Increasing Realism in Coalition Formation for Multi-Agent Systems

A thesis presented

by

Michael van de Vijsel

to

The Department of Computer Science

in partial fulfillment of the requirements

for the degree of

Master of Science

in the subject of

Computer Science

The University of Manitoba

Winnipeg, Manitoba

August 2005

0-494-08984-9

Your file Votre référence
ISBN:
Our file Notre reterence
ISBN:

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

Increasing Realism in Coalition Formation for Multi-Agent Systems

BY

Michael van de Vijsel

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of

Manitoba in partial fulfillment of the requirement of the degree

Of

Master of Science

Thesis advisor

John Anderson

Author

Michael van de Vijsel

Increasing Realism in Coalition Formation for Multi-Agent Systems

Abstract

It is well known that teams of agents in a multi-agent system often perform better than individual agents working alone. Most research in multi-agent systems has made the assumption that teams are pre-formed, and has focused on improving the performance of the existing teams. There has been far less research done on the process of *coalition formation* – the process by which agents are grouped into teams that can be successful in a given domain. Addtionally, research that has been done in the area of coalition formation has made several key assumptions that, while making either implementations or analyses easier, are generally not true of more realistic domains. This limits the applicability of current approaches to environments with a high degree of realism. In this thesis I examine existing coalition formation algorithms, enumerate common restrictive assumptions, and propose a new coalition formation algorithm that avoids these assumptions. I will also present an implementation of this new approach, and evaluate it against a baseline implementation in a software simulation.

Contents

Acknowledgments

I would like to begin by thanking Dr. John Anderson for his support, criticism, and above all, patience. He has given me space when necessary, helpful advice when requested, and the occasional kick in the pants when most needed. I would also like to thank my parents, Art and Christine van de Vijsel and my sister, Lyndsay van de Vijsel, for always being there to support me when it felt like this project would never be completed. Finally, I would like to thank the people at the Winnipeg branch of Sierra Systems Group, as well as the various clients I have worked for over the last few years, for allowing me to have a flexible schedule and supporting me in many different ways. It has been a long road, but you have all helped me see it through, and I thank you deeply.

This thesis is dedicated to my wife Jody, for her never-ending support and love, especially when times are tough.

Chapter 1

Introduction

It has been illustrated repeatedly that agents in a multi-agent system will perform better when working as a group than when working alone [Weiss, 1999; Anderson et al., 2004; Russell and Norvig, 1995; Lerman and Shehory, 2000; Sen and Dutta, 2002]. These improvements manifest themselves in many different ways: achieving goals faster or at a lower cost, delivering a higher quality of result, or having a larger number of agents work together at an acceptable rate of performance.

However, most existing research has assumed the existence of agent coalitions, or teams, and examined how those teams can work towards improving their performance [Fontan and Mataric, 1996; Balch, 1999; Anderson et al., 2002; Veloso and Stone, 1998]. There has been far less work done in the area of *coalition formation* – an examination of how individual, self-interested agents can work together in order to improve their own performance by forming groups dynamically.

Tosic and Agha [2004] identify two broad classes of agents in multi-agent system (MAS) research – agents used in Distributed Problem Solving (DPS) and agents that are self-interested. Tosic and Agha characterize these two groups of agents by how they handle their individual goals. DPS agents share common goals, and they work together to find a solution, without any regard to individual payoff or utility. DPS agents could therefore be considered *cooperative* agents. Self-interested agents do not necessarily share their goals, and each agent may have its own agenda. Coalition formation approaches are applicable to both types of agents, albeit in different manners.

In DPS systems (i.e. multi-agent systems that are composed of DPS agents), coalition formation research tends to be focused on partitioning the population of agents into a set of disjoint coalitions (referred to as a *coalition structure*) in such a way that each group has sufficient resources to work on a portion of the problem efficiently [Sandholm et al., 1998]. Once the coalition structure is formed, each coalition can then be optimized individually to solve its part of the problem. Multi-agent systems consisting of self-interested agents, on the other hand, take a different approach to coalition formation. They focus on how groups can form dynamically even though agents are only concerned with maximizing their own utility [Tosic and Agha, 2004].

The development of coalition formation approaches in systems of self-interested agents has been an active area of research of late [Abdallah and Lesser, 2004; Breban and Vassileva, 2002; Chalkiadakis and Boutilier, 2004; Mason et al., 2004]. However, these approaches have not been successfully applied to situations that display similar characteristics to those found in the real world. This is due either to their complexity [Axtell, 2002; Tohme and Sandholm, 1999] – often due to their roots in game theory – or by the simplifying assumptions that are made about either the agent model or the domain for experimentation [Breban and Vassileva, 2002; Cornforth et al., 2004; Brooks and Durfee, 2002].

Most of the existing methods assume, for example, that all members of a coalition are equal, and that one agent will bring the same amount of value to a group as another [Tsvetovat and Sycara, 2000; Brooks and Durfee, 2002; Yamamoto and Sycara, 2001]. In more realistic settings, however, coalition formation approaches must deal with *agent heterogeneity* – the idea that group members are likely to be different in terms of strengths and shortcomings, to a degree that will affect group participation significantly. In a more realistic scenario, a group may have to work around an inferior member, or decide to hand off essential tasks to only the most capable members. These inter-group dynamics are ignored in most coalition formation approaches.

As well, it is often assumed that being part of a coalition is always a good thing [Breban and Vassileva, 2002; Lerman and Shehory, 2000] – that all agents want to be part of a group if possible. This is not always the case – agents should have the ability to decide if a particular group fits with their own needs and reject or accept an invitation to join the group accordingly. Conversely, current members of a group should have the ability to decide if a new agent warrants an invitation to the group. Current research also often assumes that an existing coalition represents the same value to each of its members [Yamamoto and Sycara, 2001; Asselin and Chaib-draa, 2003]. Actual group members may belong to the group for completely different reasons – a weaker agent may be in the group simply to receive aid from stronger agents, while stronger agents may join a group in order to foster relationships with others. Even more restrictive is the common assumption that agents may only be part of a single group at a time [Breban and Vassileva, 2002; Brooks and Durfee, 2002] – obviously not the case if we are attempting to model more realistic scenarios.

Finally, the assumption is generally made that agents begin with perfect information about the domain, about the other agents in the domain and about all tasks that are assigned to them or will be assigned to them in the future [Pechoucek et al., 2000; Contreras and Wu, 1999]. It is easy to envision scenarios where agents would begin with very little, if any, information about their domain or other agents, and would have to learn these concepts as they move about the domain and encounter other agents. Any type of exploration domain, for example, where the goal of each agent is to uncover properties and information about its surroundings, would exhibit such characteristics.

The focus of this thesis is to provide an overview of the state of the art in coalition formation research, to examine the shortcomings of the existing approaches when applied to more realistic applications, and to propose a new strategy that attempts to overcome these shortcomings.

This introductory chapter begins with an identification and explanation of the terminology used in discussions of multi-agent systems in general, and coalition formation specifically. I then outline my motivation for pursuing this topic, and the methods used in undertaking this research. I will then outline the questions this research will attempt to answer, and provide an outline of the remainder of this thesis.

1.1 Motivation

The utility of coalition formation research lies not only in getting teams of software agents or physical robots to work together, but also in understanding the methods in which human groups are formed and maintained [Anderson et al., 2004]. People are forming groups all the time – "frequent buyer" clubs that reward participation with incentives, societies or classes where everyone shares common interests, or simple cliques where members enjoy each others' company, just to name a few.

These real-world groups, the people in these groups, and the environments in which these people interact all display a set of traits that can be used to characterize realistic scenarios:

- People generally have many different tasks to perform, which may conflict with each other, and they often use the group to help them out in completing these tasks
- People are not restricted to a single group, but may belong to many different groups at the same time
- People make individual decisions to join groups based on perceived benefits to them – joining a group is not forced on anyone
- People observe the behaviour of others, and learn their tendencies over time, which helps them interact successfully with others
- People are not all the same – everyone has distinct personalities that cause them to interact in different ways, and everyone has different abilities that affect the

ease with which they can complete their tasks

- Because of the previous point, certain people contribute more to a group than others, and may be more highly valued as members of the group

Thus, improving the realism of existing approaches by encompassing these characteristics will not only allow these approaches to be applied to more complex domains, they will also allow exploration and provide insight into groups that human beings form in their daily lives.

1.2 Terminology

In order to provide meaningful discussion on the concepts and issues that will arise in this thesis, I must first ensure that all terms are properly defined, to establish a common base of understanding.

I will start by defining the term *agent.* In general terms, an agent is an entity that possesses certain goals and is able to act autonomously in pursuit of those goals within the constraints of its environment [Russell and Norvig, 1995; Weiss, 1999]. This definition is broad enough to encompass both hardware agents in a physical environment as well as software agents running in a simulated environment. For this work, I am dealing solely with software agents – I will therefore restrict the term *agent* to mean software agent, and will use the term *robot* to designate a hardware or physical agent. Nothing in this work prevents the approach from being applied to a robotic implementation, but the introduction of a physical domain adds additional challenges and issues that are behind the scope of this thesis.

A *multi-agent system* is a system in which multiple agents reside. These can be cooperative agents that are all pursuing the same goal (normally seen in distributed problem-solving systems) or self-interested agents that are attempting to maximize their own utility. The latter may consist of agents that compete with one another, or agents that are completely unaware of one another's presence.

The next term that requires definition is *coalition*. This term has been defined in many different ways by different researchers, depending on their perspective. Shehory and Kraus [1998] defines a coalition as "a group of agents that have decided to cooperate and [agree on] how the total benefit should be distributed among them" (p. 170). Breban and Vassileva [2001] defines the term as "[a group of] agents that agree to cooperate to execute a task or achieve a goal" (p. 6). Sen and Dutta [2000] says that a coalition is formed when agents "'join hands' to take advantage of complementary capabilities, resources and expertise" (p. 287).

For the purposes of this thesis, I will define a coalition as a group of agents that have expressed a willingness to help each other (immediately or in the future). Coalitions are formed when two or more agents share a formal or informal agreement to provide assistance in completing a task or achieving a goal. They can grow as additional agents express a willingness to render aid, and they can shrink, either when a single agent withdraws its offer of help to the other group members, or when other group members withdraw their willingness to help a single agent. A coalition is dissolved when it has been reduced to a single member.

Existing research also provides subtle variations on coalitions, such as *congregations* [Brooks et al., 2000] and *teams* [Weiss, 1999]. These variations are often based on intention and/or perspective of the agents in the group. Teams generally consist of non-self-interested agents that work solely for the benefit of the group. Congregations, as defined by Brooks and Durfee [2003], are more simply an informal grouping of agents that have similar interests but do not necessarily share specific tasks or goals. Agents may also have differing perspectives on the coalition itself – one agent may be using the coalition for a different purpose than another. For example, one agent may join a coalition to receive aid in its goals, while another may join the same coalition to increase its network of known agents in order to gather information.

I must also define the term *domain* as it relates to this thesis. A *domain* is an environment in which an agent finds itself, and in which it has goals it must pursue. Domains can be physical or may exist only in software. The term *goal* is defined as a task or objective the agent is attempting to achieve. Agents will either be given goals before the simulation starts (for example, in the case of a soccer domain, an agent knows before it begins playing that its main goal is to put the ball in the opposing net), or they can be assigned goals as they move around their domain (for example, in the package delivery domain of Sen and Dutta [2002], agents receive packages during the simulation that must be delivered to a particular location).

Finally, I will provide a definition of a *realistic* domain or scenario. For the purposes of this thesis, I will define a realistic scenario as one that exhibits the characteristics that have been outlined in Section 1.1. It should be noted that due to the complex nature of the real world, there will be many characteristics of such domains that are not covered by this definition. This thesis attempts to cover those characteristics that I have identified as lacking in existing literature, to provide a reasonable scope for this type of research.

1.3 Method

I have developed a new coalition formation approach that encompasses all the characteristics of realistic scenarios, as described in Section 1.1. My approach embodies these characteristics in the following ways:

- Agents are heterogeneous, with varying abilities and attributes that determine how effective they can be within their domain
- My chosen domain provides agents with multiple, potentially conflicting goals at any given time. Agents must determine how to best approach their current set of goals in order to maximize their own performance.
- Agents can be part of several coalitions at the same time. Agents are invited to join an existing coalition or to form a new one, and may accept or reject the invitation depending on their evaluation of the coalition and the inviting agent.
- As coalitions are formed and their members explore the domain, the performance of members is continually tracked, allowing agents to learn about the tendencies of others. An agent's continued membership in the coalition is based on their performance as it benefits the coalition, and the group may remove the agent from the coalition if its performance over time is worse than other coalition members. Other research has focused on exclusion of agents from a coalition [Anderson et al., 2004], or on agents removing themselves from a

coalition [Breban and Vassileva, 2002]. This issue will be explored further in Chapter 3.

Two of the key indicators that have often been used to measure system performance in coalition formation research are *throughput* and *coalition stability* [Sen and Dutta, 2002; Lerman and Shehory, 2000; Brooks et al., 2000]. Throughput is defined as the number of tasks completed or goals achieved in a given time frame, which is an important measure of performance as it illustrates the amount of work an agent is able to complete in a given time. Coalition stability can be looked at in two different ways – the total number of coalitions in the system, and the total number of membership changes (agents joining or leaving a coalition) over time. Stability is also a key performance measure as it determines the efficacy of the groups being created. If the membership in a group is constantly changing, then the group is not providing benefits to its members – otherwise they would remain in the group.

To measure the effectiveness of my coalition formation approach, a baseline measure is needed against which to compare. After evaluating existing approaches (outlined in Chapter 2), I have chosen the partnership formation approach outlined by Dutta & Sen [Dutta and Sen, 2003]. Of all the existing approaches studied, the Dutta/Sen approach has made the most strides in removing the artificial constraints imposed on coalition formation systems. Chapter 2 will provide more details on the Dutta/Sen approach.

In order to be able to draw comparisons between the two agent models, a domain is required in which the agents will reside. To this end, I have adapted the package delivery domain described in [Sen and Dutta, 2002] to an implementation that has more realistic characteristics. I will gather results from the execution of the two agent models in this environment and compare them using the indicators I have described. A more detailed description of the adapted domain can also be found in Chapter 3.

1.4 Research Questions

This thesis addresses the following research questions:

1. Can a coalition formation approach be designed that is applicable to realistic scenarios as defined in Section 1.1?

2. How would the throughput of such a coalition formation approach compare with an approach reflecting the current state of the art?

3. How would the coalition stability of such a coalition formation approach compare with an approach reflecting the current state of the art?

This thesis will describe a new coalition formation approach to suffice as an answer to the first of these questions. It will then evaluate this approach against a baseline approach in order to answer questions 2 and 3.

1.5 Summary

This chapter has introduced the concept of coalition formation in multi-agent systems, and briefly described the many restrictive assumptions that exist in much of the current research in this area. I have described how I intend to illustrate that my proposed coalition formation approach removes these assumptions and is still successful in a more realistic environment, and indicated the coalition formation approach that I intend to use as a baseline for comparison. The last section of this chapter will outline the structure of the remainder of the thesis.

1.6 Thesis organization

The remainder of this document is organized as follows:

- Related Literature
- Realistic Coalition Formation
- Evaluation
- Conclusion

Chapter 2: Related Literature

This chapter presents a review of the background literature that exists in coalition formation research. Various approaches are discussed from areas such as game theory, electronic commerce and information marketplaces, among others. The assumptions and constraints inherent in each approach are also outlined. Special attention is given to the Dutta/Sen approach that has been implemented as a baseline for this thesis.

Chapter 3: Realistic Coalition Formation

This chapter discusses my proposed coalition formation approach, as well as a detailed description of the environment that I have implemented for experimentation.

Chapter 4: Evaluation

Chapter 4 evaluates the results of the experiments executed with the two agent models and coalition formation approaches, and answers the research questions posed in Section 1.4.

Chapter 5: Conclusion

Chapter 5 provides a summary and additional discussion of the experimental results, and outlines future work that can be done to extend and enhance the proposed approach.

Chapter 2

Related Literature

This chapter provides an overview of existing coalition formation research as it relates to the issues discussed in this thesis. Coalition formation is a topic that spans many areas of research, including game theory, cooperative distributed problem solving and multi-agent systems. This research topic is still in its infancy, and therefore many different protocols and methods for forming and maintaining coalitions have been proposed, and some terminology differs between researchers.

This section will describe the state of the art in coalition formation by outlining prior and current research in the following areas:

- Game Theory
- Coalition Structures
- Learning
- Coalition Formation and the Electronic Marketplace

I will then describe in detail the approach of Dutta and Sen [2003], as this approach will form the baseline for the evaluation of my own coalition formation approach.

Finally, I will discuss applications in a small number of other domains that have benefitted from coalition formation research, and illustrate that existing methods are able to work adequately in these domains, since their focus is narrow and they still contain many restrictions.

2.1 Game Theory

One of the first areas of research to explore coalition formation was game theory. According to Rapoport and Zwick [2000], game theory is "a branch of mathematics concerned with the analysis of the behaviour of decision makers (called 'players') whose choices affect one another" (p. 424). Researchers in this field use mathematics to attempt to describe and predict interactive behaviours among multiple parties.

Whenever we have multiple entities interacting, there exists the possibility of a subset of those entities banding together to attempt to gain more than any of them could individually. This realization planted the initial seeds of coalition formation concepts in this area.

While game theory was originally proposed as a way to model psychological systems, its application to areas such as economics and, eventually, distributed artificial intelligence was soon recognized. Researchers began to explore how multiple agent behaviours in an artificial system could be explored and predicted via game-theoretic concepts [Banerjee et al., 1999].

A detailed explanation of game-theoretic concepts as they relate to all of multi-agent systems is beyond the scope of this thesis. However, it is worth examining some of the mathematical approaches to coalition formation that have been proposed in game theory research, and analyzing why their applicability to more realistic scenarios (as defined in Section 1.2) is limited.

Shehory and Kraus [1999] note that coalitions formed by a group of autonomous, self-interested agents can still be beneficial to all members of the group, even if those agents are acting only to maximize their own expected utility. Their research outlines two separate algorithms of differing computational complexity that are applicable to non-super-additive environments (meaning that two separate coalitions are not necessarily best served by merging into a single coalition). The first, called DEK-CFM (Distributed, Exponential, Kernel-oriented Coalition Formation Model) is a Pareto-optimal algorithm (indicating that there exists no alternative solution that would be preferred by any of the agents in the system) that remains exponentially complex. Their second proposed algorithm, called DNPK-CFM (Distributed, Negotiation-based, Polynomial, Kernel-oriented Coalition Formation Model), is of polynomial complexity, making it more feasible for implementation. However, the algorithm requires that information about all agent resources and payoff functions, as well as all agent tasks, must be accessible to all other agents before cooperation can occur. These restrictions make it infeasible for an environment in which agents do not have perfect information about others.

Tohme and Sandholm [1999] propose another algorithm for coalition formation among agents that are self-interested. Their algorithm introduces the concept of revision of beliefs among agents. Their agents track conditional probabilities of the various potential outcomes of the coalition formation process and attempt to calculate expected payoff based on their incomplete knowledge of other agents. While this is a step towards incorporating realistic aspects into a game-theoretic approach, Tohme and Sandholm also acknowledge that "[although] this coalition structure supports a Pareto optimal outcome...the price paid is tractability: the computation of the optimal coalition formation process can be exponential in the number of agents and in the length of the negotiation process" (p. 4). This complexity makes implementation in a multi-agent system an unfeasible task.

Axtell [2002] goes one step further in his analysis of the application of game theory to non-cooperative agents. His research postulates that conventional game theory is unsuited to determining adequate solutions for implementations of multi-agent systems, and argues that a more evolutionary approach is required. In his conclusions, he writes:

> Conventional game-theory is ill-suited to studying the kinds of meta-stable structures that emerge and transiently survive in [dynamic team formation] ... to limit the focus of one's analysis [of multi-agent systems] to equilibria, while certainly augmenting mathematical tractability, is both highly restrictive and unrealistic, and likely to render the resulting models empirically false and operationally brittle. (p. 1087)

There continues to be significant research done in game-theoretical aspects of multi-agent systems and coalition formation [Maheswaran and Basar, 2003; Dang and Jennings, 2004; Caillou et al., 2002], mainly because of the appeal of the rigour inherent in the approach. However, all of these approaches suffer from either exponential complexity or restrictive assumptions on information required for the algorithm to operate. This makes them more suited towards applications such as DPS systems that require a rigourous calculation of one-time coalition structures.

2.2 Search Strategies & Coalition Structures

Early work in multi-agent systems made note of the fact that cooperation between autonomous, self-interested agents could still exist and be mutually beneficial to the cooperating agents [Shehory and Kraus, 1995; Ketchpel, 1994]. Initially, coalition formation in multi-agent systems was part of a three-step process for cooperative, distributed problem solving [Sandholm et al., 1998]:

1. Coalition Structure Generation: Agents are partitioned into a set of disjoint coalitions, inside of which agents coordinate their activities towards the achievement of a common goal or set of goals. This set of disjoint coalitions is commonly referred to as a *coalition structure.*

2. Solving the Optimization Problem: The coalition must determine how best to pool the tasks and resources of the member agents to solve the problems presented.

3. Dividing the Reward: The agents in the coalition receive a reward or payoff for achieving their goal(s), and must determine how to divide the reward among the member agents.

While the second and third items above would often fall into the realm of mathematical or game-theoretic proofs, the Coalition Structure Generation step became an active area of research. The problem of how best to partition a group of agents is at its heart a combinatorial problem with an exponential search space [Sen and Dutta, 2000] and there have been many proposed solutions. These are referred to as *search strategies* and have spawned a large amount of research.

Sen and Dutta [2000] have implemented an order-based genetic algorithm approach to quickly determine a structure for a given problem. Their approach is scalable and is an *anytime algorithm* - it can be interrupted at any point, and it will provide the best solution it has discovered so far. The algorithm uses genetic selection and recombination operators to generate new populations of candidate coalition structures, and then evaluates the resulting structures to determine the best so far. This continues until the algorithm is stopped or until a time limit is reached. Due to the algorithm's genetic nature, the optimality of the selected coalition structure cannot be guaranteed, but Sen and Dutta's experiments show a significant improvement over their chosen baseline algorithm. Their experiments were also able to handle larger search spaces that their baseline was not able to handle due to the exponentially increasing computational costs.

Sen and Dutta's algorithm shows good results for problems requiring disjoint coalitions, and domains in which it is possible to accurately measure and compare the utility of coalition structures. In realistic problems, however, this is often not the case. Agents should be able to join multiple groups at a time, and the value of those groups is often not calculable *a priori* - it must be observed over time. Thus, Sen and Dutta's approach will be applied most successfully to cooperative environments with well-defined utility functions in which the restriction of one group per agent can be safely made.

Tosic and Agha [2004] propose a graph-based algorithm for partitioning a given set of agents into coalitions. Their maximal clique-based distributed algorithm is based on the concept that, in a domain where self-interested agents are required to cooperate and coordinate in order to complete tasks, agents will prefer to join a group containing agents with which they can communicate directly. Thus, these groups are *cliques* (defined as complete subgraphs) of the graph representing the communication links between the agents. While finding a maximal clique in an arbitrary graph is an **NP**-complete problem [Garey and Johnson, 1979; Cormen et al., 1990], Tosic and Agha illustrate that if the process of finding the degree of an arbitrary graph node (in this case, the number of other agents with which an agent can communicate directly) is computationally small (i.e. $O(1)$) then finding the maximal cliques of the graph is a computationally feasible task. The algorithm proposed is a fully distributed algorithm that places a small computational burden on a single agent, and works most efficiently when the communication graph is sparse – that is, when each agent can communicate directly with only a small number of other agents.

While Tosic and Agha [2004] do not perform an evaluation of the proposed algorithm against existing methods, their search strategy of using a graph-theoretic approach is a novel one. However, the approach still has several drawbacks. It requires an attribute (in this case, communication links) on which to base the formation of coalitions, and if this attribute is not easily attainable, then the approach has no basis on which to execute. Agents in a realistic scenario may not have such an attribute available to them – they may simply be acting "in the dark" until other agents are encountered directly. In addition, Tosic and Agha's approach is most applicable when conditions are rapidly changing, and more transient coalitions are required (i.e. a coalition is in existence for a short time, until the agents have moved in such a way that the coalition is no longer useful). While there are certainly environments in which this is the case, a more common occurrence is the establishment of longer-term coalitions that provide lasting benefit to their members.

