via a GUI. These goals were then bid out to the robots by an OpExec and assigned according to the greedy algorithm described in the previous section. The robots started at different positions in the FRC high-bay and were able to complete the assigned tours. Robots were also able to dynamically handle new tasks assigned during execution. Many valuable lessons for architectural design, especially regarding communication and real-time system issues, were learned through the process of this implementation.

6. Conclusion

This report describes a market-based architecture for coordinating a group of robots to achieve a given objective. The architecture is inherently distributed, but also opportunistically forms centralized sub-groups to improve efficiency, and thus approach optimality. Robots are self-interested agents, with the primary goal of maximizing individual profits. The revenue/cost models and rules of engagement are designed so that maximizing individual profit has the benevolent effect of moving the team toward the globally optimal solution. This architecture inherits the flexibility of market-based approaches in allowing cooperation and competition to emerge opportunistically. Therefore, this approach is well suited to address the multirobot control problem for autonomous robotic colonies carrying out complex tasks in dynamic environments where it is highly desirable to optimize to whatever extent possible. Future work will develop the core components of a market-based multirobot control-architecture, investigate the use of a negotiation protocol for task distribution, design and implement resource and role management schemes, and apply optimization techniques to improve system performance.

7. Acknowledgements

This research was sponsored in part by NASA, under contract "Heterogeneous Multi-Rover Coordination for Planetary Exploration" (contract number NCC2-1243). This research was also sponsored in part by DARPA, under contract “Cognitive Colonies” (contract number N66001-99-1-8921, monitored by SPAWAR). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of the U.S. Government. The authors thank the members of the Cognitive Colonies group: Scott Thayer, Bruce Digney, Martial Hebert, Robert Zlot, Aaron Morris, Charles Smart, and Bart Nabbe and the FIRE group: Reid Simmons, Stephen Smith, Marion F. Smith, III, and Vincent Cicirello for their valuable contribution.

