APPROVAL SHEET

Title of Dissertation: Organizational Learning and Network Adaptation in Multi-Agent Systems

Name of Candidate: Matthew E. Gaston
Doctor of Philosophy, 2005

Dissertation and Abstract Approved: ______________________________
Dr. Marie desJardins
Assistant Professor
Department of Computer Science and Electrical Engineering

Date Approved: ______________________________

Name: Matthew E. Gaston

Permanent Address: 947 Deerberry Court
Odenton, MD 21113

Degree and date to be conferred: Ph.D., 2005.

Date of Birth: August 11, 1976.

Place of Birth: Harrisburg, PA.

Secondary education: Bishop McDevitt High School, Harrisburg, PA, 1994.

Collegiate institutions attended:

2002-2005	University of Maryland Baltimore County	Ph.D.	2005.
1999-2002	University of Maryland Baltimore County	M.S.	2002.
1994-1998	University of Notre Dame	B.S.	1998.

Major: Computer Science (Ph.D. and M.S.) and Mathematics (B.S.).

Professional publications:

Matthew Gaston and Marie desJardins. Agent-Organized Networks for Multi-Agent Production and Exchange. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 05)*. Pittsburgh, PA, July 2005.

Matthew Gaston and Marie desJardins. A Simple Learning Approach for Endogenous Network Formation. In *Proceedings of the AAAI 2005 Workshop on Multi-Agent Learning*. Pittsburgh, PA, July 2005.

Blazej Bulka, Matthew Gaston, and Marie desJardins. Local Learning to Improve Organizational Performance in Networked Multi-agent Team Formation. In *Proceedings of the AAAI 2005 Workshop on Multi-Agent Learning*. Pittsburgh, PA, July 2005.

Matthew Gaston and Marie desJardins. Agent-Organized Networks for Dynamic Team Formation. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 05)*. Utrecht, Netherlands, July 2005.

Matthew Gaston and Marie desJardins. Social Network Structures and their Impact on Multi-agent System Dynamics. In *Proceedings of the 18th International Florida Artificial Intelligence Research Society Conference (FLAIRS 05)*. Clearwater, FL, May 2005.

Matthew Gaston, John Simmons, and Marie desJardins. Adapting Network Structures for Efficient Team Formation. In *Proceedings of the AAAI-2004 Fall Symposium on Learning in Multi-agent Systems*. Washington, D.C., October 2004. New York, N.Y., July 2004.

Matthew Gaston and Marie desJardins. Team Formation in Complex Networks. In *Proceedings of the First North American Association for Computational Social and Organization Science Conference (NAACSOS)*. Pittsburgh, PA, June 2003.

Matthew Gaston, Lara Diamond, and Miro Kraetzl. An Observation of Power Law Statistics in the Dynamics of Communication Networks. In *Proceedings of Information, Decision and Control (IDC 2002)*. Adelaide, South Australia, Feb 2002.

Norm Curet, Jason DeVinney, and Matthew Gaston. An Efficient Network Flow Code for Finding All Minimum Cost *s-t* Cutsets. *Computers & Operations Research*. Vol. 29, Issue 3, 2002, pp. 205-219.

Professional positions held:

2002-Present	Technical Leader Advanced Analysis Laboratory, NSA.
1998-2002	Senior Operations Research Analyst (Officer, USAF) Office of Modeling and Simulation, NSA.

ABSTRACT

Title of Dissertation: Organizational Learning and Network Adaptation in Multi-Agent Systems

Matthew E. Gaston, Doctor of Philosophy, 2005

Dissertation directed by: Dr. Marie desJardins
Assistant Professor
Department of Computer Science and Electrical Engineering

In both real and artificial societies, successful organizations are highly dependent upon a structure that fosters effective and efficient behavior at both the individual and the organizational levels. In multi-agent systems, groups of agents must coordinate effectively in order to solve problems, allocate tasks across a distributed organization, collectively distribute knowledge and information, and achieve collective goals. The organizational structure of a multi-agent system dictates the interactions among the agents, and can play a significant role in the overall performance of a society of agents.

Given the importance of organizational network structures for multi-agent systems, distributed network adaptation is a promising approach for organizational learning. After reviewing related work in multi-agent learning, the structure and dynamics of networks, and organizational learning in multi-agent systems, I present the concept of *agent-organized* *networks* as an approach for organizational learning by distributed network adaptation. Supported by theoretical evidence of the complexity of organizational network design, a general learning-based agent-organized network framework is proposed and applied to two general environments: multi-agent team formation and a production and exchange economy. In addition, the general framework is used to develop distributed network adaptation strategies for specific applications in supply network formation and wireless sensor networks. Experimental results for both the general and specific multi-agent environments support the hypothesis that distributed management and adaptation of organizational network structure leads to improved collective performance in multi-agent systems. Analyses of the structural characteristics of the networks as they evolve are used to aid in understanding the behavior of agent-organized networks and further support the utility of distributed network adaptation for organizational learning in multi-agent systems.

ORGANIZATIONAL LEARNING AND NETWORK ADAPTATION

IN MULTI-AGENT SYSTEMS

by
Matthew E. Gaston

Dissertation submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2005

For Megan, Lily, and the baby to come.

Acknowledgments

First and foremost, I thank my wife, Megan, and daughter, Lily, for their unending patience, encouragement, and care over the past several years. They have endured countless hours of my absence and have never wavered in their extreme support for my work.

I am most grateful to two other amazing women, my mother and my grandmother. Without their continual support and encouragement for more than 29 years I would not be where I am today.

My advisor, Marie desJardins, has been an unbelievable mentor, advocate, promoter, teacher, and colleague. To her, I am forever grateful. Additionally, I wish to thank my committee, Tim Finin, David Jensen, Tim Oates, and Yun Peng, for their invaluable insight and support.

I would also like to thank my current employer for financial support and the wonderful possibilities afforded to me.

Finally, I thank all of my family, friends, and colleagues that have encouraged, inspired, helped, and guided me over the years. The number of people that have positively affected my success is enormous. The size of the list is directly correlated with the strength of support. For everyone that has helped and supported me I am extremely grateful. I look forward to their continued support and the opportunity to repay them in as many ways as possible in the future.

MATTHEW E. GASTON

University of Maryland Baltimore County
December 2005

Contents

Acknowledgments **iii**

1 Introduction **1**

1.1 What is Multi-Agent Organizational Learning? 1

1.2 Networked Multi-Agent Systems: It is Both Who You Know and What You Know 3

1.3 Major Contributions: Distributed Organizational Learning by Network Adaptation 5

1.4 Two Motivational Application Domains 6

1.5 Chapter Overview 7

2 Multi-Agent Learning **9**

2.1 Overview of Learning in Multi-Agent Systems 10

2.1.1 Differencing Features for Multi-Agent Learning 12

2.1.2 Challenges in Multi-Agent Learning 15

2.2 Multi-Agent Learning Paradigms 17

2.2.1 The Individual Learner Perspective 17

2.2.2 Evolutionary Computation 18

2.2.3 Multi-Agent Reinforcement Learning 19

2.2.4 Computational-Mechanism Design 25

3 Networks: Structure, Function, and Formation 27

3.1 Properties Observed in Real-World Networks 28

3.2 Modeling Regular and Complex Networks 30

3.2.1 Regular Networks 30

3.2.2 Random Graphs 32

3.2.3 Random Geometric Graphs 33

3.2.4 Small-World Networks 34

3.2.5 Scale-Free Graphs 36

3.3 Statistical Measures for Dynamic Networks 38

3.3.1 Mean Path Length 38

3.3.2 Clustering 39

3.3.3 Deviation in the Degree Distribution 40

3.3.4 Node (Degree) Correlation 41

3.4 Network Formation Games in Economics 42

3.4.1 The (Symmetric) Connections Model 43

3.4.2 Stability and Efficiency 44
3.4.3 A Dynamic Network Formation Process 48
3.4.4 An Experiment with the Symmetric Connection Model 50
3.5 Concluding Remarks 54

4 Organizational Learning and Network Adaptation 55

4.1 Organizational Learning and Network Structures 56
4.1.1 Local Perception and Local Learning 58
4.1.2 Bottom-up vs. Top-down Network Formation 59
4.1.3 On the Cost of Connections 60
4.1.4 The “Laterality” of Connections 61
4.2 Related Work on Organizational Learning 62
4.2.1 Coalition and Congregation Formation 63
4.2.2 Organizational Self-Design 65
4.2.3 Evolving Organizations 67
4.2.4 Adaptive Networks and Game Theory 68
4.2.5 Referral Networks, Peer-to-peer Systems, and Information Retrieval 69
4.3 On The Complexity of Finding Optimal Network Structures 71
4.4 Agent-Organized Networks 76
4.4.1 When to Adapt Local Connectivity? 77
4.4.2 How to Adapt Local Connectivity? 78

4.4.3 Stability and Performance 83
4.4.4 A General Learning-based AON Framework 84
4.5 Concluding Remarks 88

5 Networked Multi-Agent Team Formation 90

5.1 Overview of Multi-Agent Team Formation 91
5.2 A Model of Team Formation in Agent Networks 94
5.2.1 The Model 95
5.2.2 The Effects of Network Structure 103
5.3 Agent-Organized Networks for Team Formation 113
5.3.1 From Random to Intelligent AONs 113
5.3.2 Experimental Results 118
5.3.3 Are Push Referrals Always Best? 130
5.3.4 History-Based AONs for Team Formation 132
5.3.5 On Bilaterally Stable AONs for Team Formation 136
5.4 Concluding Remarks 139

6 Production and Exchange Networks 140

6.1 Overview of Multi-Agent Market Environments 141
6.2 Modeling Decentralized Multi-Agent Markets 145
6.2.1 A Model of Production and Exchange 145

6.2.2 The Effects of Network Structure 149

6.3 AONs for the Production and Exchange Economy 152

6.3.1 Increasingly Intelligent AONs 153

6.3.2 Using Agent Production Capacities 163

6.3.3 A Threshold-Based Alternative 165

6.4 Concluding Remarks 169

7 Applications of Agent-Organized Networks 170

7.1 Supply Chain Networks 170

7.1.1 Modeling Multi-Agent Supply Chain Networks 174

7.1.2 An AON for Supply Network Formation 181

7.1.3 Experimental Results and Discussion 183

7.2 Topology Control in Sensor Networks 189

7.2.1 Overview of Topology Control 190

7.2.2 The Topology Control Model 194

7.2.3 A Centralized Algorithm: CONNECT 198

7.2.4 A Decentralized Algorithm: LINT 201

7.2.5 An AON for Topology Control 204

7.2.6 Experimental Results and Discussion 207

7.3 Concluding Remarks 211

8 Summary and Conclusions 212

8.1 On The Design and Dynamics of Multi-Agent Organizations . . . 212

8.2 Navigating Social Structures in Networked Multi-Agent Systems . . . 214

8.3 Future Directions and Final Thoughts . . . 215

References 219

List of Tables

5.1 Properties of Simple Example Networks 109

5.2 Structural Statistics for Fixed Random Graphs 121

6.1 Structural Statistics for Fixed Random Graphs 156

6.2 Comparison of Structure Statistics for *Q/minNeighbor/pushProd* 165

6.3 Structure Statistics for *threshold/minNeighbor/pushMax* 168

7.1 Parameters for Supply Network Formation Experiments 185

7.2 Parameters for Topology Control Experiments 207

List of Figures

2.1 Examples of Matrix Games 21

3.1 A Random Geometric Graph 33

3.2 Small-world Networks 36

3.3 Scale-Free Network and Degree Distribution 37

3.4 Results of a Connections Model Experiment 51

3.5 Evolution of the Watts Network Formation Process 53

5.1 The algorithm used for each agent to decide which teams to initiate and which teams to join. 100

5.2 An Example of Networked Team Formation 101

5.3 The Ring and Star Networks 105

5.4 Effects of Network Structure on Team Formation 112

5.5 Average Performance on Fixed Random Networks 120

5.6 The Performance of Four AONs 122

5.7 Network Structure Statistics for Four AONs 125

5.8 Representative Network Structures from AONs 129

5.9 Comparing an Alternative Push Referral 130

5.10 The Network after Philanthropic Referrals 131

5.11 Results of Teammate History-Based AONs 133

5.12 Network Structures of Teammate History-Based AONs 134

5.13 Results of a Bilateral AON for Team Formation 137

6.1 Effects of Network Structure on the Production and Exchange Market . . . 151

6.2 Relative Performance of AONs 157

6.3 Structural Statistics Resulting from Four AONs 159

6.4 Sample Networks from the Four AONs 161

6.5 Production and Surplus Characteristics of AONs 162

6.6 Performance of *Q/minNeighbor/pushProd* 164

6.7 A Network resulting from *Q/minNeighbor/pushProd* 165

6.8 Performance of *threshold/minNeighbor/pushMax* 167

6.9 Sample of a Network resulting from *threshold/minNeighbor/pushMax* . . . 168

7.1 An Illustration of a Supply Network 175

7.2 A Random Supply Network 176

7.3 Observation of “The Bullwhip Effect” 180

7.4 Experimental Results for AON-based Supply Network Formation 186

7.5 Experimental Results for AON-based Supply Network Formation 188

7.6 A Network Resulting from CONNECT 199

7.7 Pseudocode for the CONNECT Algorithm 200

7.8 Networks Resulting from LINT 204

7.9 Sensor Nodes at Maximum Power 208

7.10 Networks Resulting from AON Topology Control 209

7.11 Experimental Results of AONs for Topology Control 210

Chapter 1

Introduction

In life, its not what you know or who you know that counts – it is both!
Anthony J. D'Angelo