Scully et al. [2004] examine optimal coalition formation in a marketplace domain. The problem their work aims to address is one encountered in Sen and Dutta [2000] – the problem of coalition valuation. There are many different metrics that can be used to measure the value of a specific coalition (Scully et al. [2004] mention cost, reliability, quality and dependability as several examples). The chosen metrics are not only unlikely to be of equal importance, but may actually conflict (e.g. an agent may want to minimize both cost and time, but a lower-cost solution will take longer, and a faster solution will cost more). Scully et al. have proposed a multi-objective evolutionary algorithm (or MOEA) in which coalitions are individual members that are evolved over time. Their approach is from the perspective of an individual agent – they specifically state that they are not evaluating overall system performance and they make no assumptions about the coalition formation strategies for any of the other agents in the system. In their marketplace domain, a task is submitted to the marketplace that is comprised of subtasks, and agents must form coalitions which will then compete to be awarded the task. Their objective is to calculate the optimal coalition for an individual agent to propose in order to maximize its probability of being awarded the task. The agent must be able to evaluate the other agents in the system in terms of their abilities, in order to be able to calculate the value of potential coalitions during the coalition formation process. Their evolutionary algorithm evolves a set of coalitions which approaches optimality the longer it is allowed to execute.

Once the MOEA returns a set of coalitions, the agent then selects one coalition from the set, and proposes this coalition as its contribution to the marketplace bidding process. Scully et al. use an instance-based learning algorithm to let the agent learn which of the coalitions is successful at receiving bids, and tailor its choice of coalitions based on its experience.

Scully et al.'s algorithm indirectly raises the issue of agent heterogeneity – agents can have differing abilities that will drive how coalitions are formed. This is a key issue in the real world – we must be able to accurately determine the abilities of an individual, in order to make an informed decision about whether to join a group containing that individual. Additionally, the concept of group evaluation is also a crucial one. Not only does it provide information allowing comparisons between groups, it also lets potential members gauge the ability of the group, as well as letting the group compare its collective abilities against those of potential members. These pieces are important for measuring group dynamics.

However, Scully et al.'s marketplace domain is suited towards single-cycle, temporary coalitions that exist for a single task and are then disbanded. Their approach requires prior knowledge of all agents in existence within the system, and an ability to accurately estimate their abilities in order to determine reasonable values for the metrics being used. A coalition is disbanded once the task is complete, as it may not be suitable for the next task. These restrictions on the algorithm do not fit with the definition of a realistic scenario as defined in Chapter 1.

Search strategies, as outlined in this section, are most useful when certain strict assumptions about coalitions are inherent in the domain. They work well when the domain requires transient or short-lived coalitions, when agents have *a priori* knowledge about each other or are able to provide accurate estimates about each other's abilities, or when the domain requires a disjoint coalition structure for a divide-and-conquer approach to solving a problem. In a realistic scenario, agents do not always have enough information about each other to enumerate the possible coalitions in order to construct a search space. Even if this were possible, there would not necessarily exist enough information available to support accurate evaluation of the relative utility of potential coalitions. This limits the applicability of search-based approaches to more realistic problems.

2.3 Learning

Learning is relatively a common component of coalition formation research [Scully et al., 2004; Dutta and Sen, 2003; Abdallah and Lesser, 2004; Anderson et al., 2004], both in terms of the general idea of gathering information over time, and in the sense of employing formal machine learning algorithms. Agents in a realistic scenario will often have to start out with no knowledge (or sparse and inaccurate knowledge) of their environment or the other agents in the system, and will be required to learn about these elements over time. Despite the commonality of learning as a system component, however, there are relatively few studies that have learning as the primary focus.

Chalkiadakis and Boutilier [2004] have focused on Bayesian reinforcement learning as a tool to handle uncertainty regarding the abilities of other agents. They argue that existing coalition formation approaches require a significant amount of *a priori* knowledge about other agents in the system in order for the algorithm to be successful (also noted in the previous section). It would be more realistic to assume that agents have little information about others before a task begins, and what information they do have is likely to be inaccurate. Thus, they propose a Bayesian reinforcement learning scheme to combat this uncertainty inherent in a realistic scenario.

The learning algorithm assumes that the system is comprised of a set of agents, each with its own abilities. Each agent has a set of beliefs that represent its current estimate of the abilities of others. The model calculates the value of a coalition by estimating the sum of the rewards of each potential action by the coalition, given the agent's current set of beliefs about the coalition members. It then uses calculations of expected payoffs to suggest changes to the existing coalition structure among the agents. This scheme results in each agent proposing what it feels is the optimal coalition structure at each time cycle and the action that the coalition should take, based on its uncertain beliefs at the time.

Chalkiadakis and Boutilier's agents then apply a Bayesian reinforcement learning algorithm to attempt to reduce the uncertainty about beliefs in the abilities of others. This is done through repeated coalition formation – agents propose their current belief as to the optimal coalition structure and/or the action that should be taken. The agents vote on the proposed structure and the action to take, and if all agents in the coalition agree, the structure is adopted and the action is taken. A new system state results from the action, and payoffs are generated. Agents observe the new system state and the generated payoffs and compare them to the estimated payoffs that were used when proposing the coalition structure and the action to take. Based on the accuracy of their estimates and the generated payoffs, the uncertainties in their beliefs are updated. The entire process then begins again, with new coalition structures and actions being proposed and voted on by the coalition members.

Chalkiadakis and Boutilier evaluate their coalition formation approach using a small number of agents (3 in one experiment and 5 in another). The agents were evaluated to determine if they could correctly learn the abilities of the other agents in the system, and converge on the optimal set of coalitions (referred to as the Bayesian core) given the tasks provided. Their evaluation was reasonably successful – the generated coalition structures converged to the optimal structure in between half and two-thirds of the cases.

This Bayesian reinforcement learning approach has strong applicability to realistic applications. Chalkiadakis and Boutilier realize that agents in a realistic system are likely to be heterogeneous and with no significant prior knowledge about other agents in the system. However, their scheme still requires accurate estimation of future actions by a coalition in order to estimate coalition value. Without knowledge of rewards and the ability to estimate coalition value, the algorithm runs into difficulties calculating optimal coalition structures. In addition, their evaluation used a very small number of agents – it remains to be determined if this approach will scale to tens or hundreds of agents.

Sun and Sessions [2000] have also utilized reinforcement learning, in this case in a bidding algorithm for coalition formation. In their approach, agents are working together in a team, to accomplish a single task. Each agent is represented by two modules – the Q module, which selects actions at each time cycle, and the CQ module, which determines if the agent should relinquish control of the system. Only one agent is making decisions for the team at a time, and the agent's CQ module must determine if it is more beneficial to continue to take actions and observe the results, or if it would be more beneficial to the team if another agent were to take control and begin taking action.

Every time cycle, the CQ module of the agent that currently has control of the team determines if it will take action during this time cycle. If so, the Q module determines the action to take, the action is performed, and the resulting state is observed. The active agent receives reinforcement based on its action, which is applied to both the Q and CQ modules. The cycle then continues.

Once the CQ module of the active agent determines that it would be beneficial to relinquish control and allow another agent to take action, a bidding process begins among all the other agents. An agent observes the current state, and if it feels it knows what to do in such a state, it will bid highly to take control. If an agent does not recognize the current state, then it will not bid highly and wait for a more recognized state.

Sun and Sessions's experimental domain is a set of mazes through which the agents must travel. In these mazes, many of the states (or locations in the maze) appear to be identical. Agents must learn that states that appear identical represent different places in the maze. This learning is based on the actions and observations of the other agents on the team.

Their approach provides reasonable performance but assumes that the coalition structures are defined before the system begins. That is, the agents coordinating amongst themselves are part of a single coalition, pre-defined before the system begins to execute. The reinforcement learning algorithm allows the agents to determine the abilities and location of each other in order to complete the given tasks.

Anderson et al. [2004] have applied reinforcement learning and coalition formation to the domain of robotic soccer. In their work, agents are part of a robotic soccer team where each agent has differing abilities, and no agent begins with an understanding of each other's abilities. They postulate that informal, dynamic coalitions within a team of soccer agents will allow "good" agents to realize that they can work well together, and exclude the "bad" agents from having a significant impact on the team's performance. This mirrors quite well the approach of human soccer teams, especially in situations like children's sports, where there is a significant diversity in player skill. Poor players will be excluded while the skilled players play mostly among themselves.

Anderson et al. use the Q-learning approach [Sutton and Barto, 1998] to have the agents learn estimates of the skill of other players through observing episodes of behaviour. This allows the development of the reputation of observed agents over time. They also allow an agent's reputation to be unaffected by occasional errors in judgment or poor decisions. Consistently repeated errors, however, will eventually damage an agent's reputation. They then introduce a simple, informal coalition formation technique to allow the agents to decide from whom to learn. By limiting learning to only the good agents, and avoiding reinforcement of poor habits from the bad agents, the overall performance of the system improves significantly.

This research provides an interesting approach, using coalition formation to control the direction of reinforcement learning. However, it has only been applied to small numbers of agents, such as a single soccer team, so the scalability of this approach has not been demonstrated. Coalitions in this approach are implicit – each agent has its own idea of the abilities of others, and while some of that information is shared between agents, this is only done to facilitate learning. In addition, the agents in this work are not self-interested – all agents have the same goal, to put the ball in the opposing net. There are no conflicting goals to consider – the only conflict is the determination of the action to take that will move the team closer to its collective goal.

2.4 Coalition Formation and the Electronic Marketplace

In addition to the approaches outlined in the previous sections, which focused on a particular aspect of the coalition formation process (e.g. learning, searching), there have been many other proposed approaches that have been evaluated in numerous domains. There have been a significant number of coalition formation approaches that have focused on the electronic marketplace.

The electronic marketplace is a domain in which buyer agents attempt to purchase goods or information from seller agents. Typically, buyer agents can benefit from coalition formation because the purchase of larger quantities of items from a seller means that the seller can provide a lower price per item (e.g. wholesale purchasing) [Tsvetovat and Sycara, 2000]. Seller agents can also benefit from coalition formation approaches – a group of seller agents will be able to sell to more buyer agents than a single seller agent [Yamamoto and Sycara, 2001]. In these types of approaches, coalitions are viable if the cost of maintaining the coalition is less than the discount received by buying large quantities at reduced prices. I have chosen to discuss these electronic marketplace approaches together, as they share many of the same concepts and use much of the same terminology.

For example, the coalition formation approach of Yamamoto and Sycara [2001] is concerned with large groups of buyers, such as might be found on commercial web sites. They correctly note that existing coalition formation schemes cannot handle extremely large groups of agents due to their computational complexity.

Their domain is a reverse auction system where buyers group together and pool their demand, and sellers bid discounted prices to the groups of buyers. In this domain, buyer agents create groups based on a particular product category (e.g. cameras). Anyone wishing to purchase an item in the specified category becomes part of the buyer group for that category, and then can post the items they are interested in. Postings can be made as single or multiple items, as well as OR postings – an agent can say that they would like to buy item A for price X OR item B for price Y.

Once all buyer agents have made their postings, a leader agent (simply assigned by the system in the described implementation, but open to future work for leader elections) conducts an auction with the seller agents to get the best price for the quantity desired. Once the auction is closed, the leader agent splits the buyer group into coalitions, assigns one or more selling agents to each coalition, and calculates the price that each buyer will pay. If other buyer groups for the same category have secured a lower price for the same item, then the leader agent will tell the coalition to join another buyer group to get a better price.

The core of the system is the way that the leader agent of the buyer group partitions the group into coalitions. Yamamoto and Sycara use set theory to outline their algorithm, which is summarized by simply maximizing the utility of the largest and most valuable coalition first, then the next largest and most valuable, etc. until all coalitions have been formed.

Yamamoto and Sycara's algorithm is a good approach for large-scale marketplace domains, as it can handle large numbers of buyer and seller agents. It would work well on an internet commerce site where significant numbers of transactions were taking place. However, it assumes that coalitions only exist for a single transaction, and that agents only have a single goal (to purchase or sell goods of a particular category). In addition, there is no heterogeneity in agent abilities. These factors make this approach unsuitable to realistic scenarios.

Lerman and Shehory [2000] have proposed another electronic market approach to coalition formation. Their approach is also intended to scale to thousands of agents, and is intended for internet-based marketplaces.

In order to keep complexity down and allow for scalability, their approach uses a very simple buyer agent model. In this model, implicit coalitions arise as emergent behaviour from a simple set of rules for each buyer, rather than have explicitly created coalitions. This simple model requires some relatively strict assumptions. All agents in the system are homogeneous – they have a specific product that they continually purchase, and they follow the same coalition formation strategy. Their encounters with other agents are random, and their strategy is completely driven by conditions local to their environment – no learning or history is used in making decisions. Once an agent is part of a coalition, it may leave the coalition with a pre-defined probability – this is a random act and is not a decision made by the agent. Lerman and Shehory have also placed a maximum limit on the size of a coalition, reasoning that selling agents have physical limits of the amount of product that they can sell, so if every agent in the system were to purchase from a single seller, it would not be able to fulfill the order.

The system works by starting an "order-placing" cycle that remains open for a period of time. During this cycle, agents are provided random encounters with other agents or coalitions. The agent evaluates encounter in turn, deciding to stay with the coalition (or form a new one with another single agent) by placing an order for its product. At any time, it can withdraw this order, leaving the coalition and beginning its search for another one.

Once the order-placing cycle is complete, the agents have their placed orders filled by the marketplace, and the price the agent pays is based on the size of the coalition that it has joined. Another order-placing cycle begins, in which the agent can withdraw its order and change coalitions, or remain where it is.

Lerman and Shehory present some interesting results. They find that while there are some small utility gains made when no coalition detachment is allowed (i.e. once agents choose a group, they remain there), introducing even a tiny detachment rate (e.g. a 10^{-5} probability that an agent will leave its current coalition) more than doubles the overall utility achieved in the system. However, the higher the detachment rate, the longer the system takes to stabilize, even though the utility remains high.

As with other marketplace implementations, the restrictions placed on the agents work well for the domain in question, but cause significant problems when applied to other, more realistic domains. Agents are restricted to a single coalition, have only a single goal to pursue and display no heterogeneity – all agents behave in the same way. In this work, these restrictions were conscious choices made to increase the scalability of the approach by keeping the agent model simple. Even though these restrictions improve scalability, however, they limit the approach's applicability to more complex situations.

Tsvetovat and Sycara [2000] have also examined coalition formation as it relates to the information marketplace. They outline two classes of coalition formation approaches for an information marketplace – pre-negotiation approaches and post-negotiation approaches. In a pre-negotiation approach, a single agent negotiates a price with a buyer or seller. Once the price has been secured, it advertises this price and invites other agents to join a coalition to receive the price. If the coalition ends up being too large, then the agent could have negotiated a lower price due to more quantity being purchased or sold. If the coalition ends up being too small, the buyer or seller makes less profit on the deal and will be less inclined to deal with the negotiating agent again. Thus, the pre-negotiation strategy carries significant risk due to the uncertain coalition size when negotiating.

In the post-negotiation approach, the coalition is formed first, and then a single agent does the negotiations for the group. This diminishes the uncertainty when negotiating, but adds the additional complexity of trust in a leader, as well as the entire leader election process. A collective negotiation strategy would also be possible, but has not been performed.

Ultimately, the applicability of this approach to realistic scenarios hinges on the long-term nature of the coalitions. The research implies that the created coalitions are only temporary, and exist for a single transaction before being disbanded. Agents have only a single focus – buying or selling goods – and do not have conflicting goals or decisions to make about joining multiple coalitions in a time cycle.

An interesting variation on coalitions is provided by Brooks and Durfee [Brooks et al., 2000; Brooks and Durfee, 2002, 2003]. Their approach creates teams of agents called *congregations*, specifically intended to model how humans organize themselves, indicating that clubs, churches, marketplaces and departments can all be considered types of congregations. Their distinction lies in the perceived formality of coalitions, where every agent has a distinct role or specific types of tasks they are suited for. A congregation, by contrast, is an informal structure where agents are loosely coupled, and when aid is required from a congregation member, the requesting agent can search for or look up an agent that might be helpful. Since congregating agents are expected to have long lives, their roles and suitable tasks may change over time, leading to a more fluid, dynamic group.

Brooks et al. [2000] present the following characteristics of congregations:

- Agents are individually rational and self-interested, but congregations do not have "group rationality." The group as a whole is not concerned with rewards that must be split among its members – any rewards that are generated are for individual agents only.

- Agents may be part of many different congregations, and may join or leave any of them voluntarily at any time.

- An agent's satisfaction with its membership in a congregation is dependent upon the members of that congregation. If an agent does not have the ability to achieve certain goals, it will prefer to congregate with those agents that help it achieve those goals.

- An agent's existence is long-term, and therefore it will have repeated interactions with other agents in the system. History is important here – these are not transient relationships that have no subsequent value.

- There is a cost associated with searching for partners to interact with, as well as to advertise the agent's own abilities and availability. One of the main functions of congregations is to reduce the cost of finding suitable interaction partners.

In order to facilitate congregating among their agents, Brooks et al. create *loci*, or places where agents can congregate. They also have additional agents, called *labellers*, that place particular labels on loci in order to attract congregating agents to those places. Agents are placed into "affinity groups" where agents share characteristics and preferences. The goal is to see if the agents will be able to find the members of their affinity group by congregating in appropriately labelled loci.

Subsequent research [Brooks and Durfee, 2002] applies the concept of congregations to electronic markets. In this research, the loci become marketplaces where agents congregate to buy and sell types of goods. Each agent has a price it is willing to pay for goods, and certain preferences – it will pay more for types of goods it prefers. Agents choose a marketplace, an auction is conducted for the buying and selling of goods, and then agents are free to remain in the marketplace they are in (e.g. keep their current congregation) or change to another market.

This implementation has restricted the agents to a single congregation at a time – if an agent is receiving good value, they will stay; if they are unhappy with their profit, they will leave for another congregation. As well, agents have no real conflict in terms of tasks to be performed – in both the affinity group domain and the marketplace domain, the only required action causing any conflict is whether to remain in the current congregation or try another one. In Brooks and Durfee [2003], they also note that "the affinity group is a useful domain for studying congregating, but it is considerably more simple than real-world problems." Congregations are useful concepts, and approach the concept of a coalition that this thesis is exploring, but the implementation remains restricted by assumptions that limit its realistic applications.

Breban and Vassileva [2001, 2002] have created a coalition formation approach that is based on long-term coalitions and trust relationships between agents, rather than local, temporary coalitions. They also focus on the electronic marketplace, but rather than focus on the utility of the coalition or on maximizing individual interactions, they turn to trust as a key factor in creating coalitions.

A key differentiator in their research is the inclusion of trust relationships between agents. They have adopted a formal model of trust that essentially keeps a balance of the interactions between two agents, much as Dutta and Sen [2003] have done (discussed in Section 2.5). When two agents interact, each agent evaluates their impression of the interaction. If the interaction is deemed a positive one, then the trust balance kept by the agent is increased. If the interaction is negative, then the trust balance is reduced. The authors also add an inflation factor which discounts past interactions in favour of more recent ones. This can be considered analogous to a learning mechanism for determining the agents that have compatible beliefs to our own.

The domain used by Breban and Vassileva is that of an information-trading marketplace, populated by both vendor (i.e. selling) agents and customer (i.e. buying) agents. They assume that interactions between customer and vendor agents is provided by an outside party – a matchmaking agent, or some mechanism for exploring the agent space of the system. In their coalition formation approach, there are two phases to every interaction between two agents:

- Negotiation
- Coalition Reasoning

Once two agents begin an interaction, the negotiation phase begins. In this phase, the customer and vendor attempt to agree on a price for the item being exchanged. This interaction assumes that end users (for whom each of the agents is working) have specified preferences for items, maximum/minimum prices for negotiations, etc. These preferences are critical for evaluating the interaction once the transaction has been completed, and can include such items as minimum price to sell/maximum price to buy a particular item, time constraints for executing the transaction, the importance of money to the agent's user, risk factors, etc.

At the end of the negotiation phase, a price is agreed upon. This price may then be discounted if the vendor and customer agents are part of the same coalition.

This completes the transaction between the two agents, who now (both customer and vendor) evaluate the interaction from their own perspective. A transaction is classified as a rejection (and therefore a negative experience) if a deal could not be reached between the buying and selling agent. A rejection occurs when the user preferences of the customer and vendor agents reflect some incompatibility between their respective users, and indicate a lower likelihood that agents representing these two users will interact successfully in the future. A successful transaction, on the other hand, translates into a positive experience.

The agents take their evaluations of the interaction and update their trust balances in each other, applying the discount factor for previous encounters so that more recent encounters hold more weight than encounters from long ago.

Once both the agents' trust relationships have been updated, the agents then proceed individually to the coalition reasoning phase. In this phase, the agent makes a decision to either change coalitions or remain in the current coalition. In order to do this, the agent must first classify all its current trust relationships, and order them from most to least trusted. Once this ordering has occurred, the agent can then update its current coalition status based on the results.

Breban and Vassileva present three strategies for performing this coalition update:

- *ind*: individually oriented strategy in which the agent will always prefer to be in the same coalition with the single agent in whom in has the most trust.
- *soc1*: socially oriented strategy in which the agent will always prefer to be in the coalition with which it has the largest summative trust value for all of its member agents.

- *soc2*: socially oriented strategy in which the agent will always prefer to be in the coalition with which it has the most positive trust values for all of its member agents.

The agent ensures it is in the correct coalition (according to the strategy it is assigned) by either joining an existing coalition or forming a new one. Once the coalition reasoning phase is complete, the agents are given new agents with which to interact, and the cycle begins anew.

Breban and Vassileva ran three separate experimental setups – a *simple* setup in which there were no additional factors other than what has been described, a *costs* setup, where agents must pay a cost for leaving a coalition, and a *prob* setup, in which a customer agent has a higher chance of interacting with a vendor agent from its current coalition than with a vendor agent from outside its coalition.

The results from the experimentation show that in most cases, the number of coalitions increases quickly, then decreases as coalitions begin to merge. In many cases, the number of coalitions decreased to 1, which provides global utility gain for the customers, but means that all vendors are selling at a discount, which is not a sustainable state. The authors conclude that the *soc2* strategy is the preferred strategy for both system stability and individual gain, followed by the *ind* strategy.

This coalition formation approach from Breban and Vassileva, along with Brooks and Durfee's congregating model, provides significant steps towards creating a realistic model of coalition formation. Coalitions are now being considered as long-term groups, rather than transient relationships forgotten once the next time cycle begins. Agents are learning about the preferences and abilities of other agents in the system, and are using this knowledge to drive the formation of coalitions that are mutually beneficial.

However, these approaches still have significant limitations with regards to applicability in domains that exhibit realistic characteristics. In Breban and Vassileva's model, agents have no conflicting goals or decisions to make about how to achieve them – they are simply purchasing items. Agents are also restricted to a single coalition – which works in this domain, since the agents are also restricted to a single goal. Once multiple, potentially conflicting goals are introduced, participation in multiple coalitions helps significantly. Another assumption made in Breban and Vassileva's work is the global availability of coalition memberships. Without the knowledge of all the coalitions in the system and their memberships, their coalition formation strategies are not possible, as agents will not know in which coalitions their most trusted agents reside. In a realistic scenario, it is rare that memberships are made public – the managers of the coalition may know who is part of the group, but agents at large will likely have no idea.

These electronic marketplace implementations have made strides in recent years towards a more realistic approach. Recent implementations have begun to move towards more long-term coalitions that are mutually beneficial to both buyers and sellers. However, as outlined in this section, there are still steps to be taken before these approaches can be implemented in a realistic scenario.

2.5 Dutta and Sen's Partnership Implementation

In order to evaluate the coalition formation approach proposed in Chapter 3, a baseline implementation is needed. This approach should reflect the most realistic approach available that is implementable in the chosen experimental domain.

For this purpose, I have chosen the partnership formation approach described by Dutta and Sen [2003]. This approach allows for long-term partnerships between agents, as well as learning about the abilities of others. Agents are heterogeneous, and have expertise in a given type of task that others may not possess. They interact with reciprocity – help rendered by one agent to another increases the likelihood of the second agent returning the favour in the future. The results show that such reciprocative behaviour overcomes selfishness in their chosen domain. These are all qualities that realistic approaches should exhibit.

In Dutta and Sen's approach, agents are assigned tasks of different types. Each agent is given an "expertise" in a particular task type, meaning that it can complete tasks of that type faster and with higher quality than a non-expert could. Every task in the system has an associated cost, which is proportional to the time that it takes to complete the task, and inversely proportional to the quality of the completion of the task.

Their method of determining if an agent will help another is based on probabilistic reciprocity. When an agent requests assistance from another agent that it has never met before, the probability of the other agent agreeing is 50%. As further interactions between these two agents occur, the agent requesting assistance will remember interactions where it has received a savings, and maintain a positive balance with the assisting agent. This increases the likelihood that the asking agent will reciprocate in the future when help is requested from it. Thus, the more often agent A helps agent B and provides a savings to agent B, the more likely that agent B will help agent A when asked. Conversely, the assisting agent incurs a cost when providing aid to the asking agent. This cost affects the saving balance being maintained by the helping agent. If the requesting agent continues to ask for help and does not provide any aid in return when asked, then the helping agent's chances of continuing to provide help are decreased.