8. References

1. Alur, R., Das, A., Esposito, J., Fierro, R., Grudic, G., Hur, Y., Kumar, V., Lee, I., Ostrowski, J., Pappas, G., Southall, B., Spletzer, J., and Taylor, C. J., "A Framework and Architecture for Multirobot Coordination", Seventh *International Symposium on Experimental Robotics*, 2000.
2. Andersson, M. R., and Sandholm, T. W., "Sequencing of Contract Types for Anytime Task Reallocation", *Agent-Mediated Electronic Trading, Lecture Notes in Artificial Intelligence*, Volume 1571, pp.54-69, 1999.
3. Arkin, R. C., "Cooperation without Communication: Multiagent Schema-Based Robot Navigation", *Journal of Robotic Systems*, Vol. 9, No.3, pp. 351-364, 1992.
4. Arkin, R. C., Balch, T., "AuRA: Principles and Practice in Review", *Journal of Experimental & Theoretical Artificial Intelligence*, Vol. 9, No. 2/3, pp.175-188, 1997.
5. Balch, T., *Learning Roles: Behavioral Diversity In Robot Teams*, College of Computing Technical Report GIT-CC-97-12, Georgia Institute of Technology, and AAAI Workshop on Multiagent Learning, 1997.
6. Böhringer, K., Brown, R., Donald, B., Jennings, J., and Rus, D., "Distributed Robotic Manipulation: Experiments in Minimalism*", Proceedings of the International Symposium on Experimental Robotics (ISER)*, 1995.
7. Bonasso, R. P., Firby, R. J., Gat, E., Kortenkamp, D., Miller, D. P., and "Slack, M. G., "Experiences with an Architecture for Intelligent, Reactive Agents*", Journal of Experimental & Theoretical Artificial Intelligence*, Volume 9(2/3), pp.237-256, 1997.
8. Brandt, F., Brauer, W., and Weiß, G., "Task Assignment in Multiagent Systems based on Vickrey-type Auctioning and Leveled Commitment Contracting", *Cooperative Information Agents IV, Lecture Notes in Artificial Intelligence*, Volume 1860, pp.95-106, 2000.
9. Brauer, W., and Weiß, G., "Multi-Machine Scheduling – A Multi-Agent Learning Approach", *Proceedings of the 3rd International Conference on Multi-Agent Systems*, pp. 42-48, 1998.
10. Brooks, R. A., "Elephants Don't Play Chess", *Robotics and Autonomous Systems*, Vol. 6, pp.3-15, 1990.
11. Brumitt, B. L., Stentz, A., "Dynamic Mission Planning for Multiple Mobile Robots", *Proceedings of the IEEE International Conference on Robotics and Automation*, No. 3, pp. 2396-2401, 1996.
12. Burgard, W., Moors, M., Fox, D., Simmons, R., and Thrun, S., "Collaborative Multi-Robot Exploration", *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco CA, April 2000.
13. Callaghan, V., Clarke, G., Pounds-Cornish, A., Sharples, S., "Buildings as Intelligent Autonomous Systems: A Model for Integrating Personal and Building Agents", *The 6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pp.410-415, 2000.
14. Caloud, P., Choi, W., Latombe, J-C., Le Pape, C., and Yim, M., "Indoor Automation with Many Mobile Robots", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.67- 72, 1990.
15. Castelfranchi, C., and Conte, R., "Limits of economic and strategic rationality for agents and MA systems", *Robotics and Autonomous Systems*, Volume 24, pp.127-139, 1998.
16. Chaimowicz, L., Sugar, T., Kumar, V., and Campos, M. F. M., "An Architecture for Tightly Coupled Multi-Robot Cooperation", *IEEE International Conference on Robotics and Automation*, 2001.
17. Cheng, Y-Q., Wu, V., Collins, R. T., Hanson, A. R., and Riseman, E. M., "Maximum-weight bipartite matching technique and its applications in image feature matching", *SPIE Conference on Visual Communication and Image Processing*, 1996.
18. Chevallier, D., and Payandeh, S., "On Kinematic Geometry of Multi-Agent Manipulating System Based on the Contact Force Information", *The 6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pp.188-195, 2000.
19. Cicirello, V., and Smith, S., "Insect Societies and Manufacturing", *The IJCAI-01 Workshop on Artificial Intelligence and Manufacturing: New AI Paradigms for Manufacturing*, 2001.
20. Cicirello, V., and Smith, S., "Wasp Nests for Self-Configurable Factories", *Agents '01, Proceedings of the Fifth International Conference on Autonomous Agents*, 2001.
21. Collins, J., Jamison, S., Mobasher, B., and Gini, M., "A Market Architecture for Multi-Agent Contracting", *Technical Report 97-15*, University of Minnesota, 1997.