Jingshen is the Mandarin word for spirit and vivacity. It is an important word for those who would lead, because above all things, spirit and vivacity set effective organizations apart from those that will decline and die.
James L. Hayes

1.1 What is Multi-Agent Organizational Learning?

Organizational learning can be defined, or described, in many ways. One possible description is that "organizational learning refers to the increases in productivity that are observed as firms gain experience in production" (Huberman 2001). Another description of organizational learning is the collective ability of a multi-agent system to improve its global performance based solely on the actions, decisions, and experiences of its constituent agents. This second description of organizational learning focuses on the influences that individual behaviors have on global performance (Brown & Duguid 1996). In this second connotation, global performance refers to the overall, macro-level, collective performance of the entire system. The latter description of organizational learning is more appropriately a description of *distributed organizational learning*.

In distributed organizational learning, the organization is not monolithic or centrally controlled; rather, it is comprised of a collection of semi-autonomous, interdependent agents with a common goal. Distributed organizational learning differs from centralized learning, or single-agent learning, in that both information and decisions are decentralized. That is, no single agent possesses complete information about the organization. Furthermore, all of the agents' decisions and actions influence the organization's behavior and performance. This notion of distributed organizational learning is the one I adopt throughout this dissertation.

The decentralization of information and decisions in distributed organizational learning presents a set of interesting and unique challenges.

> If one's own actions are embedded in an ecology of the actions of many others (who are simultaneously learning and changing), it is not easy to understand what is going on. The relationship between the actions of individuals in the organization and overall organizational performance is confounded by simultaneous learning of other actors (Levinthal & March 1993).

In situations where learning is distributed throughout an organization of agents, the agents must rely on local perception of global performance and partial information about the actions of other agents in the organization.

The challenges of distributed organizational learning are particularly apparent in large, open multi-agent systems, where it is impossible for all agents to interact continuously with all other agents. One example of a large-scale, open multi-agent environment is peer-to-peer information retrieval (Yu & Singh 2003). In peer-to-peer information retrieval, agents can come and go as they please and they are subject to failures. Obviously, the set of active agents and the set of active interactions among the agents will change over time. In large, open multi-agent systems of this type, the limited interactions among the agents can be the result of cognitive, communications, or computational constraints. In very large systems, it may be that the agents simply do not "know" about one another. I refer to collections of agents where the agent interactions are limited by an agent social network as *networked multi-agent systems*.

1.2 Networked Multi-Agent Systems: It is Both Who You Know and What You Know

As multi-agent systems continue to grow and migrate to heterogeneous environments such as the Internet and the Semantic Web (Hendler 2001), the structure of multi-agent societies and the interconnections among the agents in these societies will be fundamental to the effectiveness of agent organizations. Agents operating in these environments will be unable to maintain working knowledge of all other agents, leading to a social network structure induced by agent interactions. As the size and diversity of multi-agent systems grow, understanding the impact of agent social structures on the dynamics of multi-agent systems is essential.

Recent research on real-world networks has revealed that social structures have a much richer structure than regular or random networks (Newman 2003; Strogatz 2002). The types of networks analyzed in these studies include the Internet and World Wide Web (WWW), scientific collaboration networks, gene interaction networks, various social networks, and neural networks (Newman 2003). Although much research in artificial intelligence has focused on agent-level cognitive mechanisms in multi-agent systems, little effort has been devoted to understanding the impact that agent social structures have on the collective ability of a society of agents.

Recent findings suggest that the interaction topologies of multi-agent systems have a significant effect on the overall behavior of agent organizations. Work in computational organization theory (Carley & Gasser 1999; Carley 2002) highlights the importance of social structure on organizational performance for various types of artificial societies. Such studies have shown that clustered organizations, where agents have few connections but are tightly connected, are more stable than organizations with high connectivity (Huberman & Hogg 1995). Network structure has been shown to affect the rate of the adoption of social conventions in an agent society (Delgado 2002). In the context of multi-agent team formation, networks that exhibit short average path length allow for greater diversity in teams of agents as well as efficiency in forming teams (Gaston & desJardins 2003; Gaston, Simmons, & desJardins 2004). More recently, research on multi-agent games in complex network structures demonstrated that variations in network structure lead to significantly different stabilities in agent strategies (Abramson & Kuperman 2001; Holme *et al.* 2003; Kim *et al.* 2002; Szolnoki & Szabo 2004). These findings highlight the importance of the agent social network that constrains the agent interactions in a multi-agent system.

1.3 Major Contributions: Distributed Organizational Learning by Network Adaptation

Given that the network structure of a multi-agent system can have a significant impact on its organizational performance, there is a tremendous, unexploited opportunity for organizational learning in multi-agent systems. My work demonstrates, for the first time, the significant potential of endowing individual agents with the ability to dynamically adapt the structure of the agent social networks in large agent societies.

More specifically, I show that agents can improve organizational performance by adapting their network structures using only local information and relatively simple methods based on stateless Q-learning. The claim that adaptive networks lead to increased performance, and the generality of the methods for local, agent-driven network adaptation, are supported by empirical results in two generic multi-agent environments and two specific application domains. The two generic environments are multi-agent team formation and a distributed production and exchange network. The specific application domains are large-scale supply chain networks and topology control for sensor networks.

1.4 Two Motivational Application Domains

There are many potential applications of agent-organized networks. These applications center around large, open multi-agent systems, where the control of the agents does not fall under a single authority. One such application domain is e-commerce. A well-known issue for enabling agent-mediated e-commerce is the so-called "discovery problem" (He, Jennings, & Leung 2003). The discovery problem is that of getting the right agents to interact with one another. This problem becomes increasingly complex when the system is open and the number of agents is large, such as in large-scale supply networks (Walsh & Wellman 2000).

Another motivational application for agent-organized networks is the formation and reformation of virtual organizations for Grid Computing (Foster, Jennings, & Kesselman 2004; Norman *et al.* 2004). Grid computing is "predominantly concerned with coordinated resource sharing and problem solving in dynamic, multi-institutional, virtual organizations" (Foster, Kesselman, & Tuecke 2001). The premise of Grid Computing is to enable effective use of data and computational resources spread over a wide-area network. Agent-organized networks have the potential to provide an automated mechanism for coordinating and dynamically interconnecting these resources on a large scale.

1.5 Chapter Overview

The rest of this dissertation is structured as follows. In Chapter 2, I briefly survey the state of the art in a subfield of multi-agent systems and machine learning known as *multi-agent learning*. The focus of multi-agent learning is on learning schemes for individual agents in situations where they interact with other (learning) agents. In this survey, I identify several of the important differences between single-agent learning and learning when there are many agents.

Chapter 3 is an overview of the structure, function, and dynamics of networks. In this chapter, I survey the recent literature on “complex” networks. I also discuss several statistical measures of network structure that are useful in understanding the behavior of agent-driven adaptive networks discussed in later chapters.

Following the reviews of multi-agent learning and networks, Chapter 4 focuses on organizational learning. In this chapter, I survey related work on organizational learning. I also describe in detail the concept of *agent-organized networks*. The chapter concludes with a description of a general learning-based framework for agent-organized networks.

In Chapters 5 and 6, I explore the behavior of agent-organized networks in two general multi-agent domains: team formation and a general market environment. In these chapters, I develop and empirically analyze several different agent-organized networks. My analysis includes the results of virtual experiments in each of the domains as well as analyses of the network structures as they evolve under different adaptation regimes.

Moving from general multi-agent domains to more specific multi-agent environments, I focus on applications of agent-organized networks in Chapter 7. Specifically, I apply agent-organized networks to supply network formation and topology control in wireless sensor networks. In this chapter, I demonstrate the utility of agent-organized networks by showing that individual agents adapting their local connectivity structure can improve the distribution of goods in a supply chain. I also demonstrate that agent-organized networks can form power-efficient topologies in wireless sensor networks.

Finally, the dissertation concludes in Chapter 8, with a review of the major results. I also provide a sampling of important future directions. Distributed network adaptation as a means of organizational learning is a new area of study in multi-agent systems. While this dissertation lays the ground work for the study of agent-organized networks, there are many exciting directions for this research in the future.

Chapter 2

Multi-Agent Learning

Learning is not attained by chance, it must be sought for with ardor and attended to with diligence.

Abigail Adams

Of the many cues that influence behavior, at any one point in time, none is more common than the actions of others.

Albert Bandura

. . . [M]ulti-agent learning is not merely a matter of "straight" learning, but a matter involving complex patterns of social interaction and cognitive processes, which leads to complex collective functions.

Ron Sun (2001)

Machine learning is a subfield of artificial intelligence in which algorithms for learning from training examples or experience are designed, studied, and applied. Many of the techniques developed in machine learning (Mitchell 1997) can be transfered to settings where there are multiple, interdependent, interacting learning agents, although they may require modification to account for the other agents in the environment. In addition, multi-agent systems present a set of unique learning opportunities over and above single-learner machine learning. In particular, because of the nature of multi-agent systems, so-called "social learning" (Alonso *et al.* 2001; Conte & Paolucci 2001) opportunities abound. Examples of social learning include imitation, observation, and social facilitation, where agents explicitly transfer knowledge among one another (Conte & Paolucci 2001).

In this chapter, I provide an overview of multi-agent learning, including its features, its challenges, and several of the most commonly used techniques. Where possible, I attempt to place the overview in the context of organizational learning by network adaptation in multi-agent systems, the main topic of this dissertation.

2.1 Overview of Learning in Multi-Agent Systems

Learning is a fundamental part of intelligence. A typical characterization of the concept of learning is the ability to improve performance based on experience. Of course, the use of the term "experience" implies that the learning system is embedded within, able to sense, and able to affect some environment. It is common practice to refer to a learning system in an environment as an *adaptive agent*.

There is an enormous body of literature on the single-agent learning problem, in which agents use supervised or reinforcement learning mechanisms to improve their performance.[1]

[1]See Mitchell (1997) for a review of supervised learning and Sutton and Barto (1998) for a thorough review of reinforcement learning

In supervised learning, an agent, or learning algorithm, is presented with a set of training examples from which the agent builds a model for a specified decision or prediction problem. The model built from the training examples is then used to make predictions about previously unseen instances or situations. In reinforcement learning, an agent uses experience, or an explicit model of the environment, in order to learn an optimal policy for behaving in an environment. This optimal policy is based on discounted rewards received for taking actions in the environment. Despite the continued expansion and innovation in the field of machine learning, the single-agent learning problem remains an open and challenging problem.

Given that the single-agent learning problem is challenging, the learning problem becomes increasingly complex when many (learning) agents are embedded within the same environment. A system with more than one agent is referred to as a *multi-agent system*; when the agents are learning, it is considered *multi-agent learning*. The primary reason for the increase in complexity in environments where there are multiple learning agents is the non-stationarity of the environment. From a single agent's perspective in a multi-agent system, the other agents can be thought of as part of the environment. When the other agents are also learning, the environment is no longer stationary, but rather adaptive. This results in the so-called "moving target" challenge of multi-agent learning: that is, learning to behave or predict in a dynamically changing, perhaps non-deterministic, environment.

In this chapter, I survey the multi-agent learning literature in the context of the organizational learning mechanism proposed in this dissertation. This organizational learning mechanism performs decentralized network adaptation based on the individual decisions of agents in multi-agent organizations, and is discussed in detail in Chapter 4. In the remainder of this chapter, I address some of the issues, ideas, and challenges that are uniquely associated with multi-agent learning. The chapter continues with discussions of various multi-agent learning paradigms and methods and concludes with a brief overview of the focus of this dissertation: organizational learning by endogenous network formation.