This reciprocal relationship is defined by the equation:

$$Pr(i,k,j) = \frac{1}{1 + exp^{\frac{C_{ij}^{k} - \beta * C_{avg}^{k} - OP_i}{\tau}}} \tag{2.1}$$

where:

- C_{ij}^{k} is the cost incurred by agent k to complete task j for agent i
- β is a term used to set the initial cost that an agent is willing to incur when a previously unknown agent has requested help
- C_{avg}^{k} is the average cost of all tasks performed by agent k
- OP_i is the balance of past help that agent k currently has with agent i
- τ is a term used to set the shape of the sigmoidal probability curve

When the simulation begins, agents are assigned a set number of tasks – the same number of tasks for each agent. Agents then have the opportunity to ask each other for help. An agent will only agree to help if a *cooperation possibility* exists – that is, when the cost of the helping agent completing the task is less than the cost of the asking agent.

In Dutta and Sen's approach, agents begin in an exploratory mode, in which they do not attempt to choose the best agent from which to request assistance – they simply ask agents at random that they have not yet encountered, ensuring that they get a reasonable understanding of all agents in the system. Without accurate experience on which to draw, it would not be possible to make informed decisions about which agent can help with a particular task. Thus, the implementation attempts to ensure that all agents interact with all other agents at least once for each task type in the system, to have some gauge of the expertise of all the other agents. This exploratory phase is similar to the exploration vs. exploitation balance that is common in machine learning [Sutton and Barto, 1998].

The exploratory phase ends once each agent has completed a specific number of tasks. Agents then determine which agent to ask for help by sorting the agents they have interacted with in descending order of cost for the given task type. Each agent will choose that agent with which it has had the most successful interaction in the past. If this agent does not agree to provide aid, then it will move to the next agent on the list, and so on, until either all agents have been asked and have turned the agent down, or one of the agents accepts. If no agent accepts, then the agent will perform the task on its own.

In order to accurately determine if a cooperation possibility exists for a given task, agents must be able to estimate the cost of completing the task, both on its own and with aid from a specific agent. This is accomplished via a simple reinforcement learning scheme. After an agent completes a particular task of a specified type for a requestor, the requestor updates its time and quality estimates for that particular agent and task type. This will allow the requestor to more accurately determine the expertise of that agent in future.

An interesting wrinkle in this approach is that agents are not explicitly aware of their own expertise – they also learn which tasks they are good at by completing various tasks themselves and seeing which tasks they can complete faster and of higher quality.

If a cooperation possibility exists, then the helping agent uses Equation 2.1 to determine if it will render aid when asked. Initially, agents will have an even chance of helping, since the OP_i term will be zero. The more an agent owes to another agent, the larger the OP_i term becomes, increasing the probability that the agent will assist the other agent when asked. If the agent has already offered significant assistance without receiving savings in return, then the OP_i term will be smaller or negative, decreasing the probability of aid being rendered.

One additional note on Dutta and Sen's approach – reciprocative agents can request third party opinions on an agent asking them for assistance. Thus, if agent A asks agent B for help with a task, agent B can in turn ask agents C, D, E and F about Agent A, to gauge their opinions of Agent A before responding. The sum of the opinions of agents C, D, E and F would comprise the OP_i term in Equation 2.1.

When evaluating their approach, Dutta and Sen employ two different types of agents – a reciprocative agent (as outlined above) and a selfish agent. A selfish agent will not help other agents – it will only ask for help from others. If asked about another agent, a selfish agent will negate its opinion of that agent if the opinion is positive. Thus, selfish agents deliberately damage the reputation of helpful agents, while reporting truthfully on non-helpful agents.

The experiments run by Dutta and Sen first examine the relative performance of reciprocative and selfish agents, and show that selfish agents are outperformed significantly by reciprocative agents as the number of tasks increases. For small numbers of tasks (less than 300), selfish agents are able to complete their tasks more quickly (by taking advantage of helpful agents while not providing aid to anyone). However, as the number of tasks increases, reciprocative agents learn about the tendencies of selfish agents and actively avoid them, resulting in help being provided and task times being reduced. Similar results are obtained when examining the quality of performance – selfish agents exhibit poor quality, while reciprocative agents' quality increases with an increase in the number of tasks.

Another interesting result shows that agents with complementary expertise exhibit greater cost savings when interacting with each other than agents with matching expertise. This makes sense – if agent A is expert in task type A, it makes more sense for it to receive help on tasks of type B (in which agent A does not have expertise) than it would for tasks of type A. Thus, complementary agents work well together in this approach.

Dutta and Sen's reciprocative model provides many of the pieces of realistic applications. Agents can be part of multiple partnerships, and are heterogeneous – each agent has a differing expertise. Each agent has multiple goals to complete, of varying types, and learns about the abilities of other agents over time. However, there are still items missing from their approach. While agents do have multiple goals of differing types, there is no conflict between the goals. An agent must eventually complete them all, and there is no penalty for putting one goal aside to complete another. In addition, there is no explicit coalition formation in Dutta and Sen's approach. Agents use implicit coalitions to determine the likelihood of cooperation – an agent that has a high opinion of another will be more likely to interact with that agent than another agent of which it has a lower opinion.

However, Dutta and Sen's approach is the most realistic of the evaluated approaches, in that it avoids more of the restrictive assumptions outlined in Chapter 1 than others. Even though this approach was designed for the accomplishment of abstract tasks with no element of space, it has been adapted to include a spatial element, showing further versatility [McGrath et al., 2005]. For these reasons, I have chosen the Dutta and Sen approach as a baseline evaluation for my proposed coalition formation approach. The implementation and adaptation of this approach in my chosen domain can be found in Chapter 3.

2.6 Other Implementations

In addition to the other works described in this chapter, there have also been a small number of efforts involving the implementation of coalition formation strategies in physical environments. These are promising results as they show that coalition formation approaches can be successful in real-world environments. This section outlines these implementations.

Pechoucek, Marik, and Stepankova [2000] have developed a multi-agent system called ProPlanT (Production Planning Tool) that supports the production planning process at a manufacturing plant that produces television transmitters for a Czech Republic television station. This plant has no assembly line and no formal organization – they have a number of independent projects that are assembled according to unique documentation. There is little planning or simulation built into the manufacturing process.

The ProPlanT system provides plans for producing specified products by using a collaborative set of agents in a structured community. There are several different agent types, responsible for various tasks in the environment. The ProPlanT system maintains three separate knowledge bases about the agents in the system:

- **Cooperator base**: stores permanent knowledge about the collaborating agents' abilities
- **State base**: stores transient information about agents' current agendas and loads
- **Task base**: stores a set of pre-prepared plans on how to decompose and delegate potential requirements

Agents form coalitions based on the knowledge of the task at hand and the stored knowledge in the various knowledge bases. These coalitions are intended to balance out the load of the various tasks being completed at the same time by the system, and to ensure that the correct agents are available to complete subtasks as they come up.

While this system is quite efficient and effective in their manufacturing domain, it requires a significant amount of *a priori* knowledge about the agents in the system, and requires other, non-coalition agents to monitor the performance of the system and update the knowledge base as tasks get completed. Agents are designed for a single type of task and cannot complete other types of tasks – those must be delegated to other agents. These assumptions would not hold in a scenario filled with uncertainty and varying, conflicting tasks.

Pechoucek, Marik, and Barta [2002] have implemented a separate system called CPlanT that has been developed for humanitarian purposes, to provide planning for relief operations to war-stricken areas. The agents in their system represent physical resources, humanitarian organizations and places in need of aid. Each agent has characteristics that provide details about the agent: their location, abilities or requirements, and constraints on mobility, for example.

The system then organizes these agents into coalitions. This is done according to the resources required by a location in need of aid, and by the humanitarian organization that can best provide those resources. The coalition becomes the plan for the method by which the agents should work together to provide aid to the afflicted location. Other inputs for this process are existing alliances (certain aid groups have pledged to work together and share resources where possible) and barriers that inhibit communication between agents.

The research again requires a significant amount of up-front knowledge about the system in order to work accurately. As well, agents of a particular type are all essentially homogeneous – when groups are negotiating for resources, leaders responsible for the negotiations are simply elected randomly from the group, since all agents have the same capabilities. These constraints prevent the system from being a general-purpose approach to coalition formation for realistic problems.

Contreras and Wu [1999] have built a coalition formation system to aid in the planning of transmission line expansion in a deregulated electricity industry. They define three types of agents in their system: a generator agent, a load agent and a third-party agent. Generator agents provide electricity to the system. Load agents represent the demand for electricity in the system. Third-party agents are independent companies that own transmission lines required to get electricity to the demanding areas.

In their implementation, a coalition must consist of at least one agent of each of the three types – otherwise, a key element is missing that would prevent transmission of electricity. Each agent has a cost associated with its activity, and the coalitions are calculated using game-theoretical concepts to minimize the cost while still ensuring that all electricity demands are met. This cost allocation is at the heart of their work.

The approach is tailored specifically to cost-allocation domains where agents attempt to minimize costs while maximizing throughput. However, the many constraints placed on the agent and the system (such as the specific agent types and requirements for coalitions and the pre-knowledge of all costs to perform the tasks) limits the applicability of the approach.

A final implementation is worth mentioning because it is in quite a different vein than the others. Mason et al. [2004] have created a multi-agent system using coalition formation to produce drawings in the style of the artist Piet Mondriaan. This abstract painter produced grid-style paintings with segments of the grid filled in different colours. The system consists of agents that have specific rules for how the agent expresses itself artistically. This may be by creating lines on a canvas, or filling certain segments of the canvas with a particular colour.

Agents form coalitions as they discover that they may share certain goals with other agents, and that by helping each other they might increase their own internal "happiness". For example, a "line coalition" can be formed by agents that separately understand about line width, orientation and offsets from other lines. "Fill coalitions" are formed when two line coalitions merge with a colour agent, putting that colour into the area bordered by the two line coalitions. The system continues until all agents have merged into the "grand coalition" called the *gestalt*. The resulting picture is then complete, unless the artist decides to break certain coalitions and cause the system to continue drawing.

While the coalition formation approach here is obviously tailored to the environment, the authors feel it can be applied to other artistic domains such as writing and music, and potentially other domains. These approaches do not carry the complexity of other more realistic approaches, but certainly show that coalition formation can be applied to many different domains with some creativity.

All of these physical implementations use coalition formation to varying degrees. However, the domains to which these approaches have been applied all exhibit a narrow focus and contain restrictions that allow the chosen coalition formation strategies to work adequately.

2.7 Summary

This chapter has summarized the current state of coalition formation research, and has illustrated that many of the existing approaches make significant assumptions or place constraints on their domains that make the approaches unfeasible for realistic scenarios. The next chapter outlines the new coalition formation approach I have developed in order to remove these restrictive assumptions, while keeping the approach as generally applicable as possible.

Chapter 3

Realistic Coalition Formation

The previous chapter outlined the most significant recent work in coalition formation. Of these, only a small number of approaches had even some of the necessary characteristics to be applicable in a more realistic scenario. This chapter proposes a new coalition formation mechanism that avoids the assumptions and constraints that have held back previous methods from being adopted in realistic applications.

I will first outline the experimental domain for the evaluation of my approach, which is based on the package delivery domain described in Sen and Dutta [2002], in order that the coalition formation approach can be described with examples. I will then discuss the agent model (called the *vandeVijsel agent*[1]) and the coalition formation approach that I have designed, followed by its implementation in the package delivery domain. I will then describe my implementation and adaptation of Dutta and Sen's partnership formation algorithm in this domain (referred to here as the *Dutta/Sen agent*), and illustrate how it will be used as a baseline for evaluation purposes. Finally, I will provide a brief description of the implementation of the Coalition

Formation Simulator that I have constructed as part of this work.

3.1 Package Delivery Domain

The domain for this research is an adaptation of the package delivery domain used by Sen and Dutta [2002]. This particular paper does not deal with coalition formation explicitly, but is the precursor to the research discussed in Section 2.5 and introduces a multi-agent domain to allow the display of reciprocity among agents through evolving cooperation.

Sen and Dutta's package delivery domain is represented as a central depot and a series of radial fins. Agents are responsible for picking up packages from the depot, and delivering them to addresses located on one of the radial fins, at a particular distance from the depot (see Figure 3.1).

Agents begin in the central depot, and are assigned a specific number of packages to deliver. They must travel along the radial fins to the specified delivery address of the package, and cannot move between fins – once on a fin, they can only move toward or away from the package depot. Once they arrive at the delivery address, they deliver the package and return to the package depot. Sen and Dutta calculate the cost of delivering a package as twice the distance between the depot and the delivery address, since the agent will simply travel up the fin to deliver the package, and then travel back.

[1] Most coalition formation or partnering approaches are unnamed. Since I have been referring to those approaches by author name, I will continue to do so for my own.

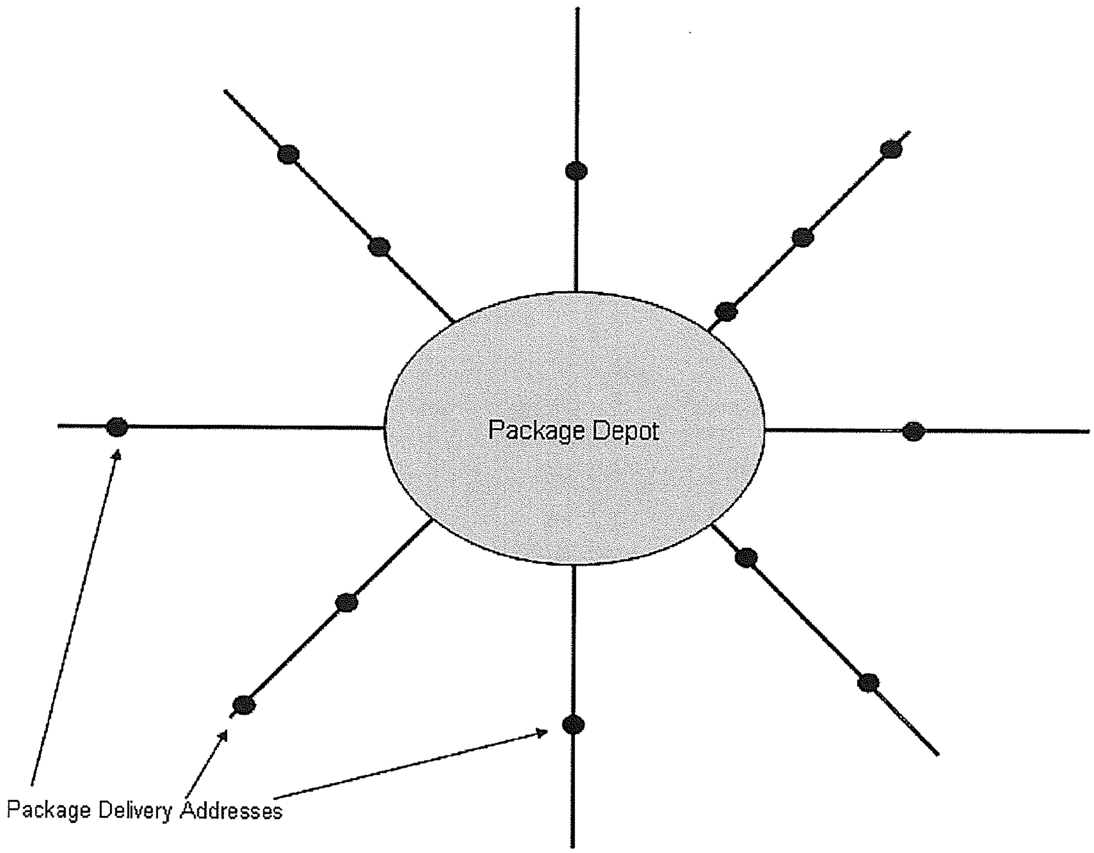

Figure 3.1: Sen and Dutta's package delivery domain

When agents are at the package depot, they can ask other agents for help delivering a package addressed to a location on a particular fin, as long as the other agent will be travelling along the same fin. Agents can only carry their own package plus one additional package, and they incur an additional unit of cost for each unit travelled carrying two packages.

I have extended Sen and Dutta's package delivery domain for this research in several important ways, in order to make it more reflective of the realistic conditions outlined in Chapter 1. The intent is to simulate a courier service that bears some of the characteristics of a real world environment from the standpoint of coalition formation, with agents acting as package couriers between addresses. One of the main points in my definition of a realistic scenario is the requirement for agents to have multiple, potentially conflicting goals. This requires supplying the agent with decisions to make, forcing it to determine which of several actions is potentially the most beneficial. To support this requirement, I have allowed agents to be assigned multiple packages upon arrival at a depot. These packages can have delivery addresses in opposite directions, providing the agent with conflicting possibilities about what action to take next. In order to implement this realistically, agents must have a greater freedom of movement through their environment. For this reason, I have moved away from the radial fin topology of Sen and Dutta and towards a grid-based implementation. Agents are allowed to move horizontally and vertically on the grid, but not diagonally.

I have also allowed for a variable number of package depots to be scattered throughout the grid. Thus, when packages are assigned to an agent, the delivery address will be another package depot somewhere else on the grid. Upon delivery of the package, the agent may receive or select additional packages for delivery to other depots (using the policy for package assignment that has been determined by the implementation of the agent model). It must then make the determination of which of its current packages it would be most advantageous to deliver next. When the grid is first created, depots will be placed randomly. However, there will be a specified minimum distance between the depots, to avoid clustering depots in a particular section of the grid.

Each package will have an initial payoff assigned to it, and the actual payoff of the package will diminish by one unit for every time cycle that passes. The initial payoff for a package, p_{init}, is calculated as:

$$p_{init} = dist(s, d) * \alpha$$

where:

- s is the grid coordinate of the source depot for the package p
- d is the grid coordinate of the destination depot for the package p
- *dist* is a function that calculates the Manhattan distance between two coordinates
- α is the system payoff factor

The initial payoff should reflect the amount of effort that it will take to deliver this package. Since the effort to deliver a package is directly related to the distance an agent must travel in order to complete that delivery, the initial payoff is thus a function of the distance between the source and destination coordinates for the package.

The payoff factor, α, is an additional factor to increase the overall payoff of the packages. This multiplication factor is required because a factor of one does not ultimately result in a positive payoff due to the diminishing return over time. With a factor of one, moving one unit of distance during each time cycle matches the payoff decrease precisely, ultimately making the payoff zero for every delivery. In addition, if the agent is carrying multiple packages at once, any other packages the agent was carrying would then provide a negative payoff. Thus, an additional factor is required in order to provide reasonable positive payoffs for package deliveries.

α should be set so that, regardless of the grid size, the agents are receiving good positive payoffs for packages delivered promptly, and reasonable negative payoffs for packages that are not delivered immediately. In order for this to occur, I have set the value of α to be a function of the grid dimensions. Via experimentation, I have found that setting the payoff factor to the sum of the grid X and Y dimensions multiplied by a factor of three, provides reasonable payoff values for package deliveries.

If a package is not delivered to a depot, for whatever reason (as we will see in Section 3.2, it is possible to lose a package), a penalty is assigned. This is intended to penalize the agent for non-delivery of the package, but should not completely wipe out an agent's currency unless it happens repeatedly. In an actual courier scenario, the loss of a single package for an experienced agent would likely mean a warning or other such sanction (perhaps paying for the replacement of the package). Repeated loss of packages, however, might be enough to warrant the loss of a job. In this simulation, repeated loss of packages will simply mean the agent's performance statistics will be lowered, likely causing the agent to be unwelcome in any group.

When a package is delivered to a depot by an agent, the payoff for that package at the time of delivery is always assigned to the agent that originally received the package for delivery, even if the original agent was not the one to deliver it (due to receiving aid from other agents). The domain allows for a package to be passed on to multiple agents if desired, although neither the vandeVijsel agent model (see Section 3.2) nor the Dutta/Sen agent model (see Section 3.3) allows this. The original agent receives notice from the simulation that their package was delivered by another agent, allowing the original agent to track the aid rendered by the delivering agent. If so desired, the original agent can then provide some remuneration to the delivering agent, although this is not required by the domain itself.

When the simulation begins, agents are randomly placed on the grid. During each time cycle, the simulation progresses through three separate phases:

- Movement Phase
- Encounter Phase
- Coalition Maintenance Phase

During the movement phase, agents are required to decide in what direction they will move during this particular time cycle, and inform the simulation of the coordinate to which they are moving. The agents are free to use whatever information they have at their disposal to make this decision. For details on how the vandeVijsel and Dutta/Sen agents make this decision, refer to Sections 3.2.1 and 3.3.1, respectively. Once the agents have made their decision, they relay that information to the simulation and they are moved appropriately. Once the agents have moved locations, if they happen to be occupying the same grid location as a package depot, then any packages that are assigned to be delivered to that depot are removed from the agent's payload. The simulation will then assign a random number of new packages to the agent. An agent may receive between zero and three new packages from a depot during a given time cycle.

During the encounter phase, agents are given a list of other agents that currently occupy the same grid location as they do. This is how coalitions are formed in this scenario – agents can only interact with other agents that they encounter on the grid, although this could easily be changed to inform agents about others within a specific distance. During these encounters, agents can discuss joining existing coalitions, forming new coalitions or receiving aid for any current packages being carried – in short, the domain allows any communication between the agents that can be implemented in the agent model. Again, for details of how the vandeVijsel and Dutta/Sen agents handle the encounter phase, refer to Sections 3.2.2 and 3.3.2, respectively.

Finally, the coalition maintenance phase exists for any actions required by a coalition that do not arise from direct encounters between agents. Certain agent models may not require any additional processing outside of these parameters, while others might. Coalitions have no direct physical presence on the grid (other than the individual presences of their members), and yet there may be a requirement for activity within a coalition, such as voting among members on an issue of interest. For example, the vandeVijsel agent approach uses this phase to evaluate the performance of each coalition's members, ensure that all members are performing as expected, and remove any members that are not contributing to the coalition (see Section 3.2.3).

Once the three phases have been completed, the simulation updates certain statistics reflecting what occurred during the previous time cycle (number of packages delivered / handed out, current time cycle etc.) and then either the next time cycle begins or the simulation ends.

This domain provides enough richness to allow for significant agent experimentation while allowing the more realistic issues in coalition formation outlined in Section 1.1 to be supported. The domain provides for multiple, potentially conflicting goals, fulfilling part of the definition of a realistic scenario proposed in Chapter 1. The remaining parts of this definition will be fulfilled by the agent model.

I will now provide details about the vandeVijsel agent model, and how it behaves in the context of this domain.

3.2 The vandeVijsel Agent Model

The vandeVijsel agent model comprises an agent with realistic limitations and a coalition formation approach that avoids many of the assumptions made in previous approaches and domains that limit applicability to the real world.

The first issue that must be addressed is agent heterogeneity. In order for this to be considered a realistic scenario, agents must differ in abilities. If all agents are identical, then there is nothing to differentiate them, and so choosing partners for coalitions becomes arbitrary.

To this end, I have put four separate attributes in the vandeVijsel agent model. These attributes are assigned random values between 1 and 10 when an agent is created, and they are unchanging throughout the lifetime of the agent. The four attributes are:

- Speed: A high speed value means the agent will be able to move through the grid at a faster rate than an agent with a low speed value. Since the courier business is highly dependent on speed, this will prove to be an important attribute for a vandeVijsel agent.

- Trust: An agent with a high trust value will be more likely to let other agents help with their package delivery. An agent with a low trust value will be more protective of its packages, preferring to deliver them itself.

- Memory: An agent with a high memory value will retain a large portion of its information, such as the locations of package depots (or any such key point in its environment) and the packages it currently has to deliver. An agent with a low memory value will occasionally forget the locations of package depots, and will also occasionally lose packages, triggering the lost package penalty to be assigned to the package's original agent.

- Honesty: An agent with a high honesty value will always portray itself accurately to other agents. An agent with a low honesty value will exaggerate its own abilities, in order to gain membership into groups that would not normally accept it as a member.

These attributes display two elements of agent heterogeneity. The first is heterogeneity in ability, represented by the speed and memory attributes. Low values for these attributes will directly affect an agent's ability to complete their tasks. The second element is heterogeneity in cooperation, represented by the honesty and trust elements. These are almost like elements of an agent's personality – low values for these attributes will affect an agent's ability to make groups and maintain relationships. For these reasons, agent heterogeneity is a key factor when considering the realism of the vandeVijsel agent model.

Since an actual courier would have physical limitations as to the number of packages he or she could carry, I have also implemented a maximum payload capacity for the vandeVijsel agent. I have set this value to 10. Agents carrying 10 packages cannot accept any more packages for delivery until they have reduced their payload, either by delivering a package or by losing one due to forgetfulness. This limitation is an improvement in realism over the Dutta and Sen [2003] approach, where agents can perform one of their own tasks each time cycle, but could also accept and perform many tasks from other agents in that time cycle.

The next few sections outline the behaviour of the vandeVijsel agent during each of the three simulation phases.

3.2.1 Movement Phase

During the movement phase, the simulation requires a decision from the agent regarding its movement during this time cycle. The agent can move vertically and horizontally, but not diagonally. It is important to note that the processing in the movement phase is largely domain-dependent. If implemented in a different domain, appropriate movement logic would need to be developed for that domain. However, the coalition formation logic (described in Section 3.2.2), is much more domain-independent.