22. Decker, K. S., and Lesser, V. R., *Designing A Family Of Coordination Algorithms, Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pp. 73-80, 1995.
23. Desai, J. P., Ostrowski, J., and Kumar, V., "Controlling formations of multiple mobile robots", *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2864-2869, 1998.
24. Dias, M. B., Stentz, A., "A Free Market Architecture for Distributed Control of a Multirobot System", *The 6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pp.115-122, 2000.
25. Excelente-Toledo, C. B., Bourne, R. A., and Jennings, N. R., "Reasoning about Commitments and Penalties for Coordination between Autonomous Agents", *Proceedings of the 5th International Conference on Autonomous Agents*, 2001.
26. Faratin, P., Sierra, C., Jennings, N. R., "Negotiation decision functions for autonomous agents", *International Journal of Robotics and Autonomous Systems*, Volume 24(3-4), pp. 159-182, 1997.
27. Gerkey, B. P. and Matarić, M. J., "Sold! Market methods for multi-robot control", *IEEE Transactions on Robotics and Automation Special Issue on Multi-Robot Systems*, submitted March 2001.
28. Gibney, M. A., Jennings, N. R., Vriend, N. J., and Griffiths, J. M., "Market-Based Call Routing in Telecommunications Networks Using Adaptive Pricing and Real Bidding", *3rd Inernational Workshop on Multi-Agent Systems and Telecommunications (IATA-99)*, pp.50-65, 1999.
29. Goldman, C. V., and Rosenschein, J. S., "Emergent Coordination through the Use of Cooperative State-Changing Rules", Proceedings of the Twelfth National Conference on Artificial Intelligence, pp. 408-413, 1994.
30. Golfarelli, M., Maio, D., Rizzi, S., "A Task-Swap Negotiation Protocol Based on the Contract Net Paradigm", Technical Report CSITE, No. 005-97, 1997.
31. Haynes, T., Lau, K., and Sen, S., "Learning Cases To Compliment Rules For Conflict Resolution In Multiagent Systems", *AAAI Spring Symposium on Adaptation, Coevolution, and Learning in Multiagent Systems*, 1996.
32. Huber, M. J., and Durfee, E. H., "Deciding When To Commit To Action During Observation-Based Coordination", *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pp. 163-170, 1995.
33. Jennings, B., and Arvidsson, Å., "Co-operating Market/Ant Based Multi-agent Systems for Intelligent Network Load Control", *Proceedings of the 3rd International Workshop on Intelligent Agents for Telecommunication Applications (IATA), LNAI*, Vol. 1699, pp. 62-75, 1999.
34. Jennings, J., Kirkwood-Watts, C., "Distributed Mobile Robotics by the Method of Dynamic Teams", *4th International Symposium on Distributed Autonomous Robotic Systems*, 1998.
35. Jensen, R. M., Veloso, M. M., "OBDD-based Universal Planning: Specifying and Solving Planning Problems for Synchronized Agents in Non-Deterministic Domains", *Lecture Notes in Computer Science*, No. 1600, pp. 213-248, 1999.
36. Jung, H., Tambe, M., and Kulkarni, S., "Argumentation as Distributed Constraint Satisfaction: Applications and Results", *Proceedings of the 5th International Conference on Autonomous Agents*, 2001.
37. Kaminka, G. A., and Tambe, M., "Robust Agent Teams via Socially-Attentive Monitoring", *Journal of Artificial Intelligence Research*, Volume 12, pp. 105-147, 2000.
38. Khuller, S., "On-line algorithms for weighted bipartite matching and stable marriages", *Theoretical Computer Science*, Volume 127(2), pp.255-267. 1994.
39. Krovi, R., Graesser, A. C., and Pracht, W. E., "Agent Behaviors in Virtual Negotiation Environments", *IEEE Transactions on Systems, Man, and Cybernetics*, Volume 29, pp.15-25, 1999.
40. Laengle, T., Lueth, T. C., Rembold, U., and Woern, H., "A distributed control architecture for autonomous mobile robots – implementation of the Karlsruhe Multi-Agent Robot Architecture (KAMARA)", *Advanced Robotics*, Volume 12, No. 4, pp. 411-431, 1998.
41. Lux, T., Marchesi, M., "Scaling and Criticality in a Stochastic Multi-Agent Model of a Financial Market", *Nature*, Vol. 397, No. 6719, pp. 498-500, 1999.
42. Maio, D., and Rizzi, S., "Unsupervised Multi-Agent Exploration of Structured Environments", *Proceedings of First International Conference on Multi-Agent Systems*, pp.269-275, 1995.
43. Matarić, M. J., "Interaction And Intelligent Behavior", Ph.D. Thesis, MIT, 1994.
44. Matarić, M. J., "Issues and Approaches in the Design of Collective Autonomous Agents", *Robotics and Autonomous Systems*, Vol. 16, pp. 321-331, 1995.
45. Matarić, M. J., "Learning To Behave Socially", *Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, pp. 453-462, 1994.