2.1.1 Differencing Features for Multi-Agent Learning

Sen and Weiss describe a set of “differencing features” for characterizing learning in multi-agent systems (Sen & Weiss 1999). Here, I present and describe each of these differencing features and then characterize organizational learning by distributed network adaptation in the context of these features.

1. **The degree of decentralization.** Is the learning centralized or distributed? The two extremes are complete centralization and complete decentralization. In complete centralization, only one agent in the system learns or there is a central adaptive controller for all of the agents in the system. In complete decentralization, all of the agents in the system learn and adapt in a distributed and possibly simultaneous fashion.

2. **Interaction-specific features.** What is the nature of the interactions among the agents in the system? Sen and Weiss discuss the "level of interaction," the "pattern of interaction," and the variability of the interactions. Interactions can be based on observations, indirect effects from the environment (i.e., the use of a shared resource), or explicit relationships and interdependencies. Additionally, interactions can change over time as agents move through an environment or modify their relationships with other agents.

3. **Involvement-specific features.** How involved are each of the individual agents in the learning process? The learning of individual agents can be local (i.e., the environment is partially observable) or global. Sen and Weiss refer to local learning agents as "specialists," because they are trying to learn given only local, partial views of the environment.

4. **Goal-specific features.** Are the goals of the agents selfish or collective? Multi-agent systems can be competitive, cooperative, and, sometimes, both. In competitive systems, such as games, agents are selfish, attempting to maximize their individual reward regardless of the reward received by other agents. In cooperative systems, agents work toward common goals where the reward structure is shared, or common, among all of the agents. An example of a system that is simultaneously competitive and cooperative is academia. The agents in academia compete with one another for publications, promotions, tenure, and funding. At the same time, collaboration

among the agents leads to the production of more papers, and therefore more publications, which in turn increases the likelihood of promotions and tenure.

5. **The learning method.** How does learning take place for agents in the system? Learning in multi-agent systems presents possibilities for alternative learning methods to methods used in single-learner settings. One way to characterize these alternative methods is *social learning*. Sen and Weiss present several social learning methods, including "learning by advice taking," "learning by observation," and "learning by example." Of course, many single-learner methods, such as "learning by discovery" and "learning from examples," are also applicable, possibly with modifications, to multi-agent learning scenarios.

6. **The learning feedback.** How do agents know if their decisions and behaviors are beneficial or detrimental? Feedback, in the form of correct answers, can be given directly to learning agents, as in supervised learning. A more likely scenario for multi-agent learning is feedback through reinforcement of behaviors. This reinforcement is not necessarily "correct" answers, but rather dispositional information about decision made or actions taken. This is the reinforcement learning model, which is described in more detail later in this chapter.

Although a complete treatment of organizational learning by network adaptation is given in Chapter 4, I characterize the problem of distributed organizational structure learning along these six differencing features here. (1) The learning problem is completely decentralized. (2) The interactions among the agents are the focal point of learning. These interactions are dynamic, changing based on decisions of the individual agents, and these dynamic interactions represent the only source of learning in the systems under considerations. (3) Because the learning is completely decentralized, all of the agents are necessarily involved in the learning process. (4) Because I focus on organizational learning, the goal of decentralized network adaptation is to improve collective performance. (5) While many learning methods are possible in the organizational learning by network adaptation problem, I primarily focus on learning by discovery and learning by advice taking (the latter leading to the referral-based strategies discussed in Chapter 4). (6) The learning feedback, in general, is based on local feedback. Further details of organizational learning by network adaptation are given in Chapter 4-7.

2.1.2 Challenges in Multi-Agent Learning

In multi-agent systems, particularly systems containing agents that learn, there are several well known social pathologies (Jensen & Lesser 2002). In multi-agent learning, social pathologies arise when one agent's adaptations lead to improved local performance for the agent, but decreased collective performance across the organization of agents.

Perhaps the mostly widely known and widely studied pathology in multi-agent systems is the *tragedy of the commons* (TOC) (Hardin 1968; Jensen & Lesser 2002). In the TOC, one agent benefits from (over) accessing or utilizing part of the environment, usually a shared resource, at the expense of the other agents in the system. The abuse of a common resource by a single agent leads to the abuse of the common resource by other agents in this system. This snowballing effect leads quickly to an over-burdening of the shared resource, diminishing the performance of the entire system.

Other pathologies found in systems of learning agents include (Jensen & Lesser 2002) *lock in*, where agents are given incentives to conform when conformity may lead to sub-optimal states; *cycling*, where one agent's adaptations lead to another agent's adaptations, leading back to adaptations of the first agent and so on; and *blocking*, where the behavior of one agent in the system prevents other agents from moving to higher-valued states or behavioral modes.

Another challenge associated with multi-agent learning, which is a carry-over from single-learner machine learning, is the *credit assignment problem*. A succinct characterization of the credit assignment problem in multi-agent learning is "what action carried out by what agent contributed to what extent to the performance change" (Sen & Weiss 1999). The credit assignment problem is further compounded in completely decentralized systems where there is no central controller and no uniform reinforcement signal. In the decentralized scenario, agents must locally perceive changes in performance and appropriately attribute these changes to the actions of others or themselves. The credit assignment problem, as well as the other challenges described above, are considerations I will address when considering decentralized organizational learning by network adaptation in later chapters.

2.2 Multi-Agent Learning Paradigms

There are many multi-agent learning paradigms. In this section, I briefly review several of the most widely studied methods in multi-agent learning. The section on reinforcement learning is given additional attention since reinforcement learning techniques are used in many of the methods proposed in this dissertation.

2.2.1 The Individual Learner Perspective

One approach to multi-agent learning is to ignore the fact that there are multiple, adaptive agents in the system. In this approach, an individual agent learns using any of the applicable standard techniques from machine learning (Mitchell 1997) and simply treats other agents in the system as part of the environment. The great benefit of this approach is the wide variety of learning algorithms and techniques available.

While the individual learner perspective may prove useful in some settings, agents that treat other agents simply as part of the environment are likely to suffer from the so-called "moving target" problem. That is, when an agent is learning from the individual learner perspective, the environment is non-stationary, but the learning agent is not accounting for this non-stationarity. A likely scenario is that the agent learning from an individual learner perspective will be continuously playing "catch-up" with a dynamic and adaptive environment.

2.2.2 Evolutionary Computation

Evolutionary computation is the discipline and practice of using the principles of Darwinian (or Lamarkian) evolution to search for solutions or approximations to problems. These principles include natural selection and the use of genetic operators such as crossover and mutation of the genes of individuals in a population (Holland 1975). Popular techniques in evolutionary computation include genetic algorithms (Holland 1975) and genetic programming (Koza 1992). In these techniques, a population of individual solutions to the problem is evolved using fitness-based selection techniques to drive the population from one generation to the next.

Co-evolutionary algorithms extend the use of genetic algorithms and other techniques to learning for multi-agent systems (Potter & Jong 2000). In co-evolutionary algorithms, the individuals in the populations actually represent collection of agents. Using this representation, the strategies of individual agents evolve, but they evolve in the context of the strategies of their “teammates.” In order to improve the performance of multi-agent teams, co-evolutionary algorithms provide reward, or reinforcement, to collections of agents as opposed to each agent receiving its own reward. Essentially, agent strategies are directly and explicitly coupled in co-evolutionary techniques.

Several studies have demonstrated the improvement that co-evolutionary algorithms provide over simple evolutionary techniques. Using cooperative co-evolution, neural network controllers for individual simulated agents in a predator-prey environment were evolved and shown to be more effective than an evolved central neural network controller (Yong & Miikkulainen 2001). Co-evolutionary techniques have also been applied to the control of multi-robot teams (Liu & Iba 2003). In another study, "adaptable auctions" based on co-evolutionary techniques were used to provide team reward functions for role learning in multi-agent systems.

While co-evolutionary techniques are designed for multi-agent systems, they remain a centralized approach. The learning, including both adaptation and reinforcement, in co-evolutionary algorithms is centralized, although the resulting behaviors of the agents in the system may be distributed. Although co-evolutionary techniques have proven useful in cooperative multi-agent domains, extending the paradigm to large-scale, open multi-agent systems is a significant challenge.

2.2.3 Multi-Agent Reinforcement Learning

Reinforcement learning (RL) is one of the most widely studied learning mechanisms for multi-agent learning. The RL approach to multi-agent learning focuses on agents learning equilibrium strategies in (repeated) stochastic games. I first briefly introduce the concept of a Markov Decision Process (MDP) and the widely studied Q-learning algorithm for finding an optimal policy for an MDP. I then discuss extending MDPs to stochastic games (and the special case of matrix games) and provide a summary of several methods that have been proposed for extending Q-learning for multi-agent learning in stochastic games.

A *Markov Decision Process* (MDP) (Bellman 1957; Sutton & Barto 1998; Kaelbling, Littman, & Moore 1996) is a tuple, ($\mathcal{S}$, $\mathcal{A}$, T, R), where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is a transition function, and $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a reward function. The transition function is a probability distribution that provides the probability of transitioning to state s' given that the current state is s and the action a is taken. The reward function maps the current state and action pair to a real-valued reward. The goal of solving an MDP is to find an optimal policy, $\pi : \mathcal{S} \rightarrow \mathcal{A}$, which maximizes the discounted future rewards. The discount factor is typically given as γ.

The basis for many existing multi-agent reinforcement learning algorithms is the much studied (single-agent) *Q*-learning strategy for solving Markov Decision Processes (MDPs) (Watkins & Dayan 1992; Kaelbling, Littman, & Moore 1996). *Q*-learning, which is used to compute an optimal policy for an MDP, is based on the following equations:

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha[R(s, a) + \gamma V(s')], \tag{2.1}$$

and

$$V(s) \leftarrow \max_{a \in A} Q(s, a), \tag{2.2}$$

where $Q(s, a)$ is the estimated value of taking action a in state s, $V(s)$ is the estimated value of state s, $R(s, a)$ is the reward for taking action a in state s, s' is the resulting state from taking action a in state s, α is the learning rate, and γ is the discount factor for future rewards. It is well known that Q-learning converges on the optimal policy, V^*, under certain assumptions about how the state space is explored (Watkins & Dayan 1992).

	heads	tails
heads	1	-1
tails	-1	1

(a)

	rock	paper	scissors
rock	0	-1	1
paper	1	0	-1
scissors	-1	1	0

(b)

Figure 2.1: Examples of two common matrix games: a) matching pennies and b) rock-paper-scissors (Ro-Sham-Bo).

Extending reinforcement learning to apply to multi-agent learning domains also involves extending MDPs to that of stochastic games. A *stochastic game* (Bowling & Veloso 2002), or *Markov game* (Littman 1994), is a tuple (n, $\mathcal{S}$, $\vec{\mathcal{A}}$, T, $\vec{R}$), where n is the number of players, $\mathcal{S}$ is the set of states, $\vec{\mathcal{A}}$ is the joint action space $A_1 \times A_2 \times \ldots \times A_n$ where A_i is the set of actions available to the ith player, $T : \mathcal{S} \times \vec{\mathcal{A}} \times \mathcal{S} \rightarrow [0, 1]$ is a transition function, and $\vec{R}$ is the set of reward functions with $R_i : \mathcal{S} \times \vec{\mathcal{A}} \rightarrow \mathbb{R}$ being the reward function for the ith player. An MDP is a special case of a stochastic game with $n = 1$. Another special case of the stochastic game is the *matrix game*. A matrix game is a stochastic game with $|S| = 1$ (i.e., a stateless stochastic game). Figure 2.1 shows the payoff matrices for several popular matrix games. Strictly cooperative stochastic games, or team games, are games where all of the agents have the same reward function. Strictly competitive stochastic games are zero-sum two-player games (Bowling & Veloso 2002). Much of the work on “solving” stochastic games is focused on finding equilibrium strategies (i.e., strategies where no agent will desire to deviate from the equilibrium strategy when all other agents continue to play the equilibrium strategy).