Before the vandeVijsel agent can decide about how it will move, however, it must first take its speed attribute into account. The lower the speed attribute, the slower the agent. This is implemented by restricting the movement of the agent based on its speed attribute. I had originally simply chosen a random number between 1 and 10, and allowed the agent to move if the speed attribute is greater than or equal to that number. However, this resulted in too great a difference between agents with a speed attribute of 1 (who would only get to move 10% of the time) and agents with a speed attribute of 10 (who would always get to move). As well, this seemed to go against the courier scenario that I am attempting to model. It would not make sense to find one courier that could deliver its packages 10 times faster than another. To remedy this situation, I instead implemented a rule that allows the agent to be able to move 40% of the time, to never be allowed to move 20% of the time, and to use its speed attribute to determine its movement 40% of the time. Thus, the agent generates a random number between 1 and 20. A result between 1 and 8 results in guaranteed movement, and a result between 17 and 20 results in no movement. A result between 9 and 16 indicates another random number to be generated between 1 and 10, with movement being allowed if the agent's speed attribute value is greater than or equal to this second number. This allows for reasonable movement while still giving the speed attribute a significant effect on an agent's ability.

The speed attribute determines the likelihood of movement during a particular time cycle. However, since each agent is only able to travel one grid location during a time cycle, the speed attribute ultimately determines the agent's movement rate over a large number of time cycles. For example, an agent with a speed value of 10 will be able to move during 80% of the time cycles (40% guaranteed by the agent model, plus the full 40% determined by the speed attribute). Over time, an agent with a speed value of 10 (able to move 80% of the time) will move 20% further over the same time period than an agent with a speed value of 5 (able to move 60% of the time). Thus, the speed attribute also determines the effective movement rate of the agent.

If the above rules have determined that the vandeVijsel agent cannot move during this time cycle, then nothing else happens during the movement phase – the agent remains where it is and waits for the encounter phase to begin.

If the agent is allowed to move, however, then it must decide in which direction to move. Since this is a self-interested agent, its movements should have the ultimate goal of maximizing the payoffs that it will receive from the current set of packages it is carrying. Thus, in order to determine the direction in which to move for this time cycle, it must have selected a package to deliver. The agent does have some information available to it – it knows the destination locations of all the packages in its payload, and so it can estimate the payoff to be received from them. However, it should be noted that these are only estimates, as it cannot guarantee that it will be able to move on every time cycle, based on the rules outlined above.

In order to maximize the payoff for a single package, it could choose the package that currently has the largest payoff, and deliver that one. However, this would simply result in the agent delivering the package that is furthest away at each time cycle, since package payoff is a function of effort required for delivery. This would mean the agent turns down many of the easy payoffs that are closer, which cumulatively may provide a larger payoff than the single package that is far away.

To take this into account, the agent instead chooses a package for delivery by cycling through all the packages in its payload, and choosing the package that not only has the maximum payoff, but maximizes the sum of the remaining packages' payoffs if they were delivered from the package's destination point. In pseudocode:

for all packages p in agent's payload **do**

  $payoff = CurrentPayoff(p)$

for all packages q in agent's payload **do**

if $p \neq q$ **then**

payoff $+=$ *CurrentPayoff(q)* - *dist(p.src, p.dest)* - *dist(p.dest, q.dest)*

end if

end for

if *MaxPayoff* $<$ *payoff* **then**

MaxPayoff $=$ *payoff*

end if

end for

Select package with *MaxPayoff*

The agent will then choose and remember the destination location of this package, and travel towards that grid coordinate until it reaches it, or until some other factor changes its payload (e.g. it loses a package or it hands off or receives a package from another agent). It does not recalculate its destination at every time cycle, as things should not change from one cycle to the next, unless it gains or loses packages.

It should be noted that this is not an algorithm that is guaranteed to optimize payoffs over a set of packages – it is a heuristic approach that attempts to select the best package based on the knowledge available at the time the decision is made. When conditions change that may affect the decision that has been made (such as a change in the current payload of packages) then the decision is revisited and the algorithm is repeated.

There is an exception to the algorithm provided above. If an agent is carrying a package for another agent, the carrier will choose delivery of that package before selecting a package of its own. As I will describe in Section 3.2.2, an agent will only accept a package from another agent if it is within 10 units from its current destination. So, once the package the agent was delivering has been delivered, it would not make sense to go off in a completely different direction without delivering the other agent's package, since this location is in its vicinity. Thus, when choosing a new package for delivery, an agent will give preference to another agent's package since the delivery location should be nearby. If an agent is carrying multiple packages for others, it will deliver them in order of proximity to its current location.

This covers the decision process that an agent uses when it has one or more packages in its payload. If it currently has no packages to deliver, then it travels to the nearest package depot that it has previously encountered (and remembered). Note that an agent begins with no knowledge of the layout of its domain (e.g. the location of the package depots) – it must learn these locations over time. If an agent currently has no memory of any package depots (due to forgetfulness, or to not having encountered any yet), then it has no information on which to base its decision, and so it moves randomly. It has a 60% chance of moving in the same direction as it did the previous time cycle, and a 10% chance of moving in any of the four directions. It will continue to move randomly until it finds a package depot and receives packages to deliver, or it encounters another agent, forms a coalition with the agent, and decides to help the agent deliver a package. On the agent's very first move, it chooses one of the four directions equally.

Once an agent has decided in which direction to move, it updates its own understanding of where it is on the grid. Once it does this, if it finds itself at a package depot, then it has several tasks to perform:

- The agent must add the depot to its list of encountered depots, if it is not already on the list
- The agent cycles through its payload of packages, and delivers any packages that are addressed to this depot
- The agent asks the depot for any new packages it can deliver, if it has room in its payload

The package delivery process involves calculating the current payoff of the package, which is given by the formula:

$$p_{curr} = (dist(s, d) * \alpha) - (t_{delivered} - t_{assigned})$$

Thus, the current payoff is the initial payoff minus the time that it took to deliver the package. This value is credited (or debited, if the payoff is negative) to the agent that was originally assigned the package for delivery. If this is different than the agent that performed the delivery, then this package is considered an *assisted package*, since the delivery agent was different from the agent originally given the package.

When an assisted package is delivered, additional updates are made to any coalitions shared by the two agents. This will be discussed further in the next section.

At the end of the movement phase, vandeVijsel agents may forget pieces of information that they have stored. The first item that may be forgotten is a package that requires delivery. Since this occurs every time cycle, having even more than a slight chance of losing packages can have a monumental effect on an agent's cumulative payoff. Also, forgetting to deliver a package that we have been assigned should be a relatively rare event no matter how poor our memory is, since delivering packages is the primary function of the agent. Thus, the system generates a random number between 1 and 10000, and if (10 minus the agent's memory attribute) is less than this value, then the agent will lose a package this time cycle. This allows for a reasonable frequency of package loss without crippling an agent's payoff values. This calculation provides agents with an extremely low memory attribute to lose a package approximately once every 1000 time cycles.

Forgetting a package triggers the same process as delivering a package, except that the forgotten package results in the lost package penalty instead of the regular payoff for the package. It also triggers updates of the agent's statistics in any coalitions of which it is a member.

In addition to forgetting a package, there is an additional effect of memory – the loss of information about the location of package depots. In a realistic scenario, this would occur more frequently than forgetting package delivery, especially if the agent was in a large grid with a large number of depots. The system generates a random number between 1 and 1000, and uses the same calculation as above to determine if an agent will forget about a package depot. If an agent is to forget a depot, one is removed from its list of depots at random.

This concludes the vandeVijsel agent processing for the movement phase. Agents have now moved to their next location, delivered packages if possible, and any processing as a result of poor memory has been done. The next section outlines what occurs during the encounter phase.

3.2.2 Encounter Phase

The encounter phase is the phase where the coalition formation process occurs. Recall that during the encounter phase, agents are given a list of others that occupy the same grid location as they do. In my approach an agent considers all encountered agents individually from the standpoint of forming or extending coalitions. In order to alleviate confusion in describing this process, I will refer to the agent doing the processing simply as the *agent* and the agent that has been encountered as the *encountered agent*.

First, I will outline the structure of a coalition of vandeVijsel agents. Coalition valuation has been a significant issue in previous research [Scully et al., 2004; Sen and Dutta, 2000; Chalkiadakis and Boutilier, 2004]. The value of coalitions in my approach is determined by the membership of the coalition. Thus, each coalition maintains *coalition attributes* – values for speed, memory, trust and honesty that are calculated as the average attribute values of its membership. Whenever an agent joins or leaves the coalition, these attributes (which are available to coalition members, if desired) are updated. Thus, it is a simple matter to determine the value of the coalition once it has been created.

In addition, some members provide different value to the coalition than other members – another significant issue in current research (e.g. Anderson et al. [2004]). Thus, whenever an assisted package is delivered for a coalition, the payoff generated by the delivery is tracked by the coalition. Every coalition maintains a list of its members' average payoffs and delivery times for assisted packages, which are also available to coalition members if desired. This helps keep track of which coalition members are contributing to the well-being of the coalition, and which are dragging the other members down. Agents that are not contributing to the coalition may be required to forfeit their membership, as discussed in Section 3.2.3.

At the beginning of the encounter phase, the agents are provided with a list of agents that currently occupy the same grid location as they do. This is provided by the software simulation – in a physical environment, there would be the need for some sort of sensory perception here to identify and communicate with agents that are nearby.

For each encounter, the agent first determines if it currently shares a coalition with the encountered agent. The agent knows of which coalitions it is a member, and as a member it is allowed to check membership of other agents, although this information is not available to non-coalition members. If the two agents already share a coalition, then they have gained the benefit of being able to ask each other for aid delivering packages. The agent will not attempt to recruit the encountered agent to any coalition, nor form a new coalition, since it already has the ability to interact satisfactorily. If, in future encounters with the agent, things have changed and the two agents do not find themselves sharing a coalition, recruitment or new coalition formation could ensue.

If the agents find they do not share a coalition, then the agent will evaluate the encountered agent against all coalitions of which it is currently a member, to determine the encountered agent's suitability for each of them, based on initial knowledge of the encountered agent. This suitability is determined by the following:

1. An agent must have the sum of its reported attribute values be no more than 5 points below the attribute values of the coalition

2. An agent must not have an individual attribute value be more than 5 points below the value of that attribute for the coalition

Once the agent has determined the coalitions that appear suitable for the encountered agent, it must then order the list of suitable coalitions by the difference calculated in item 1 above, and ask the encountered agent to join each coalition in that order. It makes the most sense, from a standpoint of self-interest, for the encountered agent to be asked to join the coalition where the difference in attributes is most significantly in favour of the encountered agent. If the encountered agent has significantly more ability than the other members of the coalition, and it agrees to join, that will increase the potential of the coalition. Thus, the agent orders the suitable coalitions based on the difference between the encountered agent attributes and the coalition average. Once the encountered agent accepts an offer to join a coalition, then the process stops – once the two agents share a coalition (as mentioned above), there is no need to get the encountered agent into additional coalitions with the agent.

Determining the attributes of an encountered agent is an especially significant aspect of evaluating coalition suitability. In previous research [Dutta and Sen, 2003; Chalkiadakis and Boutilier, 2004] it has been shown that determining the abilities of others can be a significant challenge, requiring learning algorithms or other probabilistic measures. In the package delivery domain, the abilities of other agents are determined by the sharing of attribute values between agents. This sharing provides self-interested agents the means to gain access to additional resources, since the accuracy of the shared attributes depends on the honesty attribute value of the agent.

When an agent is asked to provide its attribute values for the purposes of evaluating coalition suitability, it will exaggerate its own attributes based on its honesty value. Of course, too large of an exaggeration would not go unnoticed – if an agent indicated it had a speed attribute of 10 and then only moves every third or fourth time cycle, other agents would realize it had not been honest. But a slight exaggeration, to allow it to join coalitions it would not normally be asked to join, could provide access to coalition members that it would not be able to access otherwise. Again, this is realistic from the standpoint of agent self-interest.

Thus, when a vandeVijsel agent is created, the system assigns its real attributes, which I will refer to as its *private* attributes. The agent then inflates these randomly based on its honesty value, creating its *public* attributes – those it shares with other agents. This inflation is done by dividing (10 minus the agent's honesty score) by 3, in order to get a value between 0 and 3. A random number between 0 and this calculated factor is then added to each of the agent's private attribute scores in order to create its public attributes. These public attributes are always provided when another agent is doing an evaluation of coalition suitability. They are created and stored so that an agent doesn't always give a different set of attributes every time it is asked – otherwise it would be obvious that the agent is not being truthful.

From the opposite standpoint, when the encountered agent is asked to join a coalition, it must determine if the coalition is suitable to its needs. Self-interest precludes an agent from wanting to be in a coalition with agents of significantly less skill than itself. On the other hand, it would want to be in a coalition with agents of significantly *more* skill, as it will gain benefits from highly skilled agents helping it with package delivery. The ideal situation from the point of self-interest would be to have everyone do work for the agent, while the agent itself does nothing. Such an agent would be referred to as a *leech* in a peer-to-peer application. Mechanisms must be in place to ensure that such a purely self-interested state cannot occur, and these will be explained in section 3.2.3.

When asked to join a coalition, the encountered agent evaluates the coalition in much the same way as the coalition evaluates the agent. It uses the same two criteria enumerated above, but from the opposite viewpoint. The sum of the coalition's attributes must be no more than 5 points less than the sum of the agent's attributes (using its real, private attributes this time, since it is making an internal decision for itself), and none of its individual attributes can be more than 5 points higher than any of the coalition's attributes.

There is one additional condition to be satisfied before the encountered agent will agree to join the coalition. An agent will feel less inclined to join another coalition if it is in a number of them already. The more coalitions an agent is in, the more agents it is agreeing to help when asked, and an agent may want to limit the amount of help it will provide, due to its self-interested nature. The granting of aid is not mandatory, but is dependent upon certain conditions, as discussed below. In addition, continued membership in a coalition is based on an agent's performance when helping other coalition members, as discussed in Section 3.2.3. Being in too many coalitions at the same time may make it difficult to maintain an adequate level of performance for all of them.

Thus, before agreeing to join the coalition, the agent will check how many coalitions it is in, and agree based on a sliding scale of 100% (if the agent is in 0 coalitions) down to 20% (if the agent is in 10 or more coalitions). If this final condition is satisfied, the encountered agent will then be added to the coalition. This completes the coalition recruitment process.

After the agent has considered the encountered agent for all of its coalitions, they may or may not share a coalition. That is, none of the agent's coalitions may be suitable for the encountered agent, or the encountered agent may not find the coalitions suitable for it. If the agents still do not share membership in any coalition, there is the possibility of creating a new coalition with the two agents as the charter members. This process follows the same procedure as above, except that the agents evaluate each other instead of the coalition. Thus, the agent will agree to form a new coalition with the encountered agent if the sum of the agent's (public) attributes are within 5 points of the sum of the agent's (private) attributes, and no single attribute is more than 5 points below its own attribute value, and vice versa.

At this point in the encounter phase, the coalition formation process has either been bypassed (if the two co-located agents already shared a coalition) or completed (if the agents did not share a coalition, and evaluated the possibility of joining an existing coalition or forming a new one). If the two co-located agents now share a coalition, the agent can ask the encountered agent for help delivering any of its packages. It does this by ordering its packages in descending order of payoff remaining, and asking the encountered agent if it will deliver each package in turn. Only packages that were originally given to this agent are considered for help requests – the agent will not ask the encountered agent to help deliver a package that was not assigned to the agent initially. Once the encountered agent accepts the delivery of a package, the agent will not ask about any other packages – one accepted offer of aid is enough.

The encountered agent will agree to deliver a package on behalf of another agent on two conditions:

1. It has empty space in its payload
2. Its current destination is within a specified number of units (Manhattan distance) of the delivery location of the package being handed off.

I have experimented with several different values for the specified number of units in the second condition above, and found that 10 units provides a reasonable proximity to the agent's current destination while also ensuring that agents are not sent too far out of their way to render aid to another agent.

If these two conditions are satisfied, the agent takes over possession of the package, and will deliver it once its current package has been delivered. As mentioned previously, the encountered agent will deliver the agent's package as soon as it has delivered its current package, giving preference to this package over any of its own packages rather than leave the area of the package's destination before delivering it.

This entire process is repeated for every encounter that occurs on the grid. Note that when two agents are co-located, an encounter will be generated for each of them – the implementation of the algorithm must ensure that the coalition formation process be treated as an atomic piece of code, so that if agent A forms a coalition with agent B, agent B is not attempting to form a coalition with agent A at the same time.

Once all encounters have been processed, the encounter phase of the time cycle comes to a conclusion. The next section describes the final phase in the time cycle, the maintenance phase.

3.2.3 Coalition Maintenance Phase

Any system based on reciprocative behaviour must ensure that safeguards exist that prevent agents from simply taking advantage of others. Preventing exploitation can be handled in many different ways.

In the approach of Dutta and Sen [2003], for example, agents maintain a balance of savings incurred from interactions with others. If an agent helps another agent repeatedly, then the likelihood of continuing to help that agent goes down until some reciprocative behaviour is displayed. This prevents agents from simply having others do all their work, and forces agents to provide aid before additional aid can be received.

In the vandeVijsel coalition formation approach, the coalition maintenance phase exists to allow the system to perform actions at the coalition level that are not directly related to occurrences on the simulation grid. This is where the vandeVijsel approach handles the issue of leeching agents. A self-interested agent will want to remain in a coalition for as long as possible, as there is currently no cost to remaining in a coalition (this is left for future work). Thus, the only way for an agent to leave a coalition is to have its membership revoked.

This will occur if the agent's performance over time is significantly below the average performance of the rest of the coalition members. An agent will not be evaluated for this purpose until it has assisted in delivering 10 packages for other coalition members, giving it a chance to prove itself. Agent payoffs and delivery times are recorded every time an assisted package is delivered, and so it is a simple matter of comparing the agent's average payoff on assisted packages to the coalition average. If the agent's average (on at least 10 packages) is more than a constant factor below the coalition average, the agent's membership in the coalition will be revoked. However, the agent's history with the coalition will be retained, so if it attempts to join the coalition again, the average payoff for the coalition will have to have come down to such a level that the agent's average payoff is within acceptable limits again, otherwise the agent will be refused membership. In this way, the agent acquires a lifelong reputation, that can only be altered if the agent ever does manage to join the coalition again.

The constant factor for comparison is once again based on the size of the grid. Since payoffs are directly related to delivery distance, the larger the grid, the more leeway an agent will receive. Through experimentation, I have found that the sum of the X and Y dimensions of the grid provides a reasonable constant factor when performing coalition maintenance.

Since the coalition maintenance phase occurs every time cycle, and is concerned with items occurring outside of direct grid interactions, it is also an ideal place for any additional processing that may be required by an agent model, that does not result from grid encounters.

3.2.4 Complexity

Since the overall algorithm is split into three separate phases, a discussion of complexity can consider these phases independently. The movement phase is straightforward, with complexity $O(n^2)$ where n is the size of the agent's payload. When calculating the package that will maximize future payoffs, the agent must iterate through the list of packages, and for each package, iterate again through the packages to determine their remaining payoffs if the original package was chosen for delivery. This complexity, however, is a result of the chosen method for making decisions on movement in the package delivery domain. Other domains may require a less complex (or more complex) algorithm for making this decision.

In the encounter phase, a single encounter between two agents results in a decision process that has linear complexity $O(n)$ where n is the number of coalitions of which the agent is a member. The agent must iterate through its list of coalitions, and determine the suitability of the encountered agent for each of them. Of course, once a suitable coalition is found, the process can be abandoned. This process is domain-independent and would likely be of similar complexity in any domain in which this approach was implemented, as the complexity arises from the decision process of joining coalitions and not from any domain elements.

Similarly, the decision to request aid for a package is linear in the number of packages, as the agent must iterate through each package and request aid individually. However, this is domain specific, and agents in another domain may have a less complex (or more complex) decision process when it comes to requesting aid from another agent.

One other item of note is the number of interactions that are possible on a grid location, which can make for complex negotiations between agents. If n agents all meet at a single grid location, then there are n^2 encounters that must be processed before the encounter phase can be completed. However, this is once again a function of the domain and not necessarily of the algorithm itself – certainly the algorithm can be applied to other domains where such interactions occur more infrequently.

3.2.5 Summary

This section has described the agent model and coalition formation approach for the vandeVijsel agent. This approach displays improved realism over other approaches, as it satisfies the criteria outlined in the definition in Chapter 1:

- Agents are heterogeneous – all agents have differing abilities and attributes that change their performance in the domain and their ability to cooperate successfully with others

- Agents have multiple, conflicting goals that force them to decide between several potentially beneficial courses of action

- Agents can belong to several coalitions at the same time, and make decisions to join those coalitions independently

- The importance of agents to a coalition depends on their abilities – not all agents have the same value to the group

- Agents learn about the abilities of others over time, by comparing their performance against other coalition members. Agents that have exaggerated their abilities are removed from a coalition when their performance is significantly below the other members.

The next section discusses the adaptation of Dutta and Sen [2003]'s partnership formation algorithm to this package delivery domain, as a baseline approach for evaluation purposes.

3.3 Baseline Approach

In order to evaluate the proposed coalition formation approach, a baseline for performance is required. I have chosen the partnership formation approach of Dutta and Sen [2003] as a baseline. Details of this general approach are provided in Section 2.5. This section focusses on the adaptation of this approach to a more realistic domain, as well as issues in reimplementing this approach.

I chose the approach of Dutta and Sen [2003] as a baseline approach because it encompasses many of the factors that characterize a realistic approach (outlined in Section 1.1). As discussed in Section 2.5, the approach provides agents that are heterogeneous in abilities, and allows them to learn about the abilities of other agents over time. Agents can also participate in many different groups. Originally, this approach was implemented in a different domain, so some changes are required to allow accurate comparisons to my own approach.

Dutta and Sen's agents require expertise in particular types of tasks that sets them apart from the other agents in the system. In order to implement this aspect of the approach, I have given each Dutta/Sen agent expertise in a particular quadrant of the grid. This can be equated to the real-world scenario of particular couriers having expert knowledge of roadways in a particular section of their city. They are capable drivers in all other areas of the city, but they know their area of the city so well that they tend to be faster in that area than other drivers. Thus, when a Dutta/Sen agent is created, it is assigned an *expertise quadrant* from 0 to 3, with 0 being the top left quadrant of the grid, 1 in the top right, 2 in the bottom left and 3 in the bottom right. This separation in expertise provides agent heterogeneity, analogous to the agent attributes in the vandeVijsel agent (see Section 3.3.1 for the effect of expertise on an agent's movement).

Another modification is the calculation of cost. In Dutta and Sen's original approach, the cost of a task was proportional to both quality and time metrics. In the package delivery domain, time is the only factor that determines the cost of a delivery – there is no quantitative concept of quality that could be assigned to a package delivery. Either the package is delivered, or it is lost. Thus, the calculation of cost becomes equivalent to the delivery time for a package. The higher the delivery time, the higher the cost and the lower the payoff.

One final addition is that of agent performance tracking. An agent must learn the expertise of other agents as it receives aid, so that it can estimate the cost of another agent delivering a package in a given quadrant, which is similar to the vandeVijsel agent learning about the abilities of others. The type of a task in Dutta and Sen [2003] is equivalent to the delivery quadrant in my package delivery domain. Thus, an agent will track the savings generated for each task type (i.e. each quadrant) from a given agent. This will allow it to learn the expertise quadrant of another agent over time.

3.3.1 Movement Phase

The movement phase for a Dutta/Sen agent is very similar to the movement phase for a vandeVijsel agent (see Section 3.2.1) with some small differences. The first difference is the use of the agent's expertise quadrant. With the vandeVijsel agent model, an agent was occasionally restricted from moving based on its speed value. In the Dutta/Sen agent model, an agent will also be restricted from moving, based on its expertise quadrant, in order to facilitate a fair comparison between the two. If an agent is inside its expertise quadrant on the grid, it will be able to move on every time cycle. If the agent is outside its expertise quadrant, then the agent will only be able to move on even-numbered time cycles. Thus, in order to maximize their performance, agents will have to learn each other's expertise quadrants and match up appropriate package delivery destinations when asking other agents for help. This stays true to Dutta and Sen's work when adapted to this model – the key feature is learning about other agents' expertise.

The addition of the expertise quadrant adds an additional complexity to the Manhattan distance calculation. In the vandeVijsel agent model, it is a simple matter to calculate the distance between two points, with the understanding that this calculation may not accurately reflect the time it takes to traverse the distance, due to the effects of the speed attribute. However, in the Dutta/Sen agent model, it is necessary to be able to factor in the agent's expertise, since this could significantly change the time required to travel between two points. To accommodate this, the distance calculation can optionally take the expertise quadrant as a parameter, and will double any parts of the distance that are outside the expertise quadrant. Agents will have a predisposition to move towards their expertise quadrant if possible – so, since there are two Manhattan routes between two points (horizontally, then vertically, or vertically, then horizontally), the agent will always choose the route that passes through their expertise quadrant if possible.