46. Matarić, M., "Coordination and Learning in Multi-Robot Systems", *IEEE Intelligent Systems*, pp. 6-8, 1998.
47. Matos, N., and Sierra, C., "Evolutionary Computing and Negotiating Agents", *Agent-Mediated Electronic Trading, Lecture Notes in Computer Science*, Volume 1571, pp. 126-150, 1999.
48. Noreils, F. R., "Toward a Robot Architecture Integrating Cooperation between Mobile Robots: Application to Indoor Environment", *The International Journal of Robotics Research*, Volume 12(1), pp.79-98, 1993.
49. Osawa, E., "A Metalevel Coordination Strategy For Reactive Cooperative Planning", *Proceedings of First International Conference on Multi-Agent Systems (ICMAS-95)*, pp. 297-303, 1995.
50. Pagello, E., D'Angelo, A., Montsello, F., Garelli, F., Ferrari, C., "Cooperative Behaviors in Multi-Robot Systems through Implicit Communication", *Robotics and Autonomous Systems*, Vol. 29, No. 1, pp. 65-77, 1999.
51. Panzarasa, P., and Jennings, N. R., "Social Influence and The Generation of Joint Mental Attitudes in Multi-Agent Systems", *Proceedings of the Eurosim Workshop on Simulating Organisational Processes*, 2001.
52. Parker, L. E., "ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation", *IEEE Transactions on Robotics and Automation*, Vol. 14, No.2, pp. 220-240, 1998.
53. Parker, L. E., "Designing Control Laws for Cooperative Agent Teams", *Proceedings of IEEE International Conference on Robotics and Automation*, pp.582-587, 1994.
54. Prasad, M. V. N., Lesser, V. R., and Lander, S. E., "Learning Organizational Roles In A Heterogeneous Multi-Agent System", *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pp. 72-77, 1996.
55. Rosin, C. D., and Belew, R. K., "Methods for Competitive Co-Evolution: Finding Opponents Worth Beating", *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp. 373-380, 1995.
56. Sandholm, T., "An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations", *Proceedings, Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pp. 256-262, 1993.
57. Sandholm, T., and Lesser, V., "Advantages Of A Leveled Commitment Contracting Protocol", *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pp. 126-133, 1996.
58. Sandholm, T., and Lesser, V., "Coalition Formation Among Bounded Rational Agents", *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pp. 662-669, 1995.
59. Sandholm, T., and Suri, S., "Improved Algorithms for Optimal Winner Determination in Combinatorial Auctions and Generalizations", *National Conference on Artificial Intelligence (AAAI)*, pp. 90-97, 2000.
60. Sandholm, T., Larson, K., Andersson, M., Shehory, O., and Tohmé, F., "Coalition structure generation with worst case guarantees", *Artificial Intelligence*, Volume 111(1-2), pp.209-238, 1999.
61. Sandholm, T., Lesser, V., "Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework", *Proceedings, First International Conference on Multiagent Systems (ICMAS-95)*, pp. 328-335, 1995.
62. Sandholm, T., Sikka, S., and Norden, S., "Algorithms for Optimizing Leveled Commitment Contracts", *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 535-540, 1999.
63. Schaerf, A., Shoham, Y., and Tennenholtz, M., "Adaptive Load Balancing: A Study In Multiagent Learning", *Journal of Artificial Intelligence Research*, Volume 2, pp. 475-500, 1995.
64. Schmidhuber, J., "A General Method For Multiagent Reinforcement Learning In Unrestricted Environments", *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pp. 84-87, 1996.
65. Schneider, J., Wong, W-K, Moore, A., and Riedmiller, M., "Distributed Value Functions", *Proceedings of the Sixteenth International Conference on Machine Learning*, pp.371-378, 1999.
66. Schneider-Fontán, M., Matarić, M. J., "A Study of Territoriality: The Role of Critical Mass in Adaptive Task Division", *Proceedings, From Animals to Animats 4, Fourth International Conference on Simulation of Adaptive Behavior (SAB-96)*, MIT Press/Bradford Books, pp. 553-561, 1996.
67. Schneider-Fontán, M., Matarić, M. J., "Territorial Multi-Robot Task Division", *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 5, 1998.