It is possible to have a multi-agent system where all of the agents simultaneously learn using standard Q-learning, but this is simply the single-agent perspective (see above) (Sen, Sekaran, & Hale 1994). In order to extend the Q-learning algorithm to the multi-agent learning domain, an agent's Q values can be made a function of all of the agents' actions:

$$Q(s, \vec{a}) \leftarrow (1 - \alpha)Q(s, \vec{a}) + \alpha[R(s, \vec{a}) + \gamma V(s')], \tag{2.3}$$

where $\vec{a}$ is the vector of all agents' actions (Littman 2001b; Hu & Wellman 2002). Once the Q values are given as a function of all of the agents' actions, the question remains as to how to update the state value function V. Many methods have been proposed, with several of them briefly summarized here:

- **minimax-Q** (Littman 1994): One of the first multi-agent reinforcement learning algorithms, minimax-Q is restricted to two-player zero sum games. The state value function is updated by the minimax rule, which assumes a player's opponent is fully rational, and will choose the action that minimizes the players expected reward. The update to the state value function is given as

$$V_i(s) \leftarrow \max_{P_1 \in \Pi(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} P_1(a_1) Q_1(s, (a_1, a_2)), \tag{2.4}$$

  where $P_1(a_1)$ is the probability that player 1 selects action a_1 and $\Pi(A_1)$ is the space of all policies for player one.

- **Joint Action Learner (JAL)** (Claus & Boutilier 1998): In the JAL multi-agent Q-learning algorithm, agent i learns a probability distribution of the actions of all of the other agents in the game. Agent i then selects the maximum-utility action, using the learned probability distribution (i.e., its belief) to calculate expected value. The state value function is updated following

$$V_i(s) \leftarrow \max_{a_i} \sum_{a_{-i} \in A_{-i}} P_i(s, a_{-i}) Q_i(s, (a_i, a_{-i})), \tag{2.5}$$

where a_{-i} represents the actions of all agents other than agent i. The probability distribution over agent actions is based on past empirical frequency.

- **Nash-Q** (Hu & Wellman 1998): In this algorithm, the agents select a Nash equilibrium strategy based on the payoffs of all of the players in the game, and update the state value function according to this Nash equilibrium. While applicable to general-sum stochastic games, there can be many equilibria and they can be difficult to compute.

- **Friend-or-Foe** (Littman 2001a): The Friend-or-Foe multi-agent reinformcement learning algorithm divides stochastic games into two classes. Friend games are games where there is a globally optimal policy that maximizes the payoff for all agents. Foe games assume competition between agent payoffs. For two-person games, the state value functions are updated for the two classes of stochastic games as follows:

$$\text{Friend:} \quad V_1(s) \leftarrow \max_{a_1 \in A_1, a_2 \in A_2} Q_1(s, (a_1, a_2)) \tag{2.6}$$

$$\text{Foe:} \quad V_1(s) \leftarrow \max_{P_1 \in \Pi(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} P_1(a_1) Q_1(s, (a_1, a_2)). \tag{2.7}$$

It can be seen that the Friend update rule follows standard Q-learning and the foe update rule follows minimax-Q.

Learning in stochastic games becomes even more difficult when the environment, from the perspective of each agent, is partially observable. In this setting, each of the agents in the game may have a different belief about the true state of the world. Such games are called *partially observable stochastic games* (POSGs). Computing optimal policies for POSGs is intractable (Emery-Montemerlo *et al.* 2004). Most of the work on POSGs is focused on algorithms for finding approximate solutions (Emery-Montemerlo *et al.* 2004; Peshkin *et al.* 2000) although, more recently, a dynamic programming algorithm for finding optimal solutions has been proposed (Hansen, Bernstein, & Zilberstein 2004).

I have presented the standard formulation of reinforcement learning and surveyed extensions of Q-learning to multi-agent reinforcement learning in stochastic games. The primary focus of this dissertation, organizational learning by decentralized network adaptation, is not formulated as a multi-agent stochastic game for several reasons, including the complexity of the problem, the size of the state space, the computational complexity of solving the problem, and the uncertainty associated with the restriction to local information. However, my approach to organizational learning by network adaptation has been influenced by the previous work on multi-agent reinforcement learning.

2.2.4 Computational-Mechanism Design

Computational-mechanism design (CMD) is a relatively new subfield of research in multi-agent systems, with implications for multi-agent learning (Dash, Jennings, & Parkes 2003). CMD has grown out of the economic theory of mechanism design, which is primarily focused on designing interaction methodologies for collections of individual actors so as to achieve some desired system-wide behavior. I include CMD in the discussion of multi-agent learning paradigms because recent work has focused on distributed mechanism design, learning mechanisms, and on-line mechanisms.

In CMD, agents have preferences, called the agent *type*, which determine the agents' utilities given certain observations (i.e., occurrence of an event). The agents select strategies based on their types, attempting to maximize their utility (i.e., the agents are assumed to be economically rational). CMD also includes a so-called *social choice function*, which maps the set of agent types to a desired observation, or outcome. The goal of CMD is to design a social choice function that achieves some specific system-wide behavior, such as Pareto optimality or efficiency (Dash, Jennings, & Parkes 2003). In essence, CMD is a control mechanism for the collective behavior of a multi-agent system.

As described above, CMD is a centralized approach to governing the behavior of multi-agent systems. It involves collecting information about all of the agents, solving (or approximating) a centralized combinatorial optimization problem, and distributing the result to the individual agents. More recently, work on distributed mechanism design (DMD) has considered how to decentralize the selection of the optimal outcome (Parkes & Shneidman 2004) and has been applied to various issues in communications networks (Feigenbaum & Shenker 2002). Although possible, DMD presents a new set of challenges for computational-mechanism design, including trust, communication complexity, and the topology of the agent-to-agent interactions (Dash, Jennings, & Parkes 2003). In this light, CMD and DMD are important application domains of the work presented in this dissertation.

Closely related to computational-mechanism design is *collective intelligence* (COIN). The goal of COIN is to assign credit to individual agents such that some global objective is achieved (Wolpert & Tumer 1999; Tumer & Wolpert 2000). Essentially, COIN aims at designing solutions to the credit assignment problem. In applications of COIN, reward functions for individual agents in multi-agent systems are designed in an attempt to account for other agents in the system and to promote individual behaviors that result in common, collective goals.

There are also two emerging areas related to CMD that are directly related to learning in multi-agent systems. Moving from controlling multi-agent systems to multi-agent learning, *learnable mechanism design* (Parkes 2004) involves the design of mechanisms that aid agent learning of equilibrium strategies in multi-agent learning domains. Finallly, *automated mechanism design* (Parkes 2004), where mechanisms are formed in an online fashion and are adaptive, is also becoming an important issue for CMD (Dash, Jennings, & Parkes 2003; Freidman & Parkes 2003).

Chapter 3

Networks: Structure, Function, and Formation

> *As networks have permeated our world, the economy has come to resemble an ecology of organisms, interlinked and coevolving, constantly in flux, deeply tangled, ever expanding at its edges.*
>
> Kevin Kelley

> *Most people . . . would agree that a fundamental property of complex systems is that they are composed of a large number of components or "agents," interacting in some way such that their collective behavior is not a simple combination of their individual behaviors.*
>
> Mark Newman

The importance of networks permeates the world today. From biology to social systems, from the brain to the Internet, networks play an important and central role in the way the world works. In the last ten years, due in part to large increases in computational power, large-scale, real-world networks have received much attention from a variety of fields of study. In this chapter, I present a brief introduction to networks, including an overview of important properties observed in real-world networks, descriptions of several classes and models of network structures, and definitions of statistical measurements for characterizing network structures both statically and as they evolve. The chapter concludes by introducing the economic theory of network formation and a simple economic network formation model. The model is used to present a first example of agent-organized networks and to demonstrate the proposed statistical measurements for characterizing networks.

3.1 Properties Observed in Real-World Networks

Social networks, which have attracted the attention of scientists for many decades, are traditionally analyzed with the tools of graph theory. A *graph* $G = (V, E)$ can be used as a model of a social network, where the graph consists of a finite vertex set V, representing individuals, and a finite edge set E, representing relationships between individuals. An edge $e \in E$ is a pair of vertices denoting the endpoints of the edge. The degree of a node i, denoted k_i, is the number of edges connected to node i. The number of vertices in G is $|V|$ and the number of edges is $|E|$. Since I am primarily concerned with networked multi-agent systems, the terms node, vertex, and agent will all be used interchangeably to refer to elements of the set V in networks.

Three of the most frequently observed properties of real-world graphs are short average path lengths, excess clustering, and power-law degree distributions (Albert & Barabási

2002; Newman 2003). Commonly referred to as the "small-world effect," *short average path length* was first recognized by Milgram (1967) by analyzing the number of hops it took for postal mail to arrive at a specific destination. In these studies, mail was sent among people who knew each other on a first-name basis, with the goal of eventually getting the mail to a specified destination. Although not comprehensive, the studies concluded that it was possible for the mail to arrive at the specified destination and that it took, on average, approximately six hops to get from origin to destination. Average path length is calculated by taking the average of the lengths of the shortest paths between all pairs of nodes in the graph (Watts & Strogatz 1998; Newman 2003).

Excess clustering is found in networks that have more clustering than would be expected in a random graph of the same size (Newman 2003). The amount of *simple clustering* in a graph is defined to be the number of *three-cliques*, or triangles, that exist in the graph, normalized by the number of possible triangles. *Transitivity* is a generalized form of clustering that can be summarized as nodes having multiple shared neighbors with other nodes (e.g., in the two-dimensional lattice, the nodes on the opposite corners of each "square" share two common neighbors). Excess clustering has been observed in many real-world networks as a result of the fact that two nodes that are connected to a common neighbor are more likely to be connected.

The *degree distribution* of a network is the frequency of occurrence of nodes with each degree, or number of incident edges. Although there are many possible distributions for degree, it has been observed that many real-world networks have a highly skewed degree distribution. These distributions have "heavy tails," and in general follow a power law. That is, the probability $P(k)$ of a node in the network having degree k is proportional to $k^{-\gamma}$ for some parameter γ. Such networks have a hub-and-spoke structure, with some nodes having very large degree (Albert & Barabási 2002).

The three properties described above are used to understand the structure of complex networks. Later in this chapter, these properties will be used to suggest statistics for characterizing network structures and their evolution. In subsequent chapters, these properties and the derivative statistical measures will aid in understanding the structure of networks that evolve based on local decisions of agents in networked multi-agent systems.

3.2 Modeling Regular and Complex Networks

There are many models for the structure of networks. In this section, several network models are briefly surveyed with an emphasis on models that attempt to reproduce the properties of networks found in the real world.

3.2.1 Regular Networks

Regular graphs are best characterized as having a homogeneous connectivity pattern for all of the nodes in the graph. In these graphs, the degree distribution is trivial: all agents have exactly the same degree. Examples of regular graphs include lattices, hyper-cubes, and fully connected networks (i.e., all nodes are connected to all other nodes).

Lattice graphs are commonly used in understanding the behavior of agent-based systems (Epstein & Axtell 1996; Axtell 2000). In lattice graph topologies, nodes are given a logical ordering and connections among nodes are limited to nodes that are within close logical proximity. The dimension of a lattice determines the number of coordinates required to specify the node positions. In order to prevent boundary conditions (i.e., different node degrees for the nodes on the boundaries of a given lattice), a lattice can be wrapped upon itself. In one dimension, wrapping a lattice results in a ring topology. In two dimensions, wrapping a lattice results in a toroidal topology.

The other consideration in constructing lattice graphs is the coordination number, K (Watts 1999). In a lattice of dimension d, the coordination number determines the number of connections an agent has with its spatial "nearest neighbors" in each of the $2d$ directions. To illustrate how the coordination number affects the construction of a lattice, consider the one-dimensional lattice. The one-dimensional lattice with $K = 1$ is a simple connected ring of nodes. Increasing the coordination number to $K = 2$ results in a graph in which each node is connected to the four nodes with nearest proximity. An example of a one-dimensional lattice with $K = 2$ is shown in Figure 3.2(a). In a one-dimensional lattice with $K = 2$, each node is part of three triangles, or clusters. In this construction of lattices, if d is the dimensionality of the lattice, a node will always have $2dK$ edges. The amount of clustering in such a network is described in the discussion of small-world networks.

Lattice graph structures can exhibit excess clustering, but do not, in general, have short average path length.

3.2.2 Random Graphs

Random graphs were first introduced by Erdös and Renyi (1959). A *random graph* $G_{n,p}$ consists of n nodes where p denotes the probability of an edge existing between each pair of vertices. Random graph models have been widely studied since their properties can be computed analytically. For instance, the expected number of undirected edges in $G_{n,p}$ is $n(n-1)p/2$, and the average degree of a vertex is $k = p(n-1) \approx pn$, where the approximate solution holds for large n.