Other than this implementation of the expertise quadrant, there are no differences movement phase logic between the vandeVijsel agent model and the Dutta/Sen agent model. As mentioned at the beginning of Section 3.2.1, this logic is largely dependent on the domain, and since the two agent models share the same domain, there is no need to make any significant changes to the process.

3.3.2 Encounter Phase

The encounter phase for the Dutta/Sen agent model follows the approach outlined in Dutta and Sen [2003], with some small modifications to adapt it to the more realistic package delivery domain. As in Section 3.2.2, a Dutta/Sen agent is provided with a list of agents currently occupying the same grid location, and will deal with each one in turn. I will continue to use the terms *agent* and *encountered agent* in this section.

Since coalitions are not explicit in Dutta and Sen [2003], agents are willing to ask any other agent for help with a package, as long as they estimate that there is some gain to be had by receiving help. Even if no gain is realized, the agent gains information about the abilities of the other agent. In Dutta and Sen's original model, agents were unaware both of their own expertise and of the expertise of all the other agents. In my adaptation of this model, I have provided the agent with knowledge of its own expertise. My rationale for this is that in a real-world package delivery scenario, a courier would be aware of the area of the environment that it knew best, so it would make sense for the agent to be aware of its own expertise quadrant. However, the areas of expertise of other agents must still be learned.

Agents will help each other when cooperation possibilities exist, as per the original approach of Dutta and Sen. The existence of a cooperation possibility in this domain is based on the agent's estimation of the cost to deliver a package itself (*C1*, using Dutta and Sen's terminology), and its estimation of the cost for the encountered agent to deliver the package (*C2*). These are difficult estimates to calculate accurately, since there are so many factors involved. When will the agent get a chance to deliver this package? If the package is handed over to the other agent, will it deliver the package immediately? What is the other agent's expertise quadrant, and will that affect the estimate?

To answer these questions, the agent will make some logical assumptions to allow a reasonable estimate to be produced. First, when calculating the cost estimate if it delivered the package itself, it will assume that the package in question will be the next package chosen for delivery (once its currently selected package is delivered). It will also consider its own expertise quadrant when calculating this cost estimate. Thus, the cost estimate *C1* is:

$$C1 = dist(here, curr_{dest}, exp) + dist(curr_{dest}, p_{dest}, exp)$$

where:

- *here* is the current grid location
- $curr_{dest}$ is the agent's current destination

- *exp* is the agent's expertise quadrant
- p_{dest} is the destination location of the package whose cost is being estimated
- *dist* is the distance calculation function that takes expertise quadrants into consideration

It is more difficult to estimate *C2*, the encountered agent's cost for delivering the package, since the agent has no information about when the encountered agent might make the delivery. However, the agent can assume (for estimation purposes) that the encountered agent will only accept the delivery of the package if it is within a reasonable distance of its current location. Therefore, the calculation used for *C2*, the estimate of the encountered agent doing the work, is:

$$C2 = dist(here, p_{dest}, exp_{est}) + \gamma$$

where:

- *here*, p_{dest} and *dist* are defined as before
- exp_{est} is the estimated expertise quadrant for this agent (if no estimate is yet available, then a worst-case estimate is used where the agent does not travel through its expertise quadrant)
- γ is a proximity factor for estimation

I have set γ to 10 – this indicates that the agent assumes that the encountered agent will not agree to deliver the package unless it is within 10 units of its current destination. This is in fact the same value used by the encountered agent in actuality, but does not have to be the case.

When the agent is processing an encounter, it calculates *C1* and *C2* as above, and determines if a cooperation possibility exists. If so, it asks the agent for help delivering the package.

Recall that Equation 2.1 determines the probability of one agent helping another in Dutta and Sen's approach. The encountered agent uses this probability equation to determine if it will render aid. The equation and its terms, defined for this implementation, are:

$$Pr(i,k,j) = \frac{1}{1 + exp^{\frac{C_{ij}^{k} - \beta * C_{avg}^{k} - OP_i}{\tau}}}$$

where:

- C_{ij}^{k} is the estimated cost for the encountered agent (k) to complete the delivery of the requested package (j) for the original agent (i). This is analogous to the calculation of *C2*, above, but the expertise quadrant is no longer estimated because the encountered agent is performing this calculation, and it is aware of its own expertise.

- β is a term used to set the initial cost that an agent is willing to incur when a previously unknown agent has requested help

- C_{avg}^{k} is the average cost of all tasks performed by the encountered agent (k). The encountered agent tracks its average payoff as it delivers packages.

- OP_i is the balance of past help that the encountered agent currently has with the requesting agent (i). Since agents track help received from other agents by task type (i.e. delivery quadrant) of the package, this term is calculated as the sum of the balances for all four quadrants.

- τ is a term used to set the shape of the sigmoidal probability curve

For this implementation, β is set to 0.75 and τ is set to 25. The high value for τ reflects the large costs/payoffs that occur on a large grid, and provides a reasonable probability curve for this domain. The original work by Dutta and Sen does not indicate values that were used for these parameters, so the chosen values were selected based on experimentation.

This equation defines the nature of reciprocity in Dutta and Sen's approach – agents are more likely to accept help when the requesting agent has helped them out in the past. If the probability function indicates that the encountered agent will help, then the encountered agent assumes responsibility for the package. Once the package is actually delivered by the encountered agent, the original agent receives notice that the package was delivered, and updates its balances of payoffs for the encountered agent on that type of task. It can then use this updated knowledge the next time it is calculating a cooperation possibility for this agent and type of task.

One significant difference between my implementation of this approach and the one outlined in Dutta and Sen [2003] is the lack of an exploratory phase in my implementation. In Dutta and Sen's work, agents could request aid from any agent at any time, so that a significant volume of the space of possible agents was explored before beginning to exploit this knowledge. In my domain, agents are restricted to requesting aid from those agents they encounter on the grid. There is still exploration in this domain, but it is based on the physical movements of the agents. There is never a choice from among a set of agents when requesting aid – there is simply the question of whether to request aid from a particular agent. For this reason, agents are always learning and updating their estimates – their exploration is continual, but not as extensive as in Dutta and Sen's approach.

This completes the processing of an encounter. All encounters are processed in the same manner, and then the agent moves on to the next time cycle.

3.3.3 Coalition Maintenance Phase

Since the coalitions in the Dutta/Sen agent model are implicit, agents are never removed from a coalition in the sense they are in the vandeVijsel approach. The group of other agents with whom an agent is likely to cooperate will change, but this is internal to the agent and is not reflected in any external change[2]. Protection against leechers is built into the reciprocity model that Dutta and Sen have proposed. Thus, there is no need for any processing in the coalition maintenance phase for the Dutta/Sen agent model. The simulation still triggers this phase, but no processing occurs.

3.3.4 Summary

This section has described the implementation of the partnership formation approach developed by Dutta and Sen [2003], and the required changes to adapt the approach to the package delivery domain. The next section will briefly describe the implementation of the Coalition Formation Simulator.

[2]The nature of a Dutta/Sen coalition will become important in Chapter 4.

3.4 Implementation

The Coalition Formation Simulator (see Figure 3.2) is an object-oriented implementation using the Java programming language. The system has been developed using the IntelliJ IDEA 3.0.5 programming environment and the Java Development Kit (JDK) version 1.4.2.

The architecture of the system provides several abstract classes that allow for subclassing in order to provide customized functionality. For example, abstract classes exist for **Agent** and **Coalition** that allow future implementations to provide their own logic behind these classes. Currently I have implemented three subclasses of **Agent**: **vandeVijselAgent**, which encapsulates the logic described in Section 3.2; **DuttaSenAgent**, which encapsulates the logic described in Section 3.3; and a **RandomAgent** class, which simply moves about randomly, that was designed for testing purposes.

Each instance of **Agent** or a subclass of **Agent** is modelled as an independently executing Java thread. Thus, when the system begins, it creates and starts a new thread for as many agents as requested by the user. While the computational resources required by an agent have to do with the specific agent implementation, I have been able to run 500 separate agents of either type on a Pentium III-800 MHz machine with 512MB of RAM.

In addition, the system creates two additional threads. One is in control of the animation of the Simulation Grid, if that option is turned on. The second thread manages the simulation itself, and is an instance of the **SimulationManager** class. This thread controls the simulation phases – the movement phase, the encounter phase,

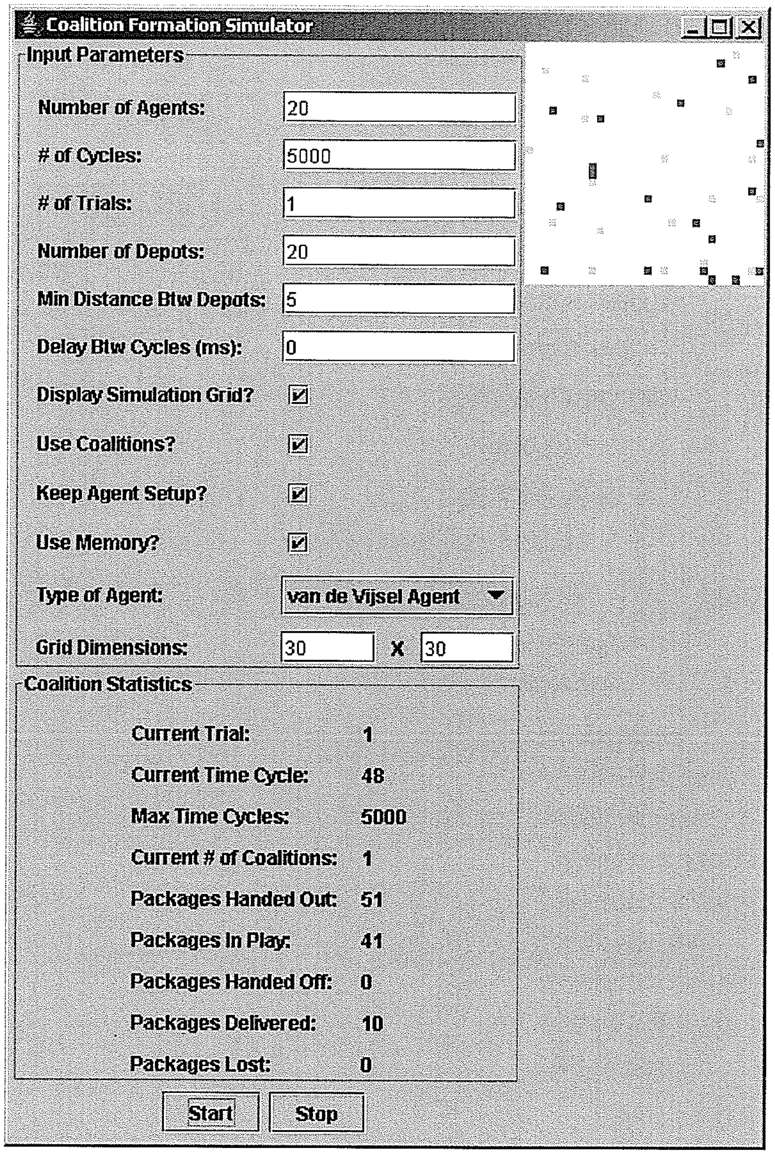

Figure 3.2: Screen shot of the Coalition Formation Simulator. Package depots are seen on the grid as lighter dots, while agents are displayed as darker dots.

and the maintenance phase. When the agent threads begin executing, they wait for the system to move into the Movement Phase state. When the **SimulationManager** moves the system into this state, the agents are free to do their required processing to determine their movements (see Sections 3.2.1 and 3.3.1). The **SimulationManager** thread requires all agents to report that they have completed their movement phase processing. Once all agents have reported, the **SimulationManager** then moves the simulation to the Encounter Phase state, and all agents begin processing their encounters (see Sections 3.2.2 and 3.3.2). Again, all agents must report that they have completed their processing. Once this is complete, the system moves into the Maintenance Phase state, and any processing that must occur in this phase begins. When the Maintenance phase is complete, the **SimulationManager** advances the time cycle ahead, and moves back to the Movement Phase state. The cycle then begins again.

There are several features built into the Coalition Formation Simulator that facilitate experimentation. A number of parameters are available for entry by the user, to customize the simulation:

- *Number of Agents*: controls how many agents are created for this simulation
- *Number of Cycles*: controls how long the simulation will run for
- *Number of Trials*: controls how many trials the system will run. Each trial lasts for the specified number of cycles.
- *Number of Depots*: controls how many package depots are placed on the grid.
- *Min Distance Between Depots*: the minimum distance placed between depots. The system will alert the user if it cannot create the specified number of depots on the grid and maintain this minimum distance.
- *Delay Between Cycles*: if displaying the Simulation Grid, this delay smooths

out the display of the agents' movement if the number is non-zero. However, the simulation will run more slowly.

- *Display Simulation Grid?*: turns the display of the grid on or off for this simulation.
- *Use Coalitions?*: determines if the agents will attempt to form coalitions. If this flag is not set, the encounter phase for the agent will not be triggered.
- *Keep Agent Setup?*: if multiple trials are being executed, this determines if the agents return to their original locations with the same attributes after each trial, or if new agents are created.
- *Use Memory?*: this turns the use of memory in the vandeVijsel agent on or off.
- *Type of Agent*: selects the type of agent to simulate. Valid types are vandeVijsel agent, Dutta/Sen agent and Random Agent (which simply moves about randomly with no logic – used for testing purposes).
- *Grid Dimensions*: determines the size of the grid.

There are also statistics that are displayed as the simulation runs. These are:

- *Current Trial*: the current trial number being run.
- *Current Time Cycle*: the current time cycle of the simulation.
- *Max Time Cycles*: the maximum number of time cycles to be executed for this trial.

- *Current Number of Coalitions*: the number of coalitions that currently exist in the system.
- *Packages Handed Out*: the total number of packages assigned to agents for delivery so far.
- *Packages In Play*: the total number of packages currently being carried by agents.
- *Packages Handed Off*: the total number of packages assigned by one agent to another for delivery.
- *Packages Delivered*: the total number of packages delivered by all agents.
- *Packages Lost*: the total number of packages lost due to forgetfulness.

Every 1000 time cycles, the system will generate several comma-delimited files, providing experimental results. These include aggregate statistics, agent-level statistics and coalition-level statistics. In addition, another file is produced that shows all coalition membership changes over the entire run of the simulation. These allow the gathering of throughput and stability information as described in Chapter 1, and will be further discussed in Chapter 4.

3.5 Summary

This section has described in detail the implementations of the vandeVijsel agent model and coalition formation algorithm, and the baseline implementation of the Dutta/Sen partnership algorithm. I have also outlined the implementation of the system in Java. The next section provides the results of experimentation and evaluates the vandeVijsel agent model against the baseline approach.

Chapter 4

Evaluation

The goals of the research presented in this thesis were to answer the three questions noted in Section 1.4:

1. Can a coalition formation approach be designed that is applicable to realistic scenarios as defined in Section 1.1?
2. How would the throughput of such a coalition formation approach compare with an approach reflecting the current state of the art?
3. How would the coalition stability of such a coalition formation approach compare with an approach reflecting the current state of the art?

In Chapter 1, I indicated the emphasis that has been placed on *system throughput* and *coalition stability* as measures of performance in previous research [Sen and Dutta, 2002; Lerman and Shehory, 2000; Brooks et al., 2000]. In this chapter, I will be using these two factors to evaluate the performance of the vandeVijsel agent model (see Section 3.2) in the package delivery domain (see Section 3.1). To facilitate this evaluation, I will also be measuring these two factors for the baseline approach of Dutta and Sen (see Section 3.3) and comparing the performance of my approach against this baseline.

4.1 Experimental Set-up

To gather the results for the experiments in this chapter, the Coalition Formation Simulator described in Section 3.4 was run on a Pentium III - 800 Mhz processor, with 512 MB of RAM, running Windows 2000 Professional Service Pack 4. To improve performance, the simulator was run without the display of the simulation grid.

4.2 Comparison of the Two Approaches

This chapter is concerned with evaluating the vandeVijsel agent model against the Dutta/Sen agent model from the perspectives of system throughput and coalition stability. However, in order to make valid comparisons between the two approaches, I must ensure that the results being compared represent similar concepts in both approaches.

From a system throughput perspective, the comparison is relatively simple. Throughput is defined as the number of tasks completed or goals achieved in a given time frame. For the package delivery domain used in this thesis, completing a task is represented by the delivery of a package. So throughput in this domain is represented by the number of packages delivered in a given time span. This can be represented either at an aggregate level, showing the total number of packages delivered by all agents in the system, or at an agent level, examining the average number of packages delivered by a single agent.

From a coalition stability perspective, the comparison between the two approaches is more complicated. Coalition stability can be considered in two different ways. First, we can measure coalition stability by examining the total number of coalitions that have been created in the system. The purpose of this measure would be to illustrate that agents are not simply creating new two-agent groups every time they meet someone new – they are attempting to integrate new encounters into existing groups, to keep the number of groups stable. Second, we can measure coalition stability by examining the rate at which agents are joining or leaving coalitions. If the number of coalitions remains relatively stable, but agents are constantly joining or leaving those coalitions, then this is a sign that the coalitions are still unstable. Some variation in coalition membership will likely always occur, since agents have the freedom to join coalitions as they choose, and coalitions have the freedom to revoke membership of agents.

When discussing coalition stability in the vandeVijsel agent model, it is straightforward to determine the number of coalitions in the system, the membership of those coalitions, and the rate at which agents are joining or leaving coalitions. This is because coalitions are explicit – agents make a conscious decision to join a coalition, and the members of a coalition make a conscious decision to revoke an agent's membership.

In Dutta and Sen's approach, however, it is less straightforward, because coalitions are not explicit. Dutta and Sen's approach incorporates learning about the expertise of other agents, and so a Dutta/Sen agent determines the expertise of the agents it encounters, given repeated interactions with those agents. With this knowledge, it more accurately estimates the cost savings to be realized when working with specific agents, and thus develops high opinions of those agents with whom it is advantageous to cooperate. Over time, every Dutta/Sen agent will have high opinions of a specific group of agents, and it will be more likely to cooperate with agents from that group. It will be less likely to cooperate with agents from outside that group, as its opinion of those agents will be lower. Therefore I can consider a Dutta/Sen agent to be in a coalition with the group of agents that it is most likely to cooperate with. In order to specify this more concretely, I must define specific criteria to determine the set of agents in the coalition. To do this, I use the opinion values that are tracked by the Dutta/Sen agent (see Section 3.3 for details). The lack of an opinion (i.e. when the Dutta/Sen agent has no information about another agent) is treated as an opinion value of zero. Therefore, a positive opinion between agents is one that has gone up since the agents met, while a negative opinion is one that has gone down. For this reason, a Dutta/Sen agent has a larger likelihood of interacting with an agent with a positive opinion than with an agent it has not met before, so I can define the criteria for membership in an agent's coalition as the set of agents with which it has a positive opinion.

I must also establish criteria as to what constitutes a membership change in a coalition. Since the Dutta/Sen coalition for a single agent is made up of agents thought of positively by that agent, tracking changes in an agent's opinion from positive to negative would indicate a change in coalition membership. However, using such a measure would be misleading. Since a Dutta/Sen agent uses reciprocity as a tool for cooperation, once an agent cooperates with another agent, it is less likely to cooperate with that agent again until it receives help in return. Its opinion of the other agent represents a savings balance that it has maintained over its interactions with the other agent. If agents are continuously being reciprocative to each other, then it is possible for their opinions to be constantly reversing polarity – the agent's opinion is positive when it receives help, then becomes negative once it gives help, then becomes positive again. This oscillation would overinflate membership changes if it were used directly as a definition.

However, over time, the savings between two agents of complementary expertise should both become positive as each agent realizes gains from being helped by the other. Thus, examining the number of positive relationships at specific time intervals in the Dutta/Sen agent model will provide an accurate picture of the change in coalition membership for a given agent. Averaging these values out over the entire agent population will give an indication of the stability of coalition membership in the Dutta/Sen approach.

One additional item of note is that dishonest agents in the vandeVijsel approach are actively deceiving their coalition partners about their own skill sets, a realistic factor that is not considered in the Dutta/Sen approach (or many others). The degree of this deception depends on the agent's honesty value. Thus, I would expect to see less stability from the vandeVijsel agents relative to the Dutta/Sen agents, as vandeVijsel coalition members realize which agents have been exaggerating their abilities. An agent that has a low honesty value and therefore significantly exaggerates its abilities will join a coalition, be discovered and removed from the coalition, join another coalition, get removed from that one, etc. So dishonest agents will likely follow a pattern of jumping from one coalition to the next, until they have been part of every coalition that they are deemed suitable for, which could take some time given the number of coalitions generated in the simulation. This dishonesty will result in higher occurrences of membership changes, but the rate of such changes should be relatively constant.

4.3 Result Files

During the execution of the Coalition Formation Simulator (described in Section 3.4), experimental data is written to a set of comma-delimited result files every 1000 time cycles.

Data for experimental trials are recorded as follows.

Aggregate Statistics

- *packages delivered*: the total number of packages delivered (up to the current time cycle) by all agents in the system
- *packages in play*: the total number of packages currently being carried by agents
- *number of coalitions*: the current number of coalitions in the system (only output for vandeVijsel agent simulations, as Dutta/Sen agents do not have explicit coalitions but instead maintain a list of positive-opinion agents)

Agent Statistics

For each agent in the system[1]:

- *agent ID*: an identifying number for this agent
- *agent type*: the current agent type
- *agent speed*: the value of the agent's speed attribute (only output for vandeVijsel agents)
- *agent memory*: the value of the agent's memory attribute (only output for vandeVijsel agents)
- *agent honesty*: the value of the agent's honesty attribute (only output for vandeVijsel agents)
- *agent trust*: the value of the agent's trust attribute (only output for vandeVijsel agents)
- *expertise quadrant*: the agent's quadrant of expertise (only output for Dutta/Sen agents)
- *currency*: the total payoff accumulated by the agent
- *number of coalitions*: the total number of coalitions of which the agent is a member (output only for vandeVijsel agents)
- *number of opinions*: the total number of opinions the agent has about other agents (only output for Dutta/Sen agents)

[1]Some of these attributes are noted as being recorded only for one agent type. This is because they are unique to that agent. See Sections 3.2 and 3.3 for explanations of each of these.

- *packages delivered*: the total number of packages the agent has successfully delivered (includes assisted packages delivered for other agents)
- *packages lost*: the total number of packages the agent has lost
- *assisted packages*: the total number of packages the agent has delivered for other agents
- *average payoff*: the average payoff value this agent has received for all its processed packages
- *average delivery time*: the average delivery time this agent has achieved for all its delivered packages (an agent's average delivery time is not updated for lost packages)

vandeVijsel Coalition Statistics

For vandeVijsel coalition purposes, statistics are tracked only for packages that are delivered by a coalition member on behalf of another member (i.e. *assisted* packages).

For each coalition currently in existence:

- *coalition ID*: an identifying number for this coalition
- *average speed*: the average speed value of all coalition members
- *average memory*: the average memory value of all coalition members
- *average trust*: the average trust value of all coalition members
- *average honesty*: the average honesty value of all coalition members

- *number of members*: the total number of agents in the coalition
- *packages delivered*: the total number of assisted packages delivered by coalition members
- *average payoff*: the average payoff value for assisted packages delivered by coalition members for other members.
- *average delivery time*: the average delivery time for assisted packages

Dutta/Sen Coalition Statistics

For the definition of a Dutta/Sen coalition, refer to Section 4.2.

- *agent ID*: the agent ID of the agent that has formed the opinion
- *opinion of agent ID*: the agent ID of whom this agent has an opinion
- *payoff balance*: the savings balance for this opinion (refer to Section 3.3 for discussion on Dutta/Sen opinions)

vandeVijsel Coalition Membership File

Note that records in this file, unlike the other result files, are written whenever a change to the membership of a vandeVijsel coalition membership is made (as opposed to every 1000 cycles for the other files).

- *Add/Delete*: whether this is an addition to a coalition or a removal
- *coalition ID*: the coalition ID being modified
- *agent ID*: the agent ID being added or removed

4.4 Experiment Structure

In order to generate the experimental data, I ran 10 trials of 50,000 time cycles for both the vandeVijsel and Dutta/Sen agent types, as defined in Sections 3.2 and 3.3, respectively. Only a single agent type was present in the simulation world at a time – the agent types were never combined. The trials were run using 500 separate agents executing on a 100 X 100 grid with 100 package depots. The minimum distance between each package depot was set to 5 units. On the hardware described in Section 4.1, a trial of 50,000 time cycles took 2.5 hours on average.

The agent setup was randomly generated for the first trial, and then retained for all subsequent trials with that agent type. On subsequent trials (i.e. trials after the first), agents were returned to their initial starting locations and their payoff totals, package payloads, and memory of package depots and of other agents were all wiped clean. All coalitions were deleted, and the system began as if the previous trial had not happened. Thus, all trials were kept as homogeneous as possible. The more homogeneous the trials, the more accurate averages between the trials will be, especially for the purposes of analyzing coalition stability. If all agents remain the same across all the trials, then the resulting coalitions should also be relatively similar, and the stability between trials should be comparable.

The following section describes and analyzes the experimental results gathered from the perspectives of system throughput and coalition stability.