68. Shehory, O., and Kraus, S., "Task Allocation Via Coalition Formation Among Autonomous Agents", *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pp. 655-661, 1995.
69. Simmons, R., Apfelbaum, D., Fox, D., Goldman, R. P., Haigh, K. Z., Musliner, D. J., Pelican, M., and Thrun, S., "Coordinated Deployment of Multiple, Heterogeneous Robots", *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, Takamatsu Japan, 2000.
70. Simmons, R., Singh, S., Hershberger, D., Ramos, J., and Smith, T., "First Results in the Coordination of Heterogeneous Robots for Large-Scale Assembly", *Proceedings of the International Symposium on Experimental Robotics (ISER)*, 2000.
71. Smith, R., "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver", *IEEE Transactions on Computers*, Vol. C-29, No. 12, 1980.
72. Stentz, A., and Dias, M. B., "A Free Market Architecture for Coordinating Multiple Robots", Technical report, CMU-RI-TR-99-42, Robotics Institute, Carnegie Mellon University, 1999.
73. Stone, P., and Veloso, M., "Multiagent Systems: A Survey from a Machine Learning Perspective", Carnegie Mellon University Computer Science Technical Report, CMU-CS-97-193, 1997.
74. Švestka, P., Overmars, M. H., "Coordinated Path Planning for Multiple Robots", *Robotics and Autonomous Systems*, Vol. 23, No. 4, pp. 125-133, 1998.
75. Sycara, K., and Zeng, D., "Coordination of Multiple Intelligent Software Agents", *International journal of Intelligent and Cooperative Information Systems*, Volume 5(2-3), pp.181-211, 1996.
76. Tambe, M., "Towards Flexible Teamwork", *Journal of Artificial Intelligence Research*, Vol. 7, pp. 83-124, 1997.
77. Tambe, M., "Tracking Dynamic Team Activity", *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.
78. Tambe, M., Pynadath, D. V., Chauvat, N., Das, A., and Kaminka, G. A., "Adaptive Agent Integration Architectures for Heterogeneous Team Members", *Proceedings of the International Conference on MultiAgent Systems*, pp.301-308, 2000.
79. Tan, M., "Multi-Agent Reinforcement Learning: Independent Vs. Cooperative Agents", *Proceedings of the Tenth International Conference on Machine Learning*, pp. 330-337, 1993.
80. Thayer, S. M., Dias, M. B., Digney, B. L., Stentz, A., Nabbe, B., and Hebert, M., "Distributed robotic mapping of extreme environments" *Proceedings of SPIE* Vol. 4195: Mobile Robots XV and Telemanipulator and Telepresence Technologies VII, November 2000.
81. Veloso, M., Stone, P., Bowling, M., "Anticipation: A Key for Collaboration in a Team of Agents", Submitted to the 3rd International Conference on Autonomous Agents, pp. 1-16, 1998.
82. Wang, ZD., Piñeros, C. V., Takahashi, T., Kimura, Y., and Nakano, E., "Arrangement and Control of a Cooperative Multi-Robot System for Object Transportation", *The 6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pp.196-203, 2000.
83. Weiß, G., "A Multiagent Perspective of Parallel and Distributed Machine Learning", *Proceedings of the 2nd International Conference on Autonomous Agents*, pp. 226-230, 1998.
84. Weiß, G., "Achieving Coordination through Combining Joint Planning and Joint Learning", *Technical Report FKI-232-99*, Institut für Informatik, TU München, 1999.
85. Wellman, M. P., and Wurman, P. R., "Market-Aware Agents for a Multiagent World", *Robotics and Autonomous Systems*, Volume 24, pp.115-125, 1998.
86. Zeng, D., and Sycara, K., "Bayesian Learning In Negotiation", *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pp. 99-104, 1996.
87. Zeng, D., and Sycara, K., "Benefits of Learning in Negotiation", *Proceedings of the AAAI National Conference on Artificial Intelligence*, pp.36-41, 1997.
88. Zhang, J., Ferch, M., and Knoll, A., "Carrying Heavy Objects by Multiple Manipulators with Self-Adapting Force Control", *The 6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pp.204-211, 2000.
89. Zlotkin, G., and Rosenschein, J. S., "Coalition, Cryptography, And Stability: Mechanisms For Coalition Formation In Task Oriented Domains", *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp. 432-437, 1994.