Recent evidence suggests that random graphs are not representative of real-world networks (Albert, Jeong, & Barabási 1999; Albert & Barabási 2002). They do possess short average path lengths (Albert & Barabási 2002): the average path length can be approximated as

$$\frac{\ln n}{\ln k} \approx \frac{\ln n}{\ln pn}. \tag{3.1}$$

On the other hand, random graphs do not exhibit clustering or a power-law degree distribution (the degree distribution follows a Gaussian distribution).

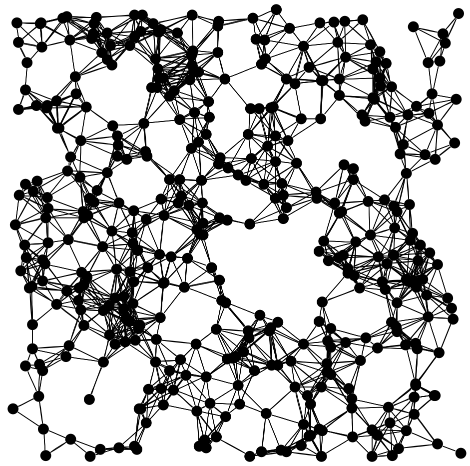


Figure 3.1: An instance of a random geometric graph on the unit square with 400 nodes and $\phi = 0.09$.

3.2.3 Random Geometric Graphs

A random geometric graph is generated by randomly placing N agents in the unit square and connecting two agents if they are within some specified distance d (Dall & Christensen 2002). More specifically, two agents, i and j are connected in a random geometric graph if $d(i, j) < \phi$, where ϕ is a threshold parameter of the model. Figure 3.1 shows an instance of a random geometric graph with $\phi = 0.09$.

Random geometric graphs are useful in modeling agent-based systems when the environment has an explicit geographic component (i.e., the agents are situated in a physical environment). In such an environment, communications are usually restricted to nearby neighbors due to line-of-sight and other considerations. An example of a multi-agent domain where random geometric graphs could serve to model the interconnectivity among the agents is intelligent sensor networks (Culler, Estrin, & Srivastava 2004).

A slight variation of random geometric graphs guarantees that every node will have at least on connection. In this variation, a preprocessing step is added to the construction. In order to produce network structures where all agents have connectivity, the minimal distance d_{min} can be computed by

$$d_{\text{min}} = \max_i \min_j d(i, j). \tag{3.2}$$

Once d_{min} is computed, a random geometric graph with $\phi = d_{\text{min}}$ guarantees that every node in the network will have at least one incident edge.

3.2.4 Small-World Networks

First proposed by Watts and Strogatz (1998), the small-world network model is an attempt to produce networks that exhibit excess clustering (or transitivity) and short average path lengths. A key observation is that small-world networks have properties that lie between those of regular (lattice) networks and random graphs.

Lattices are the basic building block of small-world networks. Lattice models lend themselves directly to increased clustering by increasing the coordination number. As described previously, the *coordination number*, K, of a lattice determines the number of nodes, sorted based on physical proximity, with which a node has connections.

To demonstrate that lattice graphs possess excess clustering, we consider the one-dimensional lattice and its corresponding coordination number. Given the description of the coordination number, each node in the one-dimensional lattice is connected to exactly

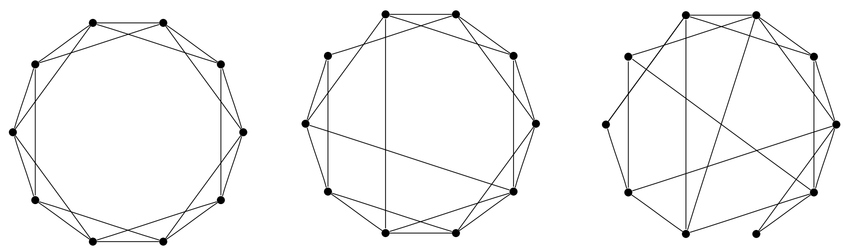
$2K$ other nodes, with the potential (i.e., if all nodes were connected to each other) for $K(2K-1)$ connections among the $2K$ neighbors. Due to the regularity of the interconnections in the one-dimensional lattice, there are actually $3K(K-1)/2$ connections among the nodes in the neighborhood of any given node. The clustering coefficient for each node is then the actual number of connections divided by the total number of possible connections:

$$C_i = \frac{3K(K-1)}{2K(2K-1)} = \frac{3(K-1)}{2(2K-1)}, \tag{3.3}$$

which converges to $3/4$ as K goes to infinity (Albert & Barabási 2002; Barrat & Weigt 2000; Newman 2003). Since every node in the one-dimensional lattice has the same local structure, the clustering coefficient of the entire graph, given by the average of all of the clustering coefficients of the individual nodes, is exactly C_i. Constructing a lattice in this fashion yields high clustering, but another mechanism is required to achieve short average path lengths.

The mechanism for decreasing average path length in small-world networks is a random re-wiring of a percentage of edges. This results in shortcut connections across the network, as seen in Figure 3.2. The parameter ρ is used to determine if an edge is replaced by a shortcut through the graph.[1] In this construction, ρ is the probability that each edge will be randomly rewired. When edges are replaced with random shortcuts with probability $\rho = 1$, the resulting graph is a random graph.

[1]One caveat in this construction of a small-world graph is that the graph can become disconnected.



(a) $\rho = 0.0$ (b) $\rho = 0.1$ (c) $\rho = 0.3$

Figure 3.2: Three increasingly random small-world networks: (a) a small world with no shortcut links; (b) the same small world with a few shortcuts; and (c) a small world with many shortcuts, which begins to resemble a random graph. All three of the networks are constructed from a one-dimensional lattice where nodes are connected to $K = 2$ other nodes in each direction, based on physical proximity.

3.2.5 Scale-Free Graphs

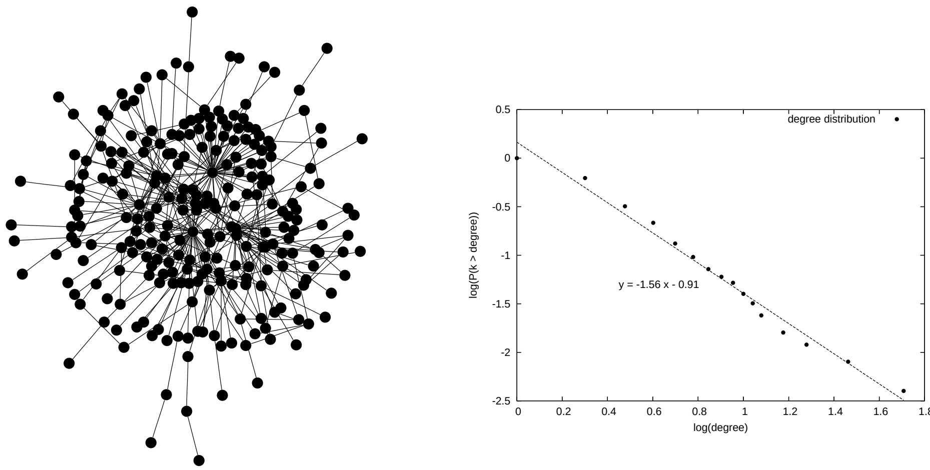
The scale-free graph model is motivated by the empirically measured degree distributions of the Internet and the World Wide Web (WWW) (Albert & Barabási 2002; Albert, Jeong, & Barabási 1999). The model is a highly intuitive model based on how networks are believed to evolve and grow in the real world.

The generation of scale-free graphs has two simple rules:

1. **growth:** at each time step, a new node is added to the graph, and
2. **preferential attachment:** when a new node is added to the graph, it attaches preferentially to existing nodes with high degree.

Preferential attachment is modeled by the equation:

$$P(e_{ij}) = mk_j \sum_{v \in V} (\frac{1}{k_v})^{\beta} \tag{3.4}$$



(a) (b)

Figure 3.3: An example of a scale-free network structure with 250 nodes: (a) a rendering of the network that clearly shows the hub-and-spoke structure, and (b) a log-log plot of the cumulative degree distribution of the network shown in (a). Note that a linear curve in a log-log plot implies a power-law behavior of the underlying system.

where i is a new node being added to the system and $P(e_{ij})$ is the probability of the creation of an edge between nodes i and j. As before, k_v is the degree of node v. There must be an initial connected core of m_0 nodes, ensuring that at the beginning of the graph generation process, nodes with non-zero degree exist. The model parameters are the number of expected initial connections that a new node will make, m, and a scaling factor for the probability of connection, $\beta \in [0, \infty)$, which forces the graph to be more or less dense. Note that when $m = \beta = 1$, the probabilities of connecting to existing nodes in the network sum to exactly one, so the expected number of edges in the graph is equal to n. Scale-free graphs exhibit both short average path lengths and skewed degree distributions but, like random graphs, lack excess clustering (Albert & Barabási 2002).

It is worth mentioning that the above sampling of models of network structures is just that: a sampling. There are many network structure models in the literature. This set of models was chosen because they are widely studied, highly intuitive, and either model the structure of real-world networks or have been traditionally used to model the interconnectivity of agent-based systems.

3.3 Statistical Measures for Dynamic Networks

Many possible measurements can be used to understand and characterize the behavior of dynamic networks. In this section, I introduce several measures that will aid in understanding the behavior of the agent-organized networks studied in later chapters. Some measurements are obvious, such as the number of edges in a network, the density of a network (i.e., the ratio of edges to the number of edges in the complete graph with the same number of nodes), and the number of components (disjoint subgraphs) in a network. I focus on measures derived from properties observed in real-world networks.

3.3.1 Mean Path Length

The *mean path length* of a network is the average shortest path between all pairs of agents in the network (Albert & Barabási 2002; Newman 2003). The mean path length of a network is given by

$$D(G) = \frac{1}{n(n-1)} \sum_{i} \sum_{j} d(i,j), \tag{3.5}$$

where $d(i, j)$ is the distance, possibly weighted, of the shortest path between agents i and j in the network. Note that $d(i, i) = 0$. With a slight abuse of terminology from the graph theory literature, mean path length will also be referred to as network *diameter*.

Measuring the mean path length of a network that is disconnected (i.e., that has more than one component) requires a slight modification to the normal notion of distance. It is normal to say that $d(i, j) = \infty$ if agents i and j have no path between them. Using this notion of distance results in $D(G) = \infty$ for all networks that have more than one component. In order to preserve some of the information about the existing paths in the network, an alternative is to assign $d(i, j) = n$ when agents i and j are not connected. This results in $D(G) \in [1, n]$, with the complete network having a mean path length of one and the network with n agents and zero edges having a mean path length of exactly n.

3.3.2 Clustering

The concept of clustering has been widely studied in real-world social, and other, networks. The intuition behind clustering is to measure the frequency of transitive relationships in networks (Newman 2003; Albert & Barabási 2002; Watts & Strogatz 1998).

There are two common measurements of clustering. The first measurement calculates the ratio of triangles in a network to the number of connected triples. This is given by

$$C_{\triangle}(G) = \frac{3 \times \text{number of triangles in the network}}{\text{number of connected triples in the network}}, \tag{3.6}$$

where the number of connected triples counts the distinct sets of two agents that are connected to a third agent (Newman 2003).

An alternative, which will be used to study agent-organized networks in later chapters, is *localized clustering*:

$$C(G) = \frac{1}{n}\sum_i \frac{2|E_k^i|}{k_i(k_i-1)}, \tag{3.7}$$

where E_k^i is the set of connections among the k_i neighbors of agent i, and k_i is the degree of agent i (Newman 2003). Note that $C(G) \in [0,1]$ where the complete network has clustering of one and the empty network has clustering of zero.[2]

3.3.3 Deviation in the Degree Distribution

Since degree distribution is one of the properties used to characterize and understand the structure of real-world networks, it is desirable to derive scalar network measurements that help capture the nature of a network's degree distribution. An obvious, and important, measurement is to calculate the network's average degree:

$$\bar{k} = < k_i > = \frac{1}{n}\sum_i k_i. \tag{3.8}$$

Although average degree captures the amount of connectivity in the network, it provides no information about the heterogeneity, or lack thereof, of degree among the agents in a network.

[2]Network structures other than the empty network can also have zero clustering. One example is the star topology.

Standard deviation is the common statistic for gleaning information about how a value varies over a distribution, and can therefore be used to partially characterize degree distribution. A slightly generalized form of standard deviation is the normalized standard deviation. This value is a more informative measurement than average degree when comparing various networks. The normalized standard deviation of degree of a network G will be denoted $NSD(G, k)$, where k refers to degree. This value is computed according to the formula (Zimmermann, Equiluz, & Miguel 2004):

$$NSD(G, k) = \frac{< k_i^2 > - < k_i >^2}{< k_i >}. \tag{3.9}$$

When $NSD(G, k)$ approaches zero for an evolving network, the network is moving toward a uniform degree distribution. An example of this will be seen in the discussion of the Symmetric Connections Model below (section 3.4.1).