4.5 Results and Analysis

4.5.1 System Throughput

In the package delivery domain, system throughput is defined as the number of packages delivered. This can be described at an aggregate level, showing how many packages were delivered over the time span of each trial, or from an agent average perspective, showing how many packages were delivered on average by an agent of either type, given these experimental conditions.

From an overall system perspective (Figure 4.1), the Dutta/Sen agents delivered a total average of 311410.9 packages over the 10 experimental trials, with a standard deviation of 3150.17 packages. The vandeVijsel agents delivered an average of 430400.3 packages over the 10 experimental trials with a standard deviation of 1808.98 packages. This represents an increase of 38%. The standard deviation numbers show consistent performance by the simulation over the 10 trials.

Figure 4.2 shows the overall average agent throughput for the vandeVijsel agent and the Dutta/Sen agent. Individual vandeVijsel agents averaged 860.80 packages delivered over the 10 trials, with a standard deviation of 178.57 packages. This standard deviation shows a reasonable difference in the individual agent averages over the 10 trials. On the other hand, individual Dutta/Sen agents averaged 622.82 packages delivered over the 10 trials, with a standard deviation of 52.82 packages, showing more consistency between the agents. Since Dutta/Sen agents are more homogeneous, differing only in quadrant of expertise, more consistency in agents should be expected.

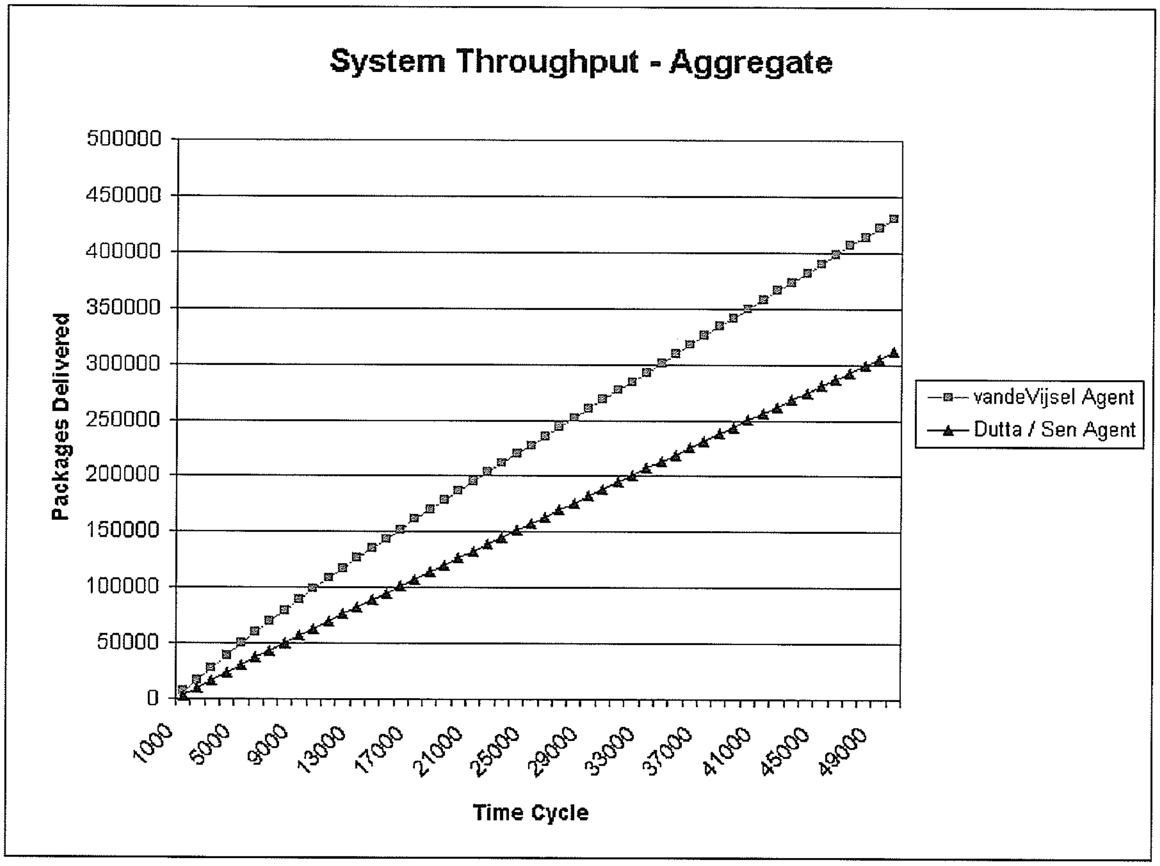

Figure 4.1: Comparison of overall system throughput for the vandeVijsel agent and Dutta/Sen agent, averaged over 10 trials

Table 4.1: Average Dutta/Sen Throughput by Expertise Quadrant after 50000 Time Cycles

Agent Expertise	Average Throughput
0	617.99
1	623.15
2	629.56
3	621.10

This point is further illustrated in Table 4.1, showing that there is no significant difference in average throughput for the different expertise quadrants in the Dutta/Sen agent model. The difference between the highest and lowest throughput value is less than 2%.

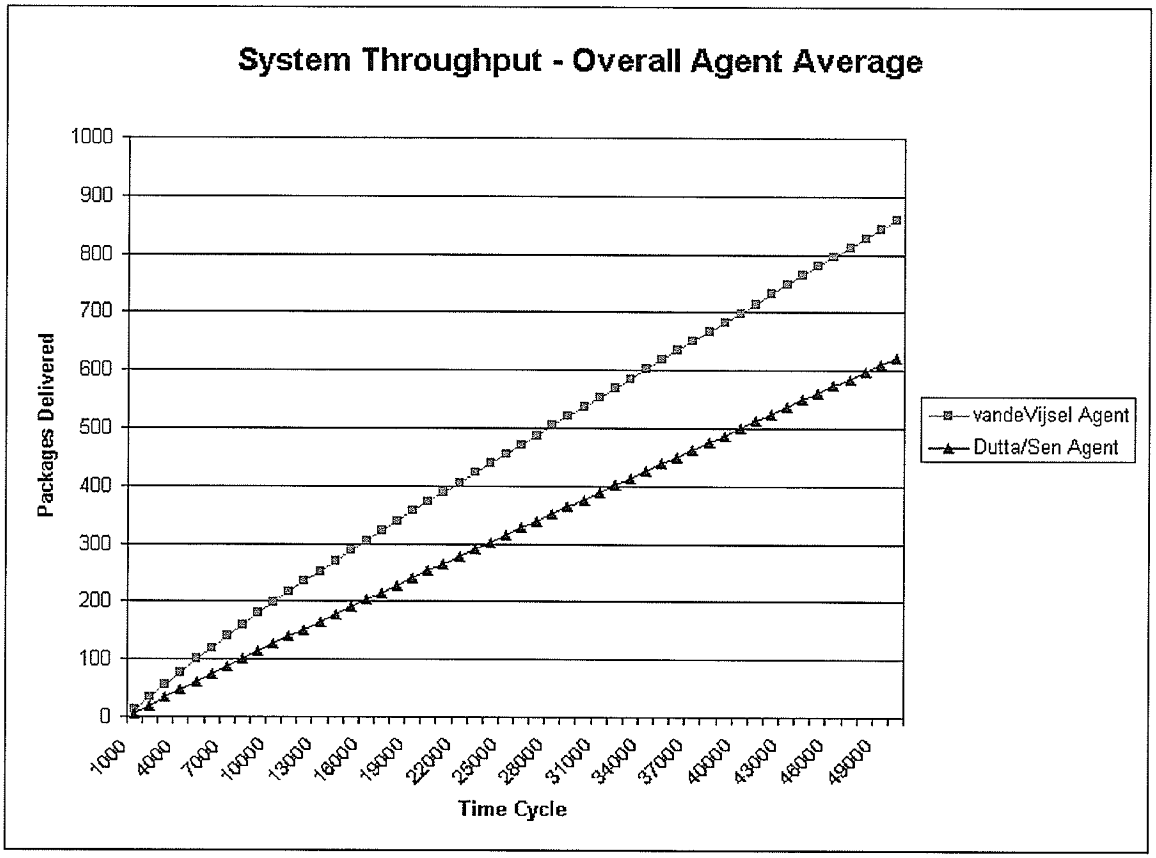

Figure 4.2: Comparison of average agent throughput for the vandeVijsel agent and Dutta/Sen agent, over 10 trials

Figures 4.1 and 4.2 show that the throughput of vandeVijsel agents is consistently higher than the throughput of the Dutta/Sen agents. Since the parameters of the domain are identical, the source of this increase must lie in an inherent difference between the approaches. The two major differences between the vandeVijsel agent model and the Dutta/Sen model (as discussed in Sections 3.2 and 3.3) are the abilities of the two agents and the respective coalition formation approaches. Although the Dutta/Sen agents are heterogeneous in that they each have expertise in different types of tasks (i.e. quadrants in the grid, as opposed to attribute-based heterogeneity in the vandeVijsel agents), Table 4.1 indicates that there is no significant difference in throughput based on the expertise of the agent (i.e. the quadrant specialization). An agent's expertise affects performance on individual tasks, not aggregated performance.

The heterogeneity inherent in the vandeVijsel agent, on the other hand, does affect the throughput for individual agents. By quantifying the effect that the difference in abilities has on the throughput of the vandeVijsel agent, and removing it from the throughput totals, I can conclude that any remaining gains in throughput are due to the coalition formation approach of the vandeVijsel agent.

In order to quantify the effect of the attribute differences on the throughput of the agent, I must first determine if there exists a relationship between each of the four vandeVijsel agent attributes and the throughput of the agent. If no such relationship exists, then the attribute in question does not affect throughput, and can be removed from the analysis (just as I have illustrated that agent expertise in the Dutta/Sen model has no direct impact on throughput). If, however, a relationship does exist, then I must attempt to determine its impact on throughput. Any remaining difference in throughput, once all four attributes have been accounted for, must be a result of the vandeVijsel coalition formation approach.

I will begin by examining the trust attribute. Figure 4.3 shows the average throughput for all agents, broken down by their trust attribute value. That is, the first data point shows the average throughput for all agents with a trust value of one, the second data point for all agents with a trust value of two, etc. The graph indicates that there is no direct relationship between the trust attribute of the agent and the throughput that agent will achieve.

In Figure 4.4, we see an analogous set of data showing the average throughput

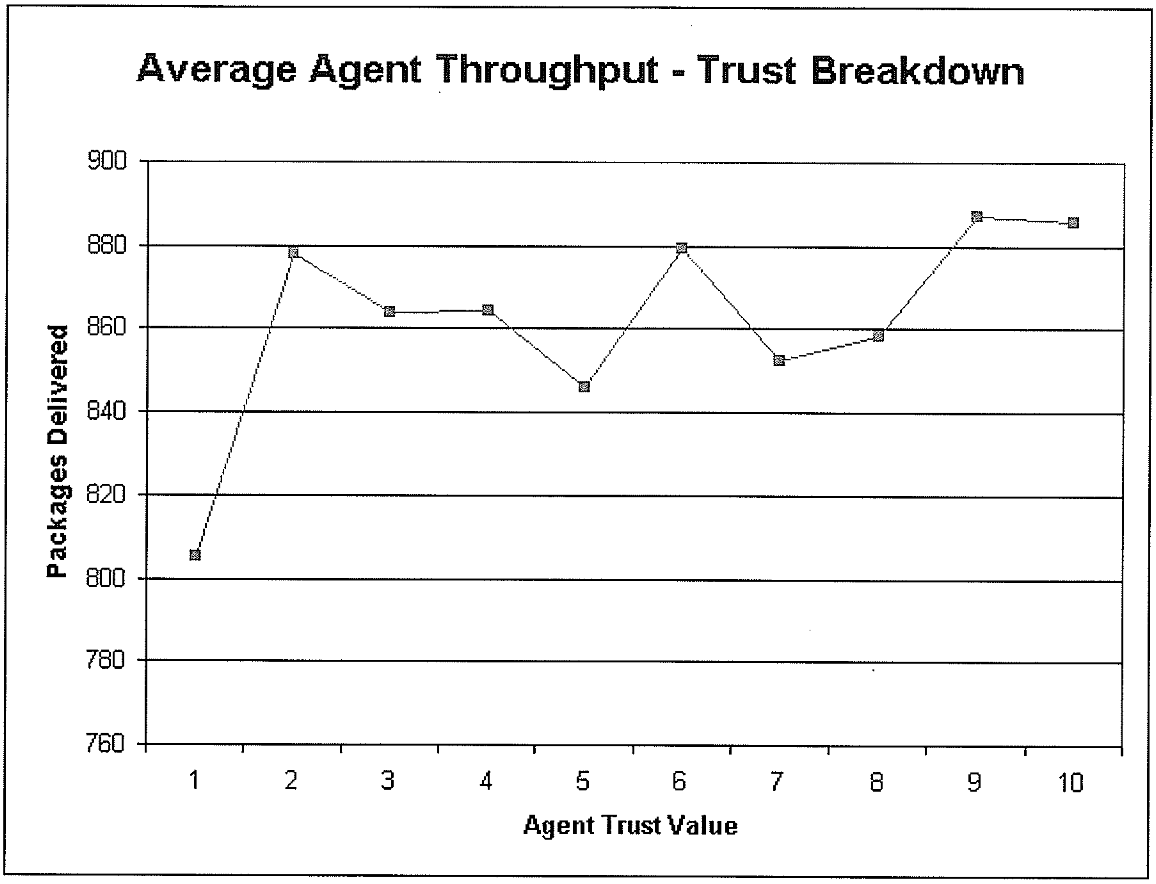

Figure 4.3: Average vandeVijsel Throughput by Trust Value after 50000 Time Cycles

for all agents in the system, broken down by agent honesty values. Once again, the graph shows no correlation between the honesty value of an agent and its resulting throughput. This is a promising result – it shows that dishonest agents do not have an advantage over honest agents from the perspective of throughput. The gains they receive from joining coalitions that they do not deserve to join are offset by their removal from those coalitions once agents learn that their actual abilities do not match those that were advertised.

Figure 4.5 provides the average agent throughput, broken down this time by agent memory attribute values. The graph again shows a lack of correlation between mem-

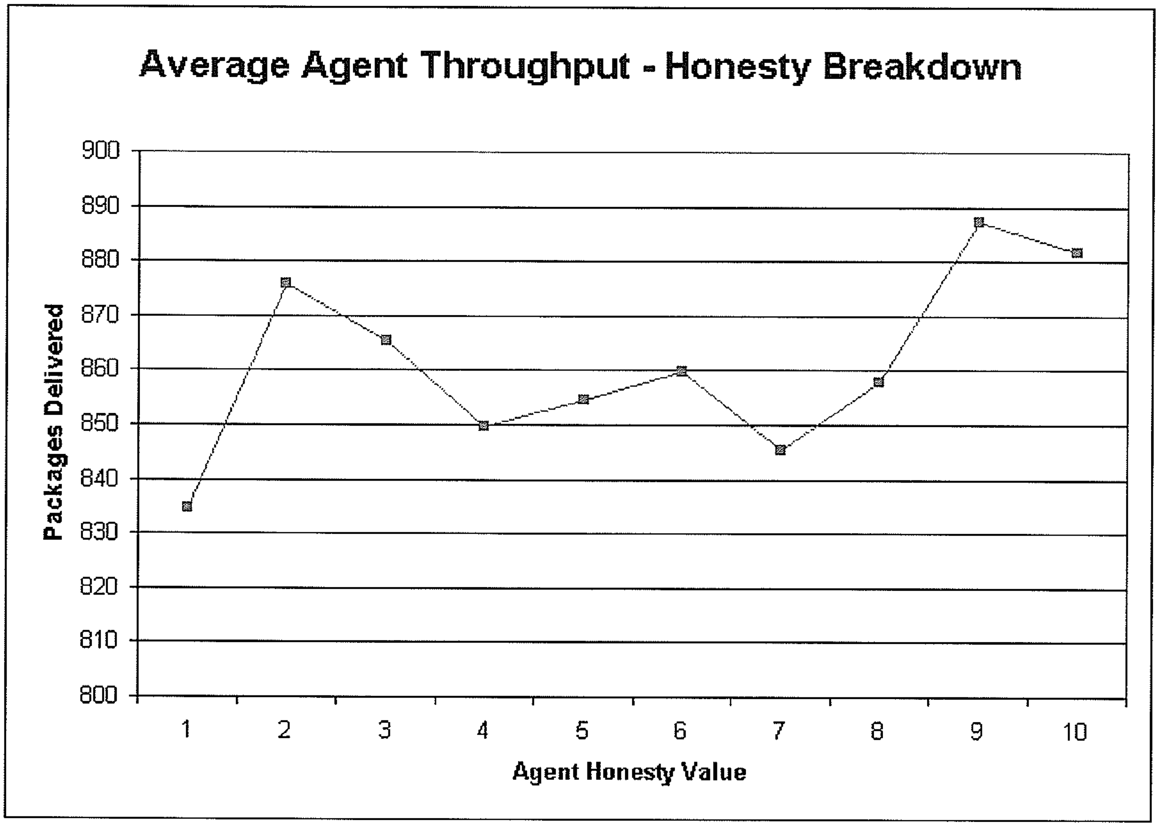

Figure 4.4: Average vandeVijsel Throughput by Honesty Value after 50000 Time Cycles

Table 4.2: Average Packages Lost by Memory Value

Memory Value	Lost Packages
1	30.63
2	28.80
3	24.48
4	21.76
5	16.86
6	13.91
7	10.05
8	6.98
9	3.65
10	0.00

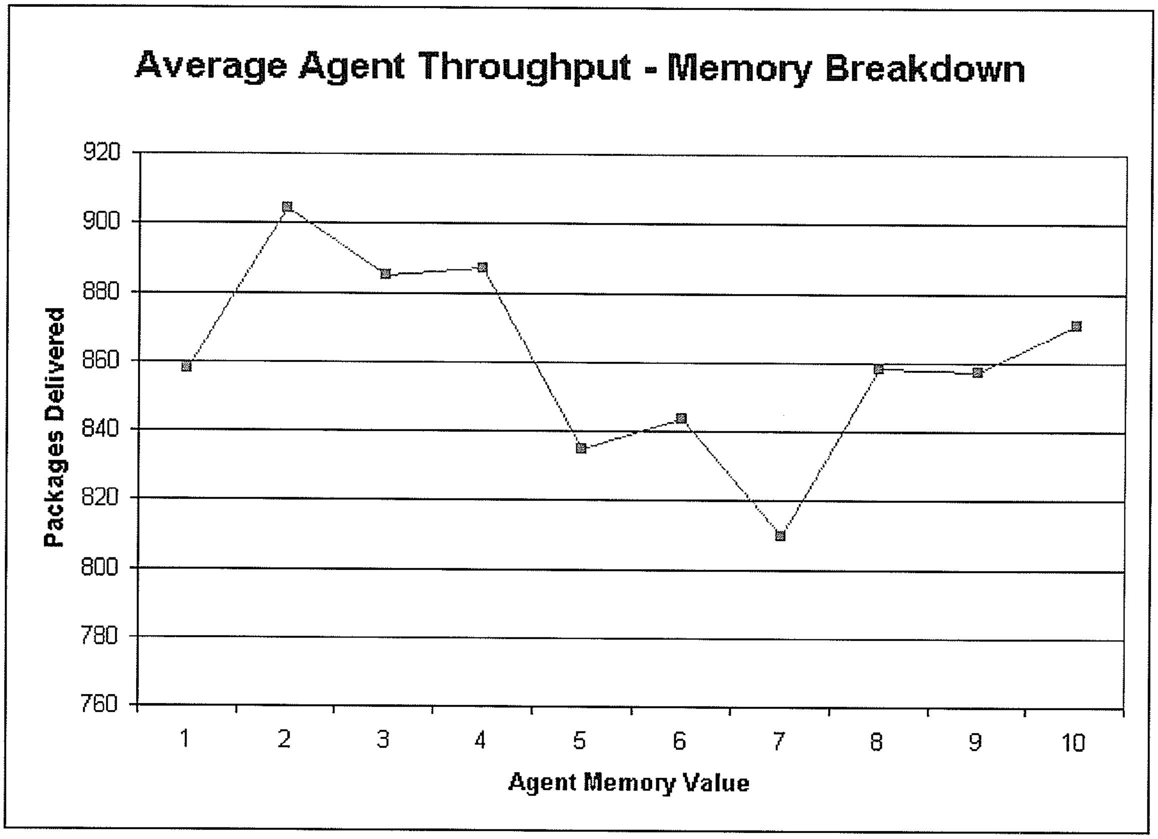

Figure 4.5: Average vandeVijsel Throughput by Memory Value after 50000 Time Cycles

ory and throughput. Memory does have some impact on throughput (as shown in Table 4.2) – a low memory score translates into a higher number of packages lost, and only packages that are successfully delivered are included in the throughput totals. However, since the chance of losing a package has been set to quite a low value, the number of lost packages is not enough to cause a significant impact to the throughput of an agent. Agents average only 15.71 lost packages per trial. Having a larger chance of forgetting a package would change this relationship, as more lost packages would translate into lower throughput.

Figures 4.3, 4.4 and 4.5 show that there are no direct correlations between through-

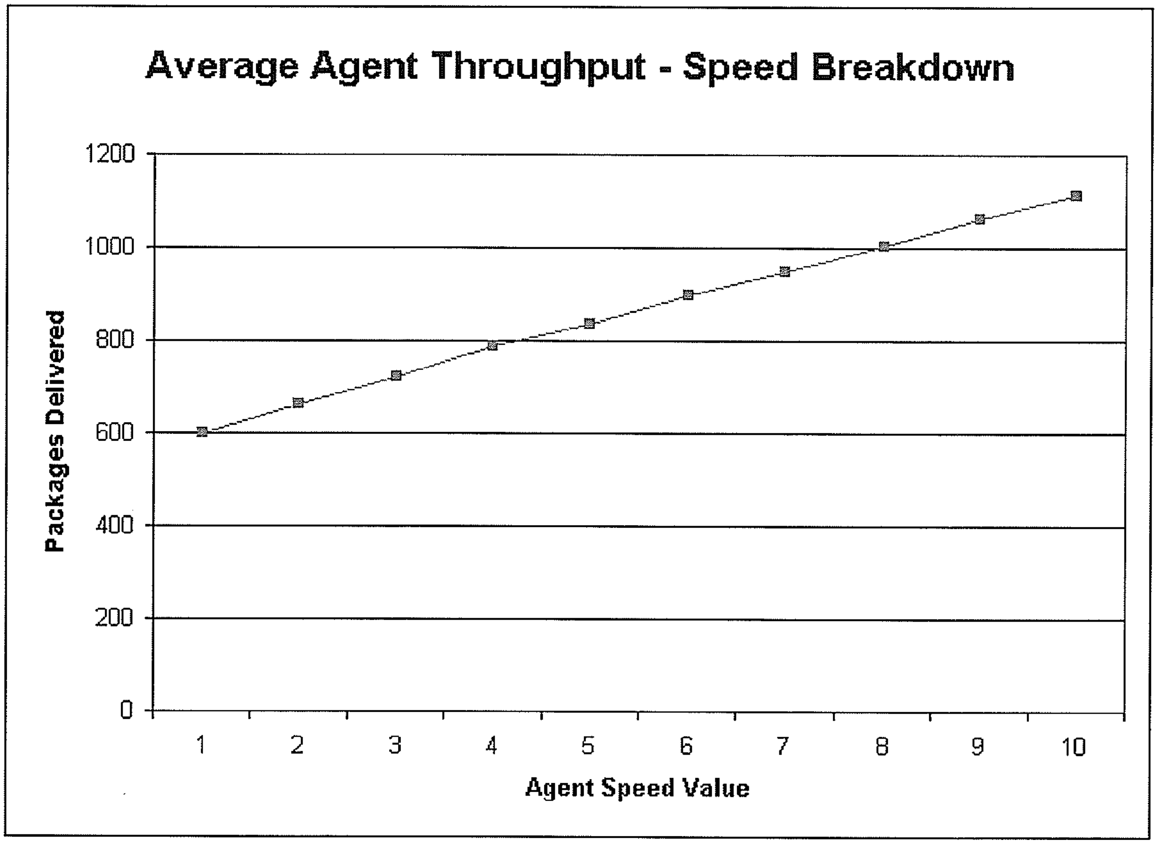

Figure 4.6: Average vandeVijsel Throughput by Speed Value after 50000 Time Cycles

put and trust, honesty or memory. Speed, however, has an obvious (and intuitive) impact, as seen in Figure 4.6. This graph provides a measure of the average throughput for all agents in the system, broken down by agent speed attribute values. The graph shows an obvious linear relationship – the slower an agent is, the fewer packages it can deliver. The faster the agent, the greater the throughput. Again, intuitively this makes sense – if one agent can cover ground at a greater rate than another, it will deliver more packages.