3.3.4 Node (Degree) Correlation

Newman (2002) introduced a measure for assortativity, or degree correlation, of adjacent nodes in a network based on Pearson's correlation coefficient. We extend Newman's formulation to measure the correlation of any properties of adjacent agents in our networks:

$$\rho(x, y) = \frac{\frac{1}{m}\sum_{ij\in G} x_i y_j - \left[\frac{1}{2m}\sum_{ij\in G}(x_i + y_j)\right]^2}{\frac{1}{2m}\sum_{ij\in G}(x_i^2 + y_j^2) - \left[\frac{1}{2m}\sum_{ij\in G}(x_i + y_j)\right]^2}, \tag{3.10}$$

where x and y are vectors corresponding to properties of the agents in the network, $m = |E|$ is the number of edges in the network, and $ij \in G$ denotes that agents i and j are directly connected in the network G. As stated above, this measure can be used to calculate the correlation among any properties of adjacent nodes in a network. The degree correlation will be denoted $\rho(k, k)$.

Note that $\rho(k, k) \in [-1, 1]$, where a strong positive correlation means that neighboring nodes have similar degrees. A strong negative correlation means the opposite: high degree nodes are connected to low degree nodes. When there is little variance in degree, the correlation is close to zero.

3.4 Network Formation Games in Economics

Network formation has recently become a topic of interest in the economics literature. Although not directly applied to multi-agent organizational learning, economic network formation provides a foundation for thinking about agent-organized networks and gives some insight into theoretical results on endogenous network formation. In this section, a simple economic model of network formation is introduced along with a discussion of the concepts of stability and equilibria. I then briefly cover several of the major theoretical results for the network formation model. The theory is demonstrated, along with the statistical measurements discussed in the previous section, by the experimental results of a simulated dynamic network formation process.

3.4.1 The (Symmetric) Connections Model

First presented by Jackson and Wolinsky (1996), *The Connections Model* (CM) is a stylized model that is representative of a larger class of network games. In such games, values are given to network structures and to the positions of agents in the network. The CM was first developed in order to characterize and study the nature of "social communications among individuals" (Jackson & Wolinsky 1996).

In the model, agents directly communicate with the agents with whom they share an undirected edge (i.e., a connection) in the network structure. The communications imply value for information flow, so agents also benefit from the indirect communications represented by their neighbor's direct connections and their neighbor's neighbor's connections. The value of the indirect communications falls off as a function of the geodesic distance (i.e., shortest path distance) in the network. In particular, the value allocated to agent i in network G in the CM is given as

$$Y_i(G) = \sum_{j \neq i} \delta_{ij}^{d(i,j)} - \sum_{j:ij \in G} c_{ij}, \tag{3.11}$$

where $ij \in G$ denotes a connection between agents i and j in the network G, and $d(i,j)$ is the shortest path distance between i and j. The parameters δ_{ij}, with $0 < \delta_{ij} < 1$, are the values of the (possibly indirect) connections between agents i and j discounted as a function of the distance between the two agents. The parameters c_{ij} are the costs of direct connections between agents i and j. Agents benefit from being "close" to other agents, discounted by distance, while they only suffer costs for their direct connections. The value of a network in the CM is

$$v(G) = \sum_i Y_i(G), \tag{3.12}$$

which is commonly referred to as *social welfare* in economics.

The model allows agents to create or remove connections, with the goal of maximizing $Y_i(G)$. The model assumes that connections must be added bilaterally (i.e., the agents at both ends of the connection must agree to the connection), but that connections can be removed unilaterally. The *Symmetric Connections Model* (SCM) has homogeneous values for all of the connections: $\forall ij \in G\ \delta_{ij} = \delta$ and $c_{ij} = c$.

The CM and the SCM are representative of a larger set of network games that have been considered in the economics literature. Network formation has also been studied in the context of trade networks, labor markets, coauthor networks, and buyer-seller networks (Jackson 2003; Goyal 2003). Much of the economics literature is concerned primarily with stability, efficiency, and equilibrium, and does not concern itself with the real-time, dynamic behavior of agents in such models. Before considering the dynamic, real-time behavior of agents in the SCM, I first present the primary stability and efficiency results for the SCM.

3.4.2 Stability and Efficiency

As mentioned above, the economic literature is primarily concerned with stable and efficient structures in network formation games. The notion of stability captures whether or not agents in the network desire to make any *single* change to the network structure.

Definition 1 *(Jackson & Wolinsky 1996)* [3] *A network G is* **pairwise stable** *with respect to allocation rule Y if*

(i) $\forall ij \in G$*,* $Y_i(G) \geq Y_i(G - ij)$ *and* $Y_j(G) \geq Y_j(G - ij)$*, and*

(ii) $\forall ij \notin G$*, if* $Y_i(G + ij) > Y_i(G)$ *then* $Y_j(G) > Y_j(G + ij)$.

In the definition, $G - ij$ and $G + ij$ represent the removal of the connection between agents i and j and its addition, respectively.

Intuitively, pairwise stability implies that no agent desires to make any *one* modification to the connections in the network (i.e., no agent desires to add or delete a connection). The emphasis on a **single** modification, known in the game theory literature as *myopic*, is important since agent strategies that rely on the notion of pairwise stability can result in locally stable network structures (see below).

A pairwise stable network is a network for which no agent desires to change any one of its connections. Efficiency is a more strict notion.

Definition 2 *(Jackson & Wolinsky 1996) A network G is* **efficient** *if* $v(G) \geq v(G') \; \forall G' \in \mathcal{G}$.

Here, $\mathcal{G}$ is the space of all network structures of a particular size (i.e., the number of agents in the system). In essence, an efficient network has a value that is at least as great as *any*

[3] This definition is taken directly from Jackson and Wolinsky (1996) with a slight change of notation due to the assumption that Equation (3.12) gives the value function of the network.

other network structure.

Given these definitions of stability and efficiency, theoretical results shed light on the behavior of networks as they form. First, the concept of pairwise stability restricts the set of network structures that network formation processes settle on under different parameter regimes.

Proposition 1 *(Jackson & Wolinsky 1996) In the symmetric connections model:*

(i) A pairwise stable network has at most one (non-empty) component.

(ii) For $c < \delta - \delta^2$, the unique pairwise stable network is the complete graph, C^N.

(iii) For $\delta - \delta^2 < c < \delta$, a star encompassing all players is pairwise stable, but not necessarily the unique pairwise stable network.

(iv) For $\delta < c$, any pairwise stable network which is nonempty is such that each player has at least two links and thus is efficient.

Here, a star network is one with every agent connected to a central hub, and no other connections. The proof of the proposition is given in the original study (Jackson & Wolinsky 1996), but the intuition behind the proof is included here. When $\delta - \delta^2 < c < \delta$, the star is pairwise stable because (1) the hub would not delete any of its connections, because $\delta > c$; and (2) none of the other agents would form a direct connection, because for any two of the agents i and j: $d(i, j) = 2$, agent i gets the δ^2 indirect benefit via its connection with the hub, and $\delta^2 > \delta - c$ (where $\delta - c$ is the net cost of the new direct connection between i and j).

The second major theoretical result from the economics literature considers the structure of efficient network structures.

Proposition 2 *(Jackson & Wolinsky 1996) The unique efficient network structure in the symmetric connections model is*

(i) the complete graph, C^N, *if* $c < \delta - \delta^2$,

(ii) a star encompassing everyone if $\delta - \delta^2 < c < \delta + \frac{(N-2)}{2}\delta^2$, *and*

(iii) no links if $\delta + \frac{(N-2)}{2}\delta^2 < c$.

The proof of the proposition can be found in the original paper (Jackson & Wolinsky 1996) with the intuition for case (ii) presented here. The star is the unique efficient structure for two reasons: (1) it has the minimum number of connections, $(n-1)$, to guarantee a single component, therefore minimizing global cost; and (2) those connections are arranged so as to minimize the average pairwise distance between all of the agents (i.e., the star minimizes the mean path length, or diameter, of the network), maximizing the benefit to all of the agents.

The theory described here is extremely useful in understanding the structure of networks and network formation. Although simple, the SCM presents a challenging problem for distributed, multi-agent cooperations. Local decisions that increase local utility (i.e., greedy decisions) can be suboptimal considering future decisions and may lead to collective network inefficiencies. A main concern of this dissertation is in designing strategies (either designed or learned) that allow agents to make local decisions in real time as networks evolve in order to form efficient network structures in a variety of multi-agent settings. The SCM provides an interesting backdrop for analyzing agent-organized networks in more complicated domains.

3.4.3 A Dynamic Network Formation Process

Using the notion of pairwise stability, Watts (2001) proposed a dynamic model for the SCM (Jackson & Watts 2002). The dynamic model starts with an empty network (i.e., no connections), and then at each iteration, two agents, i and j, are chosen randomly from a probability distribution $p(i)$ and allowed to consider their connections (or lack thereof). This mechanism for considering connections will be referred to as the *random meeting* mechanism. In order to analyze the dynamic network formation process, assume that $p(i)$ is the uniform distribution over all of the agents in the network.

The deterministic dynamic network formation process (Watts 2001) is as follows. Let G represent the graph before i and j consider their connection. If $ij \in G$, the agents remove the connection if $Y_i(G-ij) > Y_i(G)$ **or** $Y_j(G-ij) > Y_j(G)$. If $ij \notin G$, the agents add the connection if $Y_i(G+ij) \geq Y_i(G)$ **and** $Y_j(G+ij) \geq Y_j(G)$, with the inequality holding strictly for at least one of i or j. That is, the agents establish the connection if it is mutually beneficial, and they remove the connection if either benefits from its removal.

In order to prevent the model from remaining in “uninteresting” stable states (e.g., the network with no connections when $c > \delta$), a stochastic dynamic network formation process was also considered. The stochastic model is the same as above, although decisions to establish connections are randomly inverted with probability ϵ. To liken this stochastic approach to reinforcement learning, it can be considered an ϵ-greedy exploration strategy. This was meant to represent slight irrationality or small “tremors” in the deterministic process (Watts 2001).

Note that the Watts formation process requires an individual agent making a local decision to perform a global computation. That is, the agent computes the change in value (which depends on the entire network structure) after adding or deleting a connection, and then determines if the change to the connection should remain or be reversed based on whether the value increased or decreased, respectively. This assumption, that agents have global knowledge of the entire network structure for informing decisions, is not realistic in many multi-agent environments. This is a key point in understanding the complexity of multi-agent network formation. In the next chapter, the SCM is revisited to highlight the challenges of distributed, multi-agent network formation even with the idealistic assumption of perfect global knowledge.

3.4.4 An Experiment with the Symmetric Connection Model

Two of the central questions in this dissertation are:

1. How can agents organize networks in a dynamic and distributed fashion?

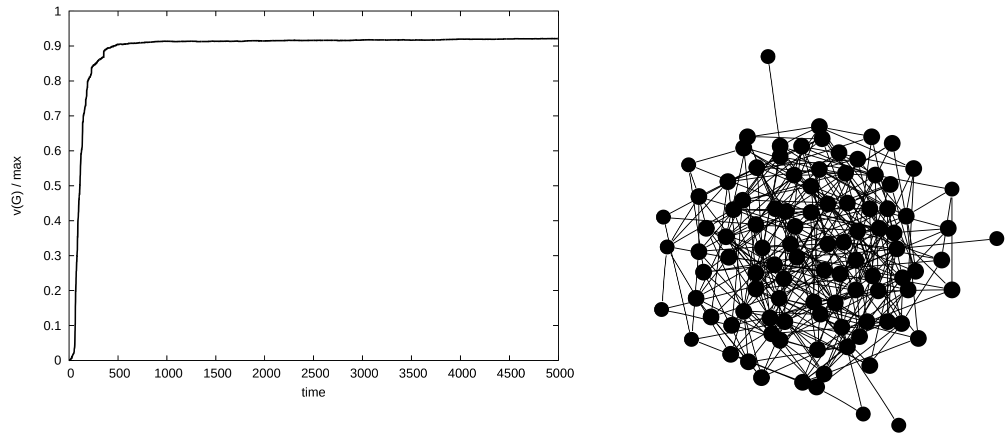
2. What are the structural properties of the networks that result from the distributed organization of networks by many agents?

In order to begin the examination of these questions and to introduce the concept of *agent-organized networks*, which will be treated more thoroughly in the next chapter, this section presents the results of operationalizing the Watts dynamic network formation process in the context of the SCM. Additionally, the results demonstrate the utility and applicability of the structural statistics for measuring changes in dynamic networks described above.

When the dynamic network formation process was first proposed, it was realized that forming an efficient and pairwise stable network (i.e., the star when $\delta - \delta^2 < c < \delta$) was difficult when the number of agents grows large.

Proposition 3 *(Watts 2001) Consider the symmetric connections model in the case where* $\delta - \delta^2 < c < \delta$*. As the number of players grows, the probability that a stable state (under the process where each link has an equal probability of being identified) is reached with the efficient network structure of a star goes to 0.*

The proof of the proposition is based on the fact that no agent wants to bear the burden of being the hub node in the star network. The hub node in a star network has a much lower



(a) (b)

Figure 3.4: Results of an experiment with the Watts dynamic network formation process for the SCM: (a) the normalized value of the network, $v(G)/v(\text{star})$, and (b) the network after 5000 iterations of the Watts process. The SCM had 100 agents, $\delta = 0.9$, and $c = 0.8$.

value than the other nodes in the network. The only way for the star network to form under the Watts dynamic network formation process is for the agent that will be the hub to meet every other agent sequentially, by chance, with no other nodes meeting each other until all have met the hub. It is obvious that the probability of this occurring goes to zero as the number of agents grows.

Given the finding that the unique, efficient, and pairwise stable network structure is very unlikely to be discovered, the question remains as to what network structures are found by the Watts dynamic network formation processes. A computational experiment using the Watts dynamic network formation process serves as a first example of an agent-organized network, demonstrates the evolution and structure of networks operating under the Watts process, and highlights the utility of the statistical measures described in the previous section. The results of this computational experiments are shown in Figures 3.4 and 3.5.

Figure 3.4(a) shows the normalized value of the network – $v(G)$ divided by the value of the uniquely efficient star network – over time as the agents use the stochastic dynamic network formation process of Watts (2001). In the experiment presented, there were 100 agents, $\delta = 0.9$, $c = 0.8$, and ϵ was initialized to 0.05 and slowly decreased over time. We experimented with a range of parameters in $\delta - \delta^2 < c < \delta$ and found similar results. The figure shows that the agents rapidly form a connected network and then the value of the network plateaus around 90% of optimal. The optimal value for the star network with $n = 100$, $\delta = 0.9$, and $c = 0.8$ is 7878.42. The structure of the network after 5000 iterations of the Watts formation process is depicted in Figure 3.4(b), which is obviously distant from a perfect star structure.

Recall that the Watts process is unlikely to find the optimal star network, because there is simply too much competition among the agents for one of them to become the hub. Even though all of the agents have access to perfect global knowledge of the network as it evolves, all of the agents are locally maximizing $Y_i(G)$. Nevertheless, the agents are able to find a network structure that supports an even distribution of value and a large network value.[4] Of course, this is expected since the agents are making perfect decisions ($1 - \epsilon$ percent of the time) with perfect, global information about the network structure.

[4]The sizes of the nodes in the network structure shown in Figure 3.4(b) are proportional to their value, $Y_i(G)$. There is little deviation in the sizes of the nodes in the network.

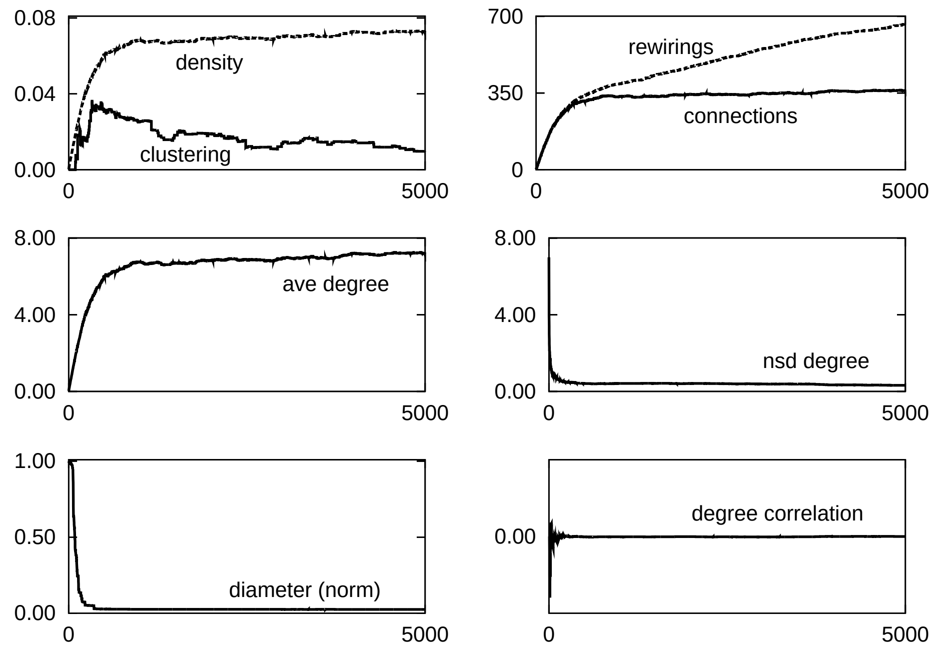


Figure 3.5: Measurements of the network structure taken during the evolution of the Watts dynamics network formation process for the SCM. The measurements help to understand the structure of the network as it evolves demonstrating the utility of the statistical measurements for understanding the behavior of agent-organized networks.

In order to better understand the structure of the network as it evolves under the Watts process, statistical measurements of the network structure, as described in the previous section, were taken at every time step. The results of these measurements are shown in Figure 3.5. As can be seen in the figure, the agents forming networks under the Watts process in the SCM form networks with low amounts of clustering and low diameter. Low clustering is desirable because the agents (locally) maximize their gain from having a direct connection, and do not duplicate connections that their direct neighbors maintain. Similarly, a small diameter, or short average path length, means that the agents in the network have short distances (i.e., greater benefit) to other agents. In addition, the figure shows that the number of connections in the network reaches a plateau as does the average degree. The values for normalized standard deviation of degree and degree correlation suggest that the Watts process leads to a nearly homogeneous degree distribution (i.e., all agents tend to have the same degree or bear the same cost in the SCM). The measurements taken in order to understand the behavior of the Watts dynamics network formation process agree with both the theory and intuition.

3.5 Concluding Remarks

Although simple, the SCM and the Watts network formation process provide an intuitive and insightful first example of an agent-organized network. In the next chapter, agent-organized networks will be more completely described, including many of the issues that are central to designing effective agent network adaptation policies.

The discussion of real-world network properties, models of network structure, and statistical measures for characterizing networks will aid in discussing and understanding the behavior of agent-organized networks in later chapters. The SCM is revisited in the next chapter in order to understand the complexity of computing optimal organizational structures and the difficulties of distributed network adaptation in multi-agent systems.

Chapter 4

Organizational Learning and Network Adaptation

> *Organizations learn only through individuals who learn. Individual learning does not guarantee organizational learning. But without it no organizational learning occurs.*
>
> Peter Senge

> *. . . [C]ollections of non-humans may come to seem more intelligent (i.e., show improved performance) even if the agents remain unchanged . . . if the connections among the agents are dynamically altered.*
>
> Kathleen Carley

An organization that learns is a group of agents that improve their collective performance based on experience. In this chapter, the concept of organizational learning through local network adaptation is further considered and explored. I first define organizational learning and discuss some of the major issues in using distributed network adaptation as a mechanism for organizational learning. After reviewing the literature on organizational learning in multi-agent systems, I consider the complexity of finding optimal and near-optimal network structures in multi-agent systems. Finally, I propose the concept of agent-organized networks and a general framework for designing local network adaptation strategies.

4.1 Organizational Learning and Network Structures

The *American Heritage Dictionary* defines the verb **organize** as "to put together into an orderly, functional, structured whole," and the noun **organization** as "the act of organizing or process of being organized." Additionally, **to learn** is defined as "to gain knowledge, comprehension, or mastery through experience or study." Using these definitions, it is possible to synthesize a definition of *organizational learning.*

Definition 3 **Organizational learning** *is the process of becoming an orderly, functional, structured whole through collective experience.*

The one additional word in this definition is "collective," meaning the joint learning of all of the actors, or agents, in the organization.

This dissertation is directly focused on organizational learning through the distributed adaptation of the underlying agent social network in a multi-agent system. That is, agents make local rewiring decisions about their connectivity with other agents in the organization, with the collective goal of improving organizational performance. Relating this method of organizational learning to the literature on dynamic organization and reorganization, distributed network adaptation may be categorized as *shared control*, or collaborative organizational structure change (Dignum, Dignum, & Sonenberg 2004). This type of organizational learning, the "automated formation and ongoing management of virtual organisations in open environments," is recognized as a major research challenge in multi-agent systems (Norman *et al.* 2004).

There have been many studies of how organizational structures, and in particular, network structures, affect the performance and behavior of multi-agent systems. Examples of these studies include the effects of network structure on information processing organizations (Carley & Gasser 1999; Carley 2002), the spread of social conventions in an agent society (Delgado 2002), distributed multi-agent team formation (Gaston & desJardins 2003; Gaston, Simmons, & desJardins 2004), the formation of firms (Axtell 2000), and graphical economic games (Abramson & Kuperman 2001; Holme *et al.* 2003; Kim *et al.* 2002; Szolnoki & Szabo 2004; Kearns, Littman, & Singh 2001). These studies demonstrate that the performance of a collection of agents is dependent upon the network structure that governs the interactions among the agents and that certain network structures perform better in certain settings. These two findings directly motivate the need for autonomous, distributed network formation in multi-agent systems.

In order to explore this type of organizational learning, in Section 4.4, I will introduce, define, and develop the concept of agent-organized networks. In the following subsections, I first consider some of the issues associated with distributed network adaptation, including local perception and learning, bottom-up versus top-down network formation, direct and indirect cost of connections, and inter-agent protocols for adapting connectivity. In section 4.2, I survey related work on organizational learning. In section 4.3, I provide theoretical results on the complexity of the design of organizational network structures that motivate the agent-organized network framework I propose in Section 4.4.

4.1.1 Local Perception and Local Learning

Part of the motivation for studying agent-organized networks is the need for decentralized adaptation in large, open multi-agent environments (Norman *et al.* 2004). In such environments, agents are unlikely to maintain or update information about the entire organizational structure (i.e., information about all other agents in the environment and their behaviors). Because of this, agents in large, open environments are required to make decisions based on limited local information, including decisions about when and how to adapt their network connectivity.

In addition to the local observability problem, local learning also presents a challenge because agents are likely to have both short- and long-range correlations. That is, the decisions made by agents in one part of the organization are likely to influence agents in nearby and far-away parts of the organization. As described in Chapter 2, multi-agent learning is a hard problem when there are few, or even two, simultaneously learning agents; the complexity is compounded by large numbers of learning agents who are coupled together in a complex organizational network.

4.1.2 Bottom-up vs. Top-down Network Formation

One distinction in decentralized network adaptation is whether the formation process is top-down or bottom-up. In *top-down network formation* one assume, initially, that all agents can interact with all other agents. Over time, in the top-down process, agents refine their respective sets of interactions, arriving at a trimmed-down organizational structure.[1]

In *bottom-up network formation* there are either no initial connections among the agents, or there is an initial network topology restricting the interactions of the agents. Recall the random mixture model described in the previous chapter in the context of the Symmetric Connections Model. In the random mixture model, there are initially no connections among the agents; at each iteration, two agents are randomly selected to meet and consider adding a new connection or removing an existing connection. While the random mixture model is reasonable when considering economic network formation, it may not be a reasonable model of network formation in many multi-agent systems.

A more realistic model for network formation in multi-agent systems is what I refer to as the *bootstrap model* of network formation. In the bootstrap model, the number of connections in an initial network topology represents the collective cognitive, communications,

[1]Of course, there is a full spectrum of network formation processes between pure top-down and pure bottom-up processes.

and resource constraints imposed on the organization.[2] The agents in the organization must "bootstrap" the initial network topology into a more efficient structure, given the initial constraints on the network. That is, the agents can only perform *rewiring* operations when adjusting their local connectivity (i.e., agents can only redirect connections in the network and are prohibited from either permanently deleting or creating new connections). The bootstrap model is more realistic for networked multi-agent systems where the size and complexity of the system prevents full and constant interaction among all of the agents. In such settings, agents with finite computational and cognitive capacities must constantly balance the use of resources for maintaining and utilizing their social connections. In an effort to explicitly include computational and cognitive constraints on the agents, I will use the bootstrap model of network formation to demonstrate agent-organized networks in various multi-agent environments.