Determining the impact of speed on throughput requires some discussion on the nature of speed in each of these models. As discussed in Section 3.2.1, a vandeVijsel agent uses its speed value to determine its rate of movement on the grid. An agent will always be allowed to move 40% of the time, and will not be allowed to move 20% of the time. The other 40% of the time, the agent generates a random number between 1 and 10, and if the agent's speed attribute value is greater than or equal to this random number, then the agent gets to move. So an agent with a speed attribute of 1 will have a 40% chance of moving (guaranteed) plus 1/10 of the 40% chance that is determined by the speed attribute, for a total chance of moving of 44%. An agent with a speed attribute of 10, on the other hand, will have a 40% chance of moving (guaranteed) plus the full 40% determined by the speed attribute, for a total chance of moving of 80%.

The vandeVijsel agent, therefore, has a worst-case likelihood of moving on a given time cycle of 44%, and a best-case likelihood of moving on a given time cycle of 80%, and these percentages are derived from its speed attribute. Over a large number of time cycles, this likelihood of movement effectively becomes the movement rate of the agent, since an agent can only move one grid location per time cycle. An agent with a speed value of 10 has a movement rate of 80% over a large number of time cycles. Over a specific length of time, this agent will move approximately 20% farther than an agent with a speed value of 5, which has a movement rate of 60%. This is because at every time cycle, the agent with a speed of 10 has a 20% greater chance of being able to move one grid location than the agent with a speed of 5.

The Dutta/Sen agent also has its movement restricted, using its expertise quadrant. As discussed in Section 3.3.1, a Dutta/Sen agent moves 50% of the time (i.e. every even-numbered time cycle) when it is outside its expertise quadrant, and 100%

of the time when it is inside its expertise quadrant. Because of this spatial variability, a Dutta/Sen agent's movement rate is not constant – it depends on the grid location of the agent. However, determining the proportion of time an agent spends inside its expertise quadrant will allow the effective movement rate to be determined.

In order to determine this effective movement rate for a Dutta/Sen agent, I ran a separate experiment. This experiment consisted of four trials, and was run to gather the number of time cycles that a Dutta/Sen agent spends inside its expertise quadrant, and the number of time cycles it spends outside its expertise quadrant.

These additional trials indicate that, out of 50000 time cycles, the Dutta/Sen agent spends an average of 22402.18 time cycles (or 45%) within its expertise quadrant, and an average of 27597.82 time cycles (or 55%) outside of its expertise quadrant. While the proportion of time spent in the agent's expertise quadrant seems low, it is important to note that an agent outside of its expertise quadrant moves twice as slowly, and so it only makes a movement decision on half of the time cycles it spends in this quadrant. Thus, the agent is making a movement decision 22402 times in its expertise quadrant, and only 13798 times outside of its expertise quadrant, which are reasonable numbers with respect to expected proportions.

In addition, a Dutta/Sen agent that must venture outside of its expertise quadrant (because it no longer has any packages to deliver inside its expertise quadrant) must deliver at least one package outside of its expertise quadrant. Agents choose a package for delivery based on estimated payoffs, and then deliver that package to its destination. They do not choose another destination until either they receive a new package for delivery, or they deliver their package. When travelling towards a destination outside of their expertise, it is unlikely that they will take on packages from other agents. Recall from Section 3.3.1 that an agent only renders aid if the assisted package's destination is within 10 units of their current destination. Since packages that are to be delivered within 10 units of a location outside of their expertise are likely to also be outside of their expertise, it is unlikely that a cooperation possibility will exist in this scenario. It is more likely the case that the agent will accept additional packages on its way back to its expertise quadrant. Thus, the ratio of 45% inside its expertise quadrant to 55% outside its expertise quadrant is not unexpected.

Since the movement rate of a Dutta/Sen agent is 100% inside its expertise quadrant (where the additional trials indicate it spends approximately 45% of its time) and 50% outside of its expertise quadrant (where the trials indicate it spends approximately 55% of its time), this translates to an overall effective movement rate of 72.5%.

It is interesting to examine the change in the effective movement rate of a Dutta/Sen agent over time. Figure 4.7 shows the effective movement rate for each 1000-time cycle interval for the Dutta/Sen agents (averaged over the 4 trials). This is not a cumulative total, but rather the rate that the agent experienced in that 1000-time cycle interval. The graph shows a low starting point, followed by agents learning relatively quickly that they should attempt to use their own quadrant as much as possible, and some small fluctuations over time after that.

VandeVijsel agents have a movement rate of between 44% and 80%, depending on their speed attribute. The rate climbs linearly, based on the speed attribute, between 44% and 80%, with each additional unit of speed accounting for an additional 4%

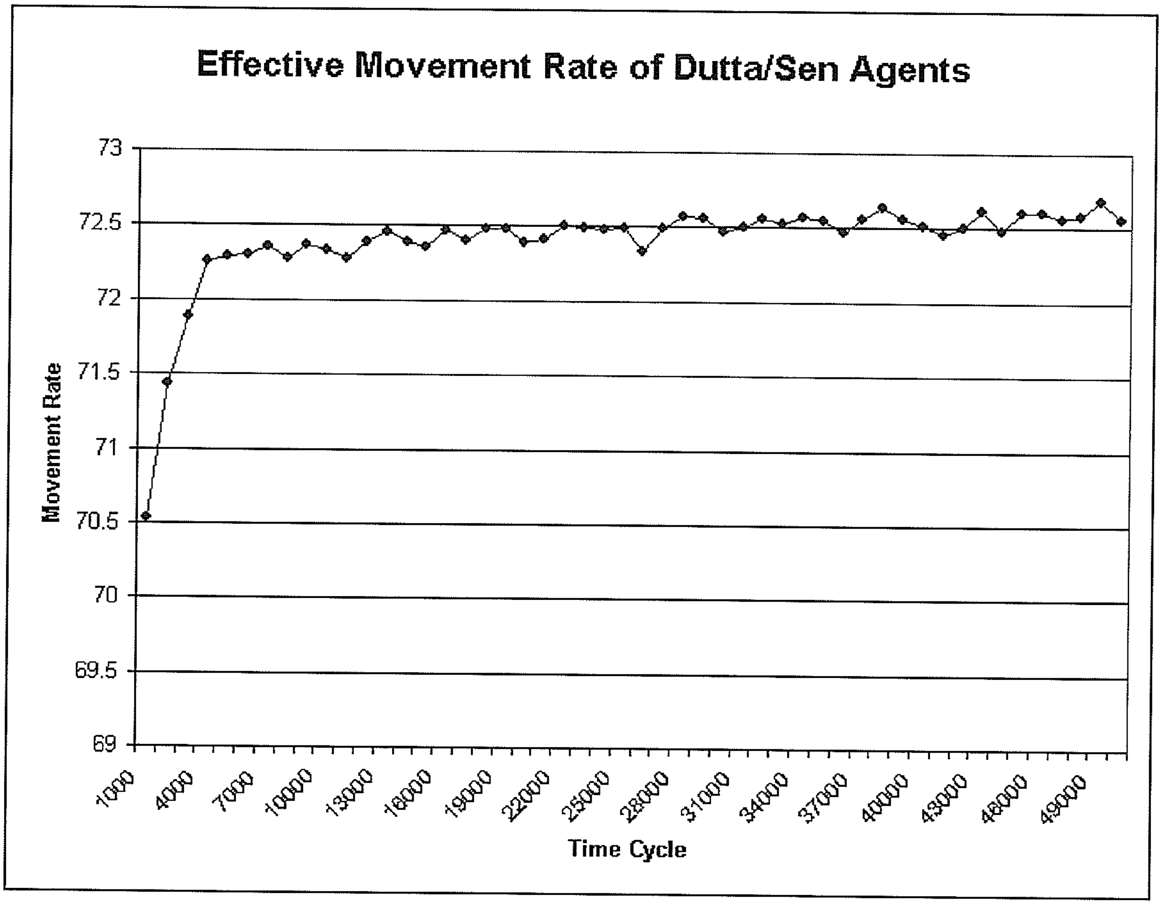

Figure 4.7: Effective movement rates for 1000-time cycle intervals for Dutta/Sen agents

rate of movement. If the Dutta/Sen agent moved as a vandeVijsel agent did, then 40% of this 72.5% effective movement rate would be guaranteed, and the remainder would be determined by its speed attribute. By removing the guaranteed 40% from the effective movement rate, we are left with 32.5%. Since each unit of speed is worth 4% movement rate, dividing 32.5% by 4% provides the effective speed value for the Dutta/Sen agent of 8.125. In other words, a speed attribute value of 8.125 in a vandeVijsel agent would result in the same effective movement rate that we observe experimentally for a Dutta/Sen agent. Therefore, if the coalition formation

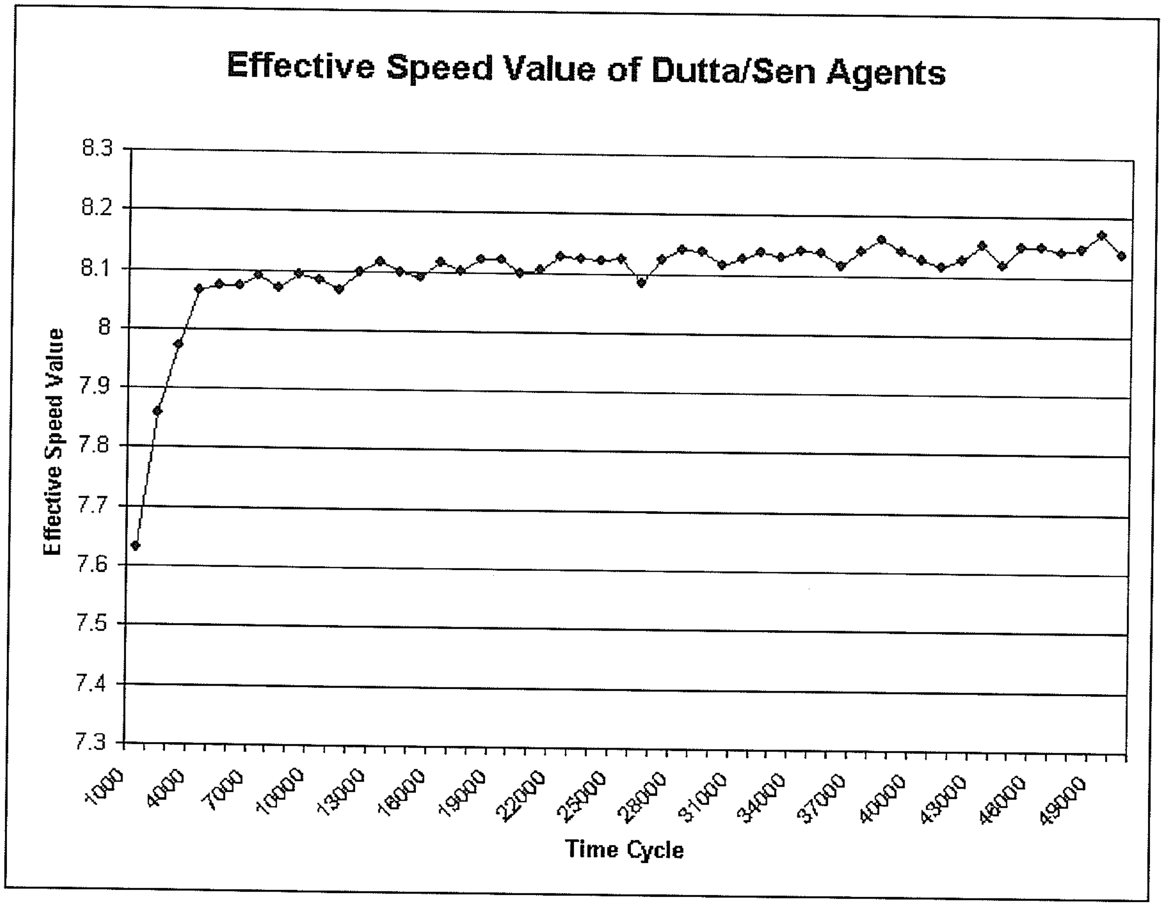

Figure 4.8: Effective speed values for 1000-time cycle intervals for Dutta/Sen agents

approaches were working with identical efficacy (i.e. if the effect of the coalition formation approach were to be identical in both agent models), then I would expect the Dutta/Sen agents to display the same throughput as a vandeVijsel agent with a speed attribute of 8.125[2].

Figure 4.9 shows that this is clearly not the case. In fact, the Dutta/Sen agents display similar average throughput to a vandeVijsel agent with a speed attribute

[2]Note that this is the average value over the entire length of the experiment. Figure 4.8 shows the effective speed values of the Dutta/Sen agents in 1000-time cycle intervals, showing a similar pattern to the movement rate illustrated in Figure 4.7.

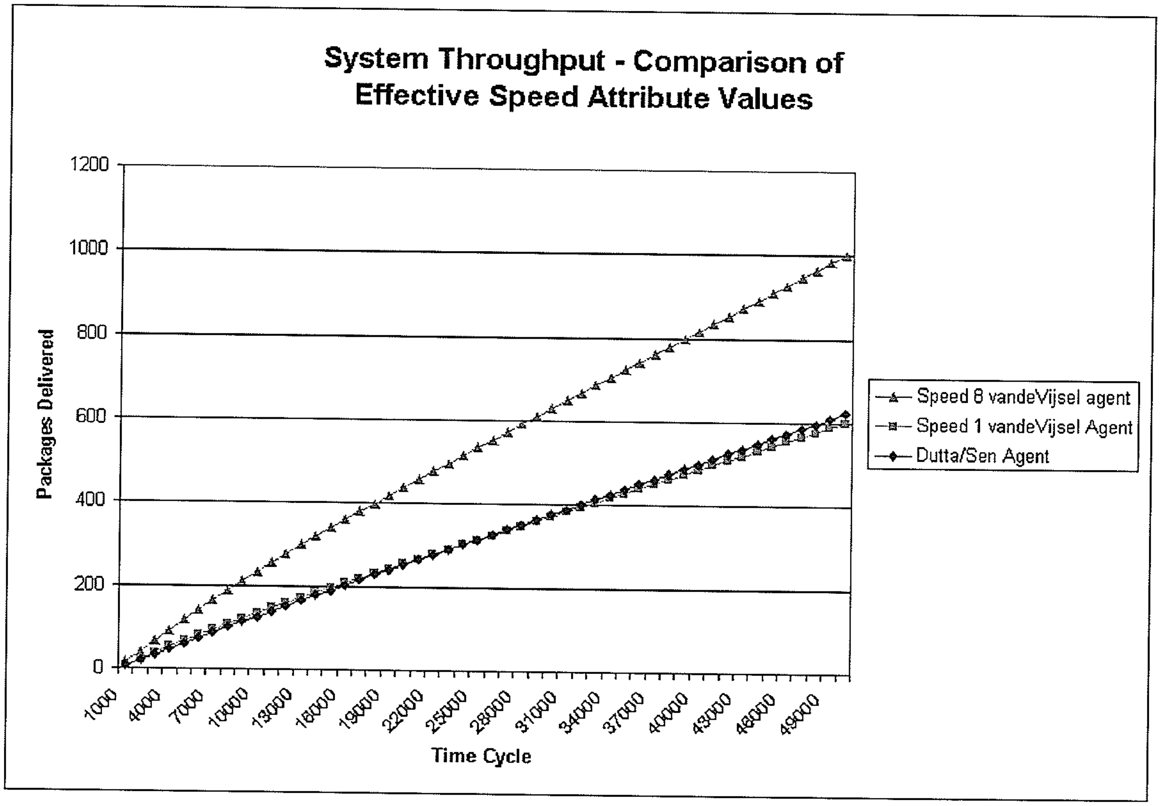

Figure 4.9: Comparison of actual throughputs for those vandeVijsel agents with Speed values 1 and 8, and all Dutta/Sen agents

value of one, despite their effective movement rate that translates to a value of 8.125. Therefore, the coalition formation approach used by the vandeVijsel agent (which is the only remaining variable that has not yet been accounted for) allows a vandeVijsel agent with a speed attribute of one (the worst case for a vandeVijsel agent) to perform as well as a Dutta/Sen agent with an effective speed attribute of 8.125. This also puts the results shown in Figure 4.1 in perspective. Not only are the vandeVijsel agents attaining higher throughput, but they are doing so while moving more slowly, on average, than the Dutta/Sen agents.

In summary, I have shown that the only contributing factors to the difference in throughput between the two agent models are the coalition formation approaches and the difference in movement rates for the two agents. By determining the effective movement rate of the Dutta/Sen agents, I have shown that the coalition formation approach used by the vandeVijsel agent model is giving vandeVijsel agents with an effective movement rate of 44% the same throughput as a Dutta/Sen agent with an effective movement rate of 72.5%.

4.5.2 Coalition Stability

As mentioned in Section 4.2, the vandeVijsel and Dutta/Sen agent models have different concepts of coalitions. The vandeVijsel model uses explicit, closely-knit coalitions, and a single agent can be in many different coalitions. The Dutta/Sen agent uses opinions to define loose groups of agents that present a larger likelihood of cooperation. As indicated in Section 4.2, for the purposes of this analysis I will consider each Dutta/Sen agent to define its own coalition, and that coalition will consist of other agents for whom the agent currently maintains a positive opinion.

Section 4.2 described two distinct measures of coalition stability – the total number of groups in the system, and the rate of change in the membership of those groups. When discussing the total number of groups, it is difficult to make a fair comparison. Initially, I attempted to define a Dutta/Sen coalition as a pair of agents that share a positive opinion about each other. This approach provides an unfair comparison, however, because this definition restricts a Dutta/Sen coalition to two members (i.e. a partnership), while the vandeVijsel approach is still considering much larger groups. Adjusting the vandeVijsel approach to consider partnerships is also unfair, because the number of effective partnerships increases combinatorially when a single agent is

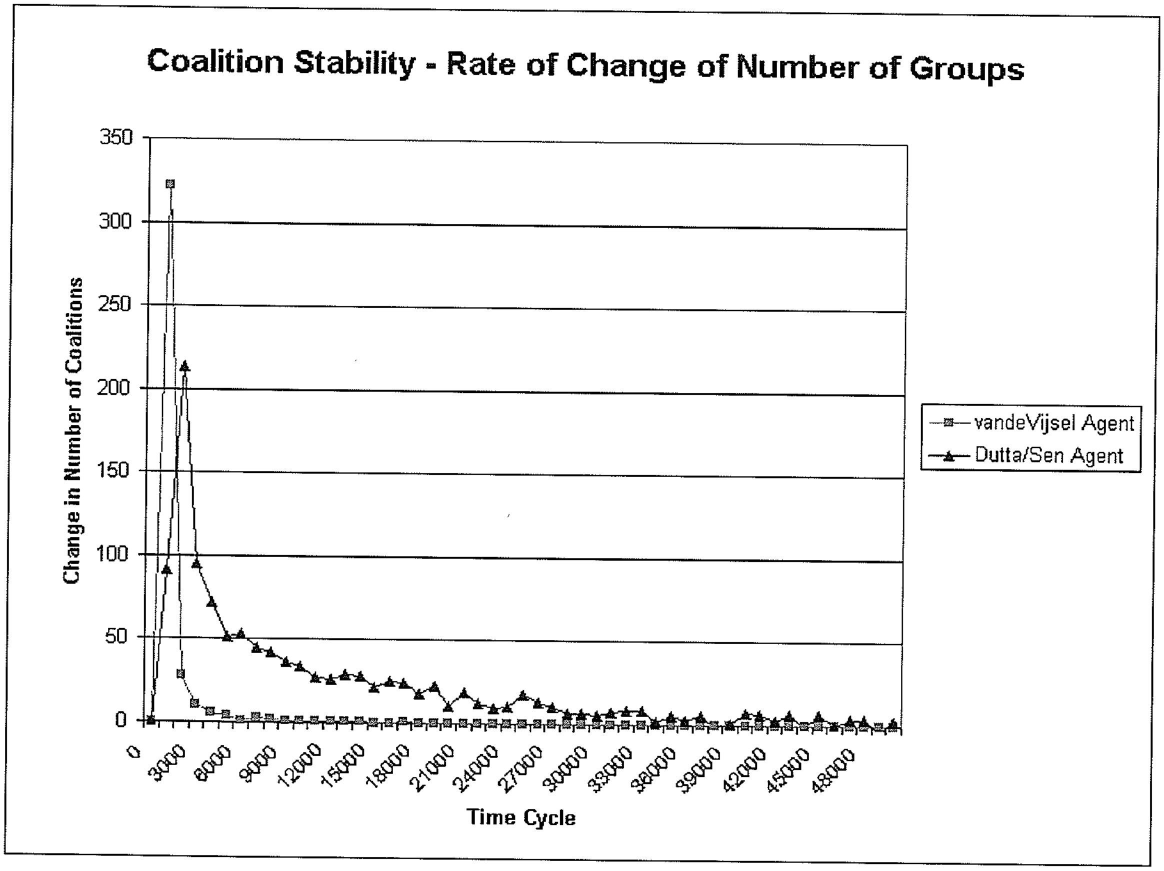

Figure 4.10: Rate of change of the aggregate number of groups for the vandeVijsel and Dutta/Sen agents

added to a coalition.

Additionally, if we use the definition of a Dutta/Sen coalition as outlined in Section 4.2, then each agent maintains one coalition of agents of which it has a positive opinion. This definition gives us a constant number of Dutta/Sen groups in the system, equal to the number of agents, which is not useful for comparative purposes.

Instead of examining the actual number of groups in the system, another approach that was considered was examining the rate of change of the number of groups. Figure 4.10 shows this rate of change, considering a Dutta/Sen coalition to be a pair of agents that share a positive opinion about each other. (Using the definition from

Section 4.2 would result in a rate of change of zero for the Dutta/Sen agent, since the number of groups would always equal the number of agents.) However, every time the Dutta/Sen agent receives help from a new agent, it may form a new partnership, and as agents learn about each other's abilities, more and more positive partnerships will be created. On the other hand, in the vandeVijsel approach, new groups are rarely required between agents after a certain amount of time, since it is highly likely that if two agents do not share a coalition after some length of time, one agent or the other will have knowledge about a group for which both agents will be suitable. Thus this comparison does not treat the Dutta/Sen approach fairly.

Because of these issues, I have chosen to focus my analysis of coalition stability on the rate of change of group membership in the system. Section 4.2 outlines the method used for calculating the rate of coalition membership change for the two approaches.

There is one additional barrier to making a reasonable comparison between these two approaches. The potential dishonesty of vandeVijsel agents will likely cause additional coalition membership changes, as coalitions learn about the dishonest agent's true abilities (see Section 4.2). However, the dishonest vandeVijsel agent can join multiple coalitions at the same time, presumably causing instability in each coalition. In the Dutta/Sen approach, interactions between two agents are localized to only those two agents, and no other agents' opinions are affected. When a Dutta/Sen agent determines that another agent is not useful to it, its opinion of the other agent is reduced, and a single coalition membership change results. So a single vandeVijsel agent will result in numerous coalition membership changes, simply by the nature

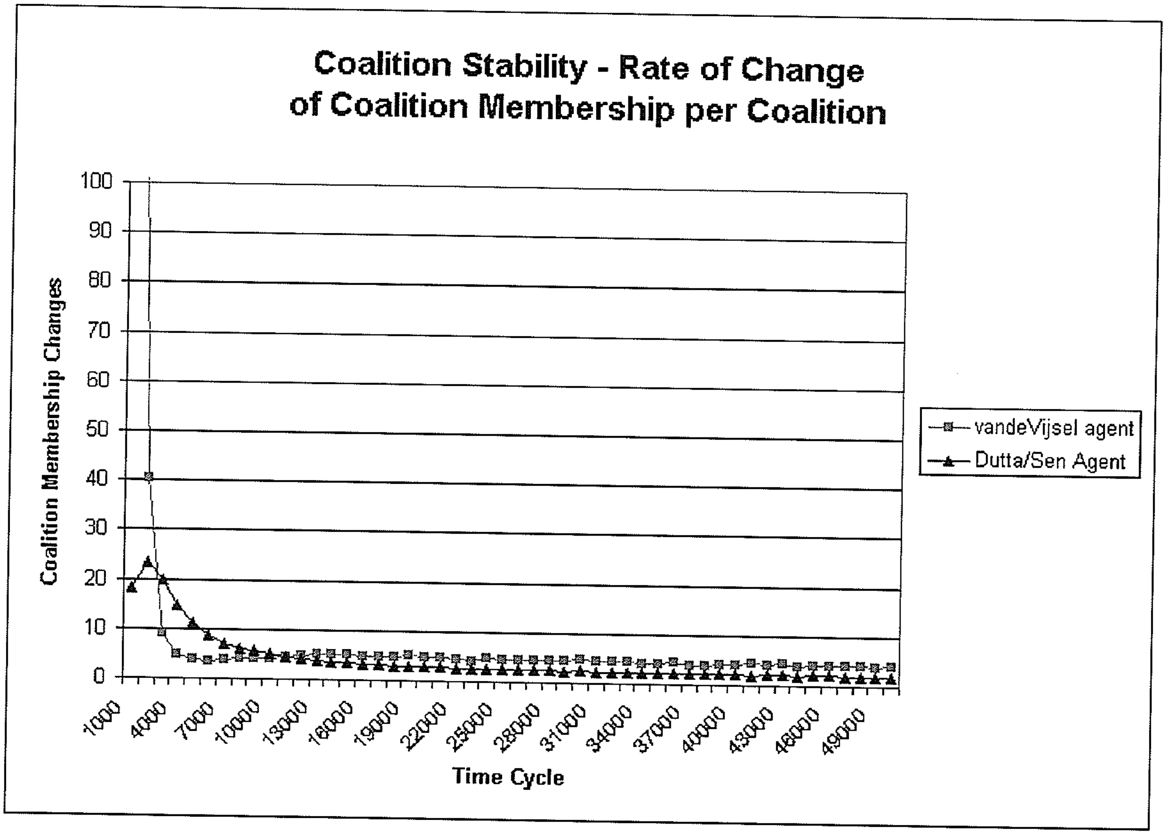

Figure 4.11: Rate of change of coalition membership by time cycle for Dutta/Sen and vandeVijsel agents

of the vandeVijsel agent model because of one of the realistic factors it incorporates which the Dutta/Sen approach does not.

To allow for accurate comparisons between the stability of the two approaches, therefore, I have calculated the coalition membership changes divided by the number of coalitions in which the agent currently participates. For the Dutta/Sen approach, the number of coalitions in which the agent participates is one, by definition – every agent maintains its own coalition. For the vandeVijsel approach, the number of coalitions varies from agent to agent.

Figure 4.11 shows the rate of change in coalition membership per coalition (as defined above) for both the Dutta/Sen and vandeVijsel approaches. This graph represents the average number of coalition changes for a single coalition over time. Both the vandeVijsel and Dutta/Sen approaches reach a stable state relatively quickly, where there are generally a fixed number of changes every 1000 time cycles.

The vandeVijsel agent model exhibits a slightly higher rate of change in coalition membership due to the point raised above – it must contend with dishonest and fallible agents in its coalitions. In the Dutta/Sen approach, there is no such dishonesty – the observations of the agent are all it uses to determine usefulness, and so it makes sense that there are fewer membership changes over time. However, both approaches do reach a reasonable level of stability within approximately 10000 time cycles, and do not waver significantly from that level for the remainder of the simulation.

4.5.3 Additional Results

In addition to the data captured on system throughput and coalition stability, I have analyzed two more aspects of the behaviour of the vandeVijsel and Dutta/Sen agents.

First, I have examined the number of packages, on average, that agents are carrying at any given time. These results are shown in Figure 4.12. As discussed in Section 3.2, agents are only able to carry a maximum of 10 packages at a time. Figure 4.12 shows that, after only a few thousand time cycles, agents are carrying close to their maximum capacity of packages most of the time. Thus, every time they choose to deliver a certain package from their payload, they must select one out of 9 or 10 possibilities. Since an agent is often carrying its maximum payload of packages, its

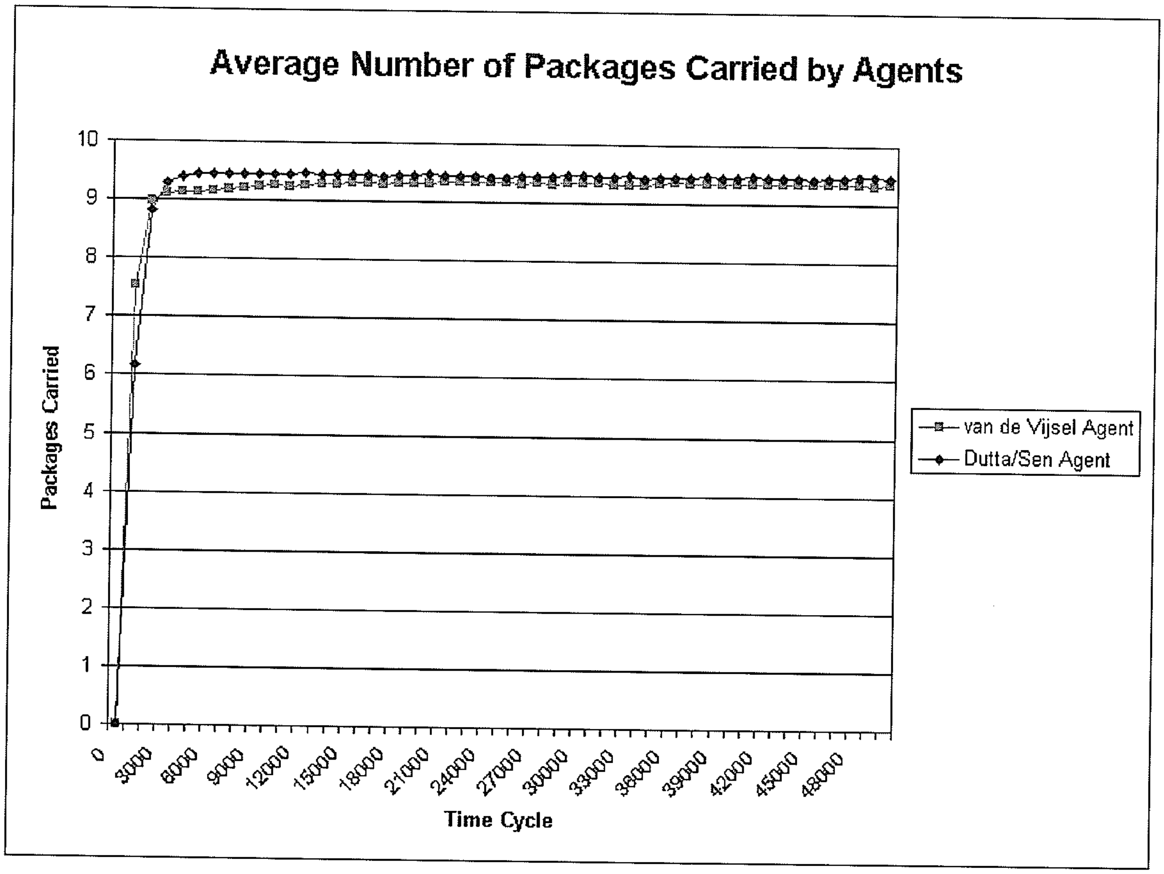

Figure 4.12: Average number of packages being carried by vandeVijsel and Dutta/Sen agents, over time

ability to help other agents deliver packages is reduced as it will likely not have the capacity to take on another package from another agent.

I have also examined the number of vandeVijsel coalitions created, on average. These results can be found in Figure 4.13. Since the agents do not have the ability to merge coalitions, there are a large number of coalitions formed in the initial phases of the simulation. Coalitions between agents of average ability will grow in size, while coalitions between "fringe" agents (agents of unusually high or low abilities) will tend to remain small.

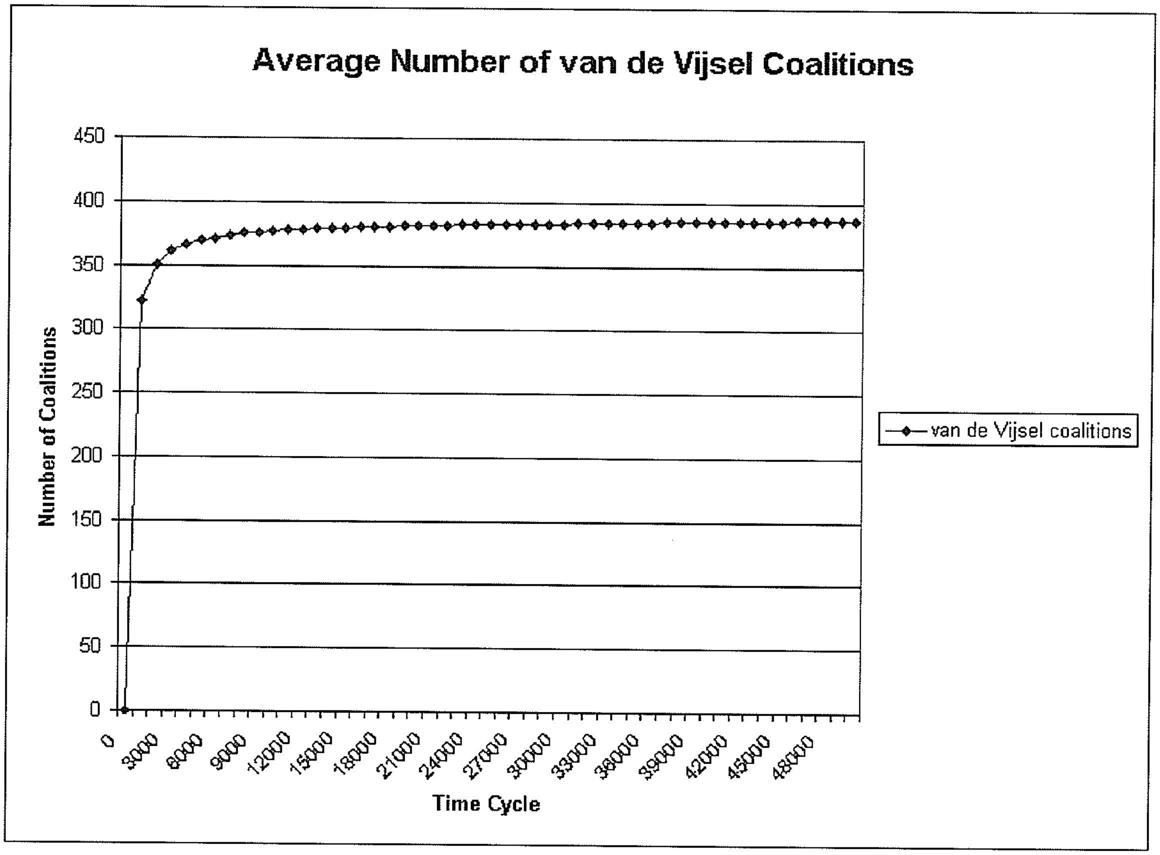

Figure 4.13: Average number of vandeVijsel coalitions created, over time

4.6 Summary

This chapter has outlined that the vandeVijsel approach provides significant gains in throughput for the package delivery domain, while resulting in comparably stable coalitions when evaluated against the baseline approach of Dutta and Sen. The following chapter will summarize this research and outline future work.

Chapter 5

Conclusion

In this thesis, I have presented a new coalition formation approach that avoids many of the restrictive assumptions that prevent other coalition formation approaches from being applicable to more realistic problems. The agent model I have described encompasses many factors that increase the realism of the approach:

- Agents are heterogeneous in their abilities
- Agents can have multiple, potentially conflicting goals
- Agents can be part of multiple coalitions at the same time
- Coalitions learn about the actual abilities of their members over time, exposing dishonest agents that have exaggerated their capabilities and removing them from the group

Based on a review of existing coalition formation research, I have identified the approach that is the most realistic – that is, the approach that avoids more of this restrictive assumptions than other approaches. I have implemented both my approach and this baseline approach in a package delivery domain, and evaluated my approach against the baseline using the measures of *system throughput* and *coalition stability.*

5.1 Findings and Analysis

System throughput was 38% higher in my approach as compared to the Dutta/Sen approach. However, there were two differences in the approaches that could have given rise to this difference – the difference in agent ability and the coalition formation approach itself.

I was able to determine that the expertise quadrant in the baseline approach had no effect on the throughput of the agent – all agents displayed comparable throughput regardless of their expertise quadrant. Similarly, I found no correlation between the throughput of a vandeVijsel agent and the agent's trust, memory or honesty values. However, there was a definite correlation between the speed attribute of a vandeVijsel agent and its throughput. Examining how much time a Dutta/Sen agent spent in and out of its area of expertise allowed the calculation of the effective speed attribute of a Dutta/Sen agent, allowing a direct speed comparison to my agents. This comparison showed that Dutta/Sen agents were effectively faster than mine. The improvement in throughput shown by my approach is even more positive in this light.

There were two different measures that could be taken to compare the coalition stability of the two approaches – the rate of change in the number of coalitions in the system, and the rate of change in the membership of those groups. I concentrated on examining stability from the standpoint of the rate of membership change, because this factor could be examined more effectively despite the differences in the nature of a coalition between these two approaches. In examining the rate of change in a Dutta/Sen coalition and comparing it to the vandeVijsel approach, adjusting for the number of coalitions, the rate of membership change in the two approaches was shown to be similar.

Overall, my approach showed greater productivity as a result of forming coalitions, as shown by throughput, while still producing stable coalitions.

5.2 Future Work

The goal of this thesis was to define and evaluate a coalition formation approach that encompassed a number of important characteristics of the real world. While this work has encompassed many such characteristics, there are other enhancements that could still be made to increase the realism of the approach.

For example, the system currently works with a fixed number of agents. In a real-world scenario, this would not be the case – there would always be new agents being added to any environment, and there would also be agents leaving an environment. Adding this element to the described approach would be an interesting line of future research. Since the agents currently make no assumptions about the total number of agents in the system, the addition of new agents should not affect the results significantly – an individual agent has no way of knowing whether another agent it has just met is new to the system or not. The departure of agents, however, would be a larger issue to handle, especially if the agents were to leave the system without notification. Coalitions of which the agent was a member might assume it is still active, even though it is no longer contributing, and additional maintenance to the groups (such as removing agents that have not assisted on any coalition packages over a certain time frame) might be required. There may also be new schemes for deception that could be perpetrated if agents could depart without notice as well.

Another significant topic of future research would be the potential variability of agent attribute values. Currently, these values are fixed for the life of the agent. In a real-world environment, these could be variable. A person could increase their speed by working out, losing weight, or purchasing a faster vehicle. A person's memory could improve by using some sort of recording system to counteract the effect of forgetting things. Attributes that are reflective of a person's personality, such as trust and honesty, could be changed simply by making a decision to attempt to be more honest or trustworthy. Conversely, all these items could be decreased using analogous scenarios.

The effects of such variability in attribute values would be interesting. Changes in the ability-based attributes of speed and memory would not only result in changes in the agent's throughput, but also their value to any coalitions of which they are members. Coalitions may decide to re-evaluate members periodically to see if their attributes have changed, changing their suitability for the coalition. There are many different factors that could be explored if attribute variability is introduced into the agent model.

Another item that was not introduced in this research is the cost of participating in a coalition. Currently, an agent can join a coalition and remain in the coalition for an indefinite period of time without incurring an explicit cost. There are costs such as the responsibilities of being asked to help fellow members, but there are no explicit costs such as membership fees. An agent is also removed from the coalition without penalty (other than the implicit penalty of losing the potential aid of the coalition members). Evaluating the effects of such costs on agent performance would provide interesting results.

There has also been no exploration of the financial aspects of the package delivery domain. Agents receive payoffs for delivering packages, and receive penalties for losing them. Agents with higher attribute values should have more accumulated currency than agents with low values. An analysis of effects of agent attributes on the accumulated payoffs of agents would give further insight into the effectiveness of the approach. Since we have gathered data on financial aspects, immediate future research includes analyzing this data for interesting phenomena and designing future experiments based on the results of this.

Another interesting path of future research would be the possibility of allowing coalition mergers, almost like an additional level of coalition formation between coalitions. The model allows for multiple coalitions to emerge that have similar values – that is, the average attribute values of its members are similar. It would be useful to explore how those coalitions could merge into a single group, and if this would increase the performance of agents in the system. There are many facets of this issue to explore – how would agents identify a merger possibility? Would a vote or approval from the member agents be required? Would agents be able to opt out of joining the merged coalition? This scenario can be compared to a corporate merger in the real world, and many of the issues in such a scenario could be explored from a coalition perspective.

The applicability of this approach would also be enhanced by implementation in several other realistic domains. While the coalition formation approach discussed in this thesis works well for the package delivery domain, and it encompasses many different aspects of realistic scenarios, additional research is required to ensure that the approach will work equally well when applied to other domains.

Along the same lines, it would be interesting to see the creation of a physical domain with agents that have an actual physical presence (i.e. robots) and that employ this coalition formation approach. Currently, the Coalition Formation Simulator provides a small amount of infrastructure that allows the agents to access the information they require – for example, the simulator provides a list of encountered agents at every time cycle. Implementation into a physical domain would pose challenges, but could provide interesting results that would further serve to illustrate the approach's potential in realistic scenarios.

5.3 Summary

In this thesis, I have made the argument that existing coalition formation approaches do not adequately handle many of the issues that occur in more realistic scenarios. Coalition formation research is still in its infancy, as evidenced by the fact that there is no strong agreement in the field on any one particular approach. As multi-agent systems become more and more prevalent in real-world applications, flexible and computationally tractable coalition formation approaches will be required to handle the inevitable demands that agents work together to achieve their goals.

The approach described in this thesis makes strides in this direction, by providing an agent model with heterogenous abilities, multiple goals and the ability to form multiple groups, while learning about the abilities of others and attempting to determine the degree to which agents have been dishonest.

However, there still remains much work to be done to provide a generalized approach that will be applicable to a large number of domains. I have outlined several directions for future research that will make this approach more applicable to other areas. With the incredible popularity of distributed computing via the Internet, end users will begin to use agents as their representatives more often, and the public demand for approaches that allow agents to work together to achieve their goals while protecting their own interests will grow. I believe that coalition formation research will prove to be one of the fastest-growing areas of artificial intelligence research over the coming years, and my hope is that this thesis, and the future work it may inspire, can aid in the realization of a generalized approach applicable to any realistic domain.

Bibliography

Sherief Abdallah and Victor Lesser. Organization-based coalition formation. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 3, pages 1296–1297, March 2004.

John Anderson, Brian Tanner, and Jacky Baltes. Dynamic coalition formation in robotic soccer. In *Proceedings of the AAAI-04 Workshop on Forming and Maintaining Coalitions and Teams in Adaptive Multiagent Systems*, San Jose, CA, July 2004.

John Anderson, Brian Tanner, and Ryan Wegner. Peer reinforcement in homogeneous and heterogeneous multi-agent learning. In *Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing (ASC2002)*, July 2002.

Frederick Asselin and Brahim Chaib-draa. Coalition formation with non-transferable payoff for group buying. In *The Second International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, pages 922–923, July 2003.

Robert L. Axtell. Non-cooperative dynamics of multi-agent teams. In *Proceedings* *of the first international joint conference on Autonomous agents and multiagent systems.* ACM Press, 2002.

Tucker Balch. The impact of diversity on performance in multi-robot foraging. In *Proceedings of the Third Annual Conference on Autonomous Agents*, pages 92–99, April 1999.

Suryapratim Banerjee, Hideo Konishi, and Tayfun Sonmez. Core in a simple coalition formation game. Technical Report 449, Boston College Department of Economics, December 1999.

Silvia Breban and Julita Vassileva. Long-term coalitions for the electronic marketplace. In B. Spencer, editor, *Proceedings of the E-commerce Application Workshop*, June 2001.

Silvia Breban and Julita Vassileva. Using inter-agent trust relationships for efficient coalition formation. In R. Cohen and B. Spencer, editors, *Proceedings of the 13th Canadian Conference on AI*, May 2002.

Christopher H. Brooks and Edmund H. Durfee. Congregating and market formation. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1–100, July 2002.

Christopher H. Brooks and Edmund H. Durfee. Congregation formation in multiagent systems. *Journal of Autonomous Agents and Multi-agent Systems*, 7(1–2):145–170, 2003.

Christopher H. Brooks, Edmund H. Durfee, and Aaron Armstrong. An introduction to congregating in multiagent systems. In *Proceedings of the Fourth International Conference on Multiagent Systems (ICMAS-2000)*, pages 79–86, July 2000.

Philippe Caillou, Samir Aknine, and Suzanne Pinson. A multi-agent method for forming and dynamic restructuring of pareto optimal coalitions. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 1074–1081, 2002.

Georgios Chalkiadakis and Craig Boutilier. Bayesian reinforcement learning for coalition formation under uncertainty. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 3, pages 1090–1097, 2004.

Javier Contreras and Felix F. Wu. Coalition formation in transmission expansion planning. *IEEE Transactions on Power Systems*, 14(3):1144–1152, August 1999.

T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms.* MIT Press, 1990.

David Cornforth, Michael Kirley, and Terry Bossomaier. Agent heterogeneity and coalition formation: Investigating market-based cooperative problem solving. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 2, pages 556–563, February 2004.

Viet Dung Dang and Nicholas R. Jennings. Generating coalition structures with finite bound from the optimal guarantees. In *Third International Joint Conference on*

Autonomous Agents and Multiagent System (AAMAS), volume 2, pages 564–571, February 2004.

Partha Sarathi Dutta and Sandip Sen. Forming stable partnerships. *Cognitive Systems Research*, 4(3):211–221, 2003.

Miguel Schneider Fontan and Maja J. Mataric. A study of territoriality: The role of critical mass in adaptive task division. In Pattie Maes, Maja Mataric, Jean-Arcady Meyer, Jordan Pollack, and Stewart W. Wilson, editors, *Proceedings, From Animals to Animats IV, Fourth International Conference on Simulation of Adaptive Behaviour*, pages 553–561. MIT Press/Bradford Books, 1996.

M.R. Garey and D.S. Johnson. *Computers and Intractability: a Guide to the Theory of NP-Completeness.* W.H. Freedman and Co., New York, 1979.

Steven Ketchpel. Forming coalitions in the face of uncertain rewards. In *National Conference on Artificial Intelligence*, pages 414–419, 1994.

K. Lerman and O. Shehory. Coalition formation for large-scale electronic markets. In *Proceedings of the Fourth International Conference on Multiagent Systems (ICMAS-2000)*, pages 216–222, July 2000.

Rajiv T. Maheswaran and Tamer Basar. Coalition formation in proportionally fair divisible auctions. In *The Second International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, pages 25–32, July 2003.

Kaye Mason, Jorg Denzinger, and Sheelagh Carpendale. Negotiating gestalt: Artistic expression and coalition formation in multiagent systems. In *Third International*

Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), volume 3, pages 1350–1351, March 2004.

Sara McGrath, John Anderson, and Jacky Baltes. Improving cooperation in spatially distributed agents. In *Third International Conference on Computational Intelligence, Robotics and Autonomous Systems*. Singapore, 2005. (submitted).

Michal Pechoucek, Vladimir Marik, and Jaroslav Barta. A knowledge-based approach to coalition formation. *IEEE Intelligent Systems*, 17(3):17–25, 2002.

Michal Pechoucek, Vladimir Marik, and Olga Stepankova. Coalition formation in manufacturing multi-agent systems. In M. Pechoucek and V. Marik, editors, *Workshop on Industrial Applications of Holonic and Multi-Agent Systems, 11th International Conference on Database and Expert Systems Applications*, pages 241–246. IEEE Computer Society, September 2000.

A. Rapoport and R. Zwick. *Encyclopedia of Pyschology*, pages 424–426. Oxford University Press, New York, 2000.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

Tuomas Sandholm, Kate Larson, Martin Andersson, Onn Shehory, and Fernando Tohme. Anytime coalition structure generation with worst case guarantees. In *Fifteenth National Conference on Artificial Intelligence (AAAI)*, pages 46–53, July 1998.

Ted Scully, Michael G. Madden, and Gerard Lyons. Coalition calculation in a dynamic agent environment. In *Proceedings of the 21st International Conference on Machine Learning*, July 2004.

Sandip Sen and Partha Sarathi Dutta. Searching for optimal coalition structures. In *Fourth International Conference on MultiAgent Systems (ICMAS)*, pages 287–292, 2000.

Sandip Sen and Partha Sarathi Dutta. The evolution and stability of cooperative traits. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*. ACM Press, 2002.

Onn Shehory and S. Kraus. Coalition formation among autonomous agents: Strategies and complexity. In *From Reaction to Cognition*, number 957, pages 57–72, 1995.

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

Onn Shehory and Sarit Kraus. Feasible formation of coalitions among autonomous agents in non-super-additive environments. *Computational Intelligence*, 15(3), 1999.

Ron Sun and Chad Sessions. Multi-agent reinforcement learning with bidding for segmenting action sequences. In *From Animals to Animats: Proceedings of the International Conference of Simulation of Adaptive Behaviour (SAB)*. MIT Press, 2000.

R. Sutton and A. Barto. *Reinforcement Learning*. MIT Press, Cambridge, MA, 1998.

Fernando Tohme and Tuomas Sandholm. Coalition formation processes with belief revision among bounded-rational self-interested agents. *Journal of Logic and Computation*, 9:1–23, 1999.

Predrag Tosic and Gul Agha. Maximal clique based distributed group formation for autonomous agent coalitions. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2004.

Maksim Tsvetovat and Katia Sycara. Customer coalitions in the electronic marketplace. In *Proceedings of Autonomous Agents 2000*, pages 263–264, 2000.

Manuela Veloso and Peter Stone. Individual and collaborative behaviours in a team of homogeneous robotic soccer agents. In *Proceedings of the Third International Conference on Multiagent Systems*, pages 309–316, 1998.

Gerhard Weiss. *Multiagent Systems.* MIT Press, Cambridge, MA, 1999.

Junichi Yamamoto and Katia Sycara. A stable and efficient buyer coalition formation scheme for e-marketplaces. In *Proc. International Conference on Autonomous Agents*, May 2001.