4.1.3 On the Cost of Connections

A third consideration in distributed network adaptation is the cost of searching, changing, and maintaining connections. Again, recall the Symmetric Connections Model and the explicit cost associated with having a direct connection in the organizational network. This explicit cost may represent resources, such as time or money, spent in maintaining connections. In multi-agent systems, cost is more likely to represent resource constraints such as

[2] Another interpretation of the bootstrap model is to consider the initial set of connections as the constraints on individual agents.

computational, communications, cognitive, or memory limitations.

An alternative to explicit cost for maintaining network connections is implicit cost, such as in the bootstrap model of network formation described above. The collective cost of connectivity is represented by the limited set of connections in the initial network structure. While the agents suffer no direct, explicit cost for connectivity, the organization is collectively resource-bounded and must attempt to make the best use of the number of connections with which it is initially endowed.

In addition to the cost of maintaining network connectivity, the cost of searching and rewiring must also be considered. One method for representing the cost of searching and rewiring is to define an explicit cost. This approach is difficult because of the variety of multi-agent environments for which agent-organized networks are likely to apply. An alternative is to simply analyze the amount of searching and rewiring required for different network adaptation strategies in any specific environment. This latter approach is the one adopted in the study of agent-organized networks in this dissertation.

4.1.4 The "Laterality" of Connections

When considering agents that will autonomously or cooperatively form connections in networks, it is important to consider the concerns of the agents on both sides of the connection. This is true for the removal of a connection and the creation of a connection (or both in the case of rewiring). In the economics literature, it is typically assumed that connections must be established bilaterally, but connections can be removed unilaterally (Jackson 2003; Goyal 2003). That is, both agents involved in the creation of a connection must agree to establish the connection, while a single agent can decide to remove a connection.

In general, the rules for how connections are established or removed can be considered a social norm (i.e., dictated by the multi-agent environment, or by the creators of the environment). In the specific domains presented in later chapters, the nature of the creation and removal of connections will be specified. In practice, the economic model (i.e., bilateral creation and unilateral removal) is a good model. In some settings, various "laterality" models will be considered and compared.

4.2 Related Work on Organizational Learning

The importance of organization in multi-agent systems is a widely studied topic. Multi-agent system organization can be formal or informal; designed or emergent. In this section, I briefly survey several of the most widely studied topics in multi-agent organizations. With the exceptions of graphical games (Kearns, Littman, & Singh 2001) and peer-to-peer systems (Yu, Venkatraman, & Singh 2003), the role of networks, let alone network adaptation, is rarely studied. In the next several subsections I cover the topics of coalition formation, organizational self-design, graphical games, organizational evolution, and adaptive peer-to-peer systems and relate these topics to organizational learning via decentralized network adaptation.

4.2.1 Coalition and Congregation Formation

In cooperative problem-solving environments, coordination and negotiation among the many agents in a system are difficult. One solution to the challenges of coordination and negotiation in large multi-agent systems is *coalition formation* (Abdallah & Lesser 2004; Sandholm & Lesser 1997). The process of coalition formation assigns agents to, or allows the agents to form, smaller groups within which intra-group coordination and negotiation may be easier. The goal of coalitions of agents is then to solve cooperative problem-solving task. The process of coalition formation, especially when the formation is determined in a decentralized manner, can be thought of as a form of organizational learning.

There are at least two approaches to coalition formation in multi-agent systems: centralized and decentralized. It is well known that the problem of finding optimal coalition structures is NP-complete (Sandholm *et al.* 1998; Abdallah & Lesser 2004; Sandholm & Lesser 1997). Centralized methods for coalition formation include a central manager and auctioneer for determining coalition structures (Kraus, Shehory, & Taase 2003), bounded partial search for finding approximate coalition structures (Sandholm *et al.* 1998), and genetic algorithms for evolving near-optimal coalition structures (Sen & Dutta 2000).

As mentioned above, distributed coalition formation falls close to the realm of organizational learning as defined in the beginning of this chapter. One approach to distributed coalition formation is a hierarchical organization of agents where only the leaf agents participate in coalitions and the higher-level agents decide on coalition structures. The agents at higher levels in the hierarchy can use Q-learning, for example, to learn to determine the structure of effective coalitions (Abdallah & Lesser 2004). In the sensor network domain, incremental and fully distributed negotiations have been used to form coalitions for task completion (Sims, Goldman, & Lesser 2003). As a third example of distributed coalition formation, a physics-motivated protocol has been proposed, where agents move randomly in a physical space and make decisions regarding forming coalitions when they encounter one another (Lerman & Shehory 2000). Although distributed coalition formation can be thought of as organizational learning, there is not typically a notion of an "agent social network" in the study of coalition formation.[3]

An special case of coalition formation is partnership selection, in which agents work together, in pairs, to complete tasks. The approach is top-down: in the beginning, all agents can work with all other agents, with the selection of partners refined over time using a "simple reinforcement process" (Dutta & Sen 2003). An extension of this work attempts to incorporate both historical information and future expectations for partnership selection (Saha, Sen, & Dutta 2003).

In work closely related to coalition formation, reinforcement learning has been applied to dynamically select coordination mechanisms in multi-agent systems (Excelente-Toledo & Jennings 2003; Excelente-Toledo 2004). Like the partnership formation work, there is no social network restricting the interactions of the agents; rather, all agents are able to

[3]The hierarchical structure used by Abdallah and Lesser (2004) is a control structure that does not impose any restriction on which "leaf" agents can participate in coalitions together.

interact with all other agents and the coordination mechanisms for specific interactions are learned.

A related problem is *congregation formation* (Brooks & Durfee 2003). Much like coalition formation, in congregation formation, agents dynamically form groups in order to gain economic advantage or efficiency. A typical application of congregation formation is market formation (Brooks & Durfee 2002). In this setting, the benefits of congregations include greater efficiency in clearing markets (i.e., the congregations are necessarily smaller than the whole agent population) and greater commonality of interests among the agents within the congregations. As with coalition formation, congregation formation does not take into account an agent social network; the only structure of the organization is the individual congregations, within which it is possible for all agents to interact with one another.

4.2.2 Organizational Self-Design

Another form of organizational learning in multi-agent systems is *organizational self-design* (OSD) (Corkill & Lesser 1983; Gasser & Ishida 1991; Ishida, Gasser, & Yokoo 1992). The key tasks in OSD are monitoring, design, evaluation and selection, and implementation. OSD considers tasks, subtasks, agents, assignment of tasks to agents, work flow structure, and resources to be part of the organization. Much of the emphasis in OSD is on adaptive role allocation, task decomposition, and load balancing (So & Durfee 1993). One obvious approach to organizational self-design is a centralized, global, top-down designer (Corkill & Lesser 1983). An alternative is a decentralized, local, bottom-up approach (Gasser & Ishida 1991). In both cases, the main focus is on allocation and re-allocation of roles, decomposition and distribution of tasks, and distribution of load. In OSD, some structure may be imposed on the agents (e.g., a hierarchy), but the nature and constitution of that structure is not modified as part of the adaptive organizational design.

Applications of OSD include distributed network management (So & Durfee 1993) and distributed production systems (Ishida, Gasser, & Yokoo 1992). OSD has been considered as a mechanism for adapting organizational structure for *partial global planning* (PGP) (Durfee & Lesser 1991). In this work, various small network topologies were considered and determined to have a significant effect on the performance of the system. The networks were lateral (or flat – all agents can communicate with all other agents), one-level hierarchies (centralized), and two-level hierarchies. The focus was on selecting from a predetermined set of structural choices, since the effects of structure on organizational performance were a concern: ". . . as the number of agents increases, choices of organizational structures strongly impact the combinatorics of coordination" (Durfee 1993).

More recently, OSD has served as the basis for designing a general diagnostic subsystem using the TÆMS modeling language, which was demonstrated for load balancing and plan coordination in a producer-consumer-transporters domain (Horling, Benyo, & Lesser 2001). A similar approach was used to coordinate the organizational behaviors of sensors in a sensor network for target tracking (Horling *et al.* 2003).

4.2.3 Evolving Organizations

In the management, economics, and social science literature, the structure of an organization and its effect on organizational performance have been widely studied (Axtell 2000; Carley 1998; Miller 2001; DeCanio *et al.* 2000). In much of this work, the structure of an organization is modeled as a network that guides and restricts the interactions among the member agents of the organization. The observation that organizational structure affects performance (i.e., that some organizational structures are more effective than others) motivates the question of how organizations adapt and evolve in order to improve performance.

Surprisingly, much of the work on studying organizational adaptation and developing mechanisms for organizational learning through structural changes focuses on centralized heuristic approaches. Genetic algorithms have been used to evolve information processing organizations in order to find organizational structures that are more effective at disseminating information among the agents (DeCanio *et al.* 2000; Miller 2001). In a similar study, simulated annealing served as an analogy for CEO decision making in order to adapt organizational structure online, while agents in the organization continued to perform tasks (Carley 1998). In both cases, the heuristic search techniques successfully improved the performance of the organizations. While successful and insightful, these search techniques are inherently centralized, in contrast with the distributed agent-organized network approach presented and demonstrated in this dissertation.

4.2.4 Adaptive Networks and Game Theory

Related to both economics and multi-agent learning, graphical games use an explicit network structure (Kearns, Littman, & Singh 2001). In graphical games, agents that are embedded within a network structure play repeated matrix games (e.g., the iterated prisoner's dilemma (IPD)). While such games are interesting in their own right, they become even more interesting when the agents are allowed, or required, to adapt their connectivity along with their strategy.

Most of the work on game theory and adaptive organizational structure focuses on co-evolving cooperative strategies and the network topology that dictates the interactions among the agents.[4] An evolutionary rewiring algorithm (ERA), including the processes of replication and mutation, has been applied to networks of agents playing fixed strategies in the IPD (Hales & Edmonds 2005). Hales and Edmunds found that purely cooperative structures emerged due to network rewiring process even in the absence of strategy learning. In similar studies, leadership and hierarchical structures emerged from distributed network adaptation in an organization of agents playing the IPD (Zimmermann, Equiluz, & Miguel 2001; 2004; Anghel *et al.* 2004). In these studies, the network adaptations were mostly parameterized, random rewirings.

Finally, a top-down, "urn-based" reinforcement scheme was applied to partner selection in the stag hunt game (Skyrms & Pemantle 2000). "Top-down" means that initially all

[4] Perhaps surprisingly, most of this work is done by physicists.

agents could interact with one another and mutually beneficial partnerships were refined over time. In this study, there was no explicit concern for the network topology, since the focus was on improving the performance of the players.

4.2.5 Referral Networks, Peer-to-peer Systems, and Information Retrieval

One of the functions of peer-to-peer systems is distributed information retrieval. In such systems, agents are "peers" who can respond to or route the information request queries of other agents. Much of the work on peer-to-peer systems focuses on effective searching and routing algorithms for fixed peer-to-peer network topologies. The alternative to this approach is to allow agents in a peer-to-peer network to evolve their network topology.

There is a limited amount of work on network formation in multi-agent peer-to-peer systems, primarily focused on constructing networks that are efficient for distributed search. One approach starts with a random network and proceeds by allowing each agent to refine their neighborhood based on "context," or similar interests (Zhang *et al.* 2004a). In this "agent-view reorganization algorithm," network formation is performed before the distributed informational retrieval dynamics begin in the network and they do not analyze the resulting topology. In a second approach, a centralized monitor observes the network and performs "edge-thinning" and "diameter folding" operations, emphasizing various network properties in order to create a more efficient search topology (Silvey & Hurwitz 2004).

The majority of work on distributed adaptation of network topologies in peer-to-peer systems focuses on peer referrals and modeling the interests and expertise of other agents in the network (Yu & Singh 2003; Yu, Venkatraman, & Singh 2003; Yolum & Singh 2003; Ramanathan, Kalogeraki, & Pruyne 2002). The basic premise in this approach is that agents monitor their neighbors (i.e., direct connections) and their acquaintances (i.e., indirect connections that respond to their queries). As acquaintances begin to outperform neighbors (e.g., provide more and better information), the agents adjust their connections to change their acquaintances into direct neighbors. Note that the interests of agents are assumed to be explicit and observable by all other agents in peer-to-peer systems (e.g., the agents can learn other agents' interests by observing which queries are responded to by which agents over time). This assumption is one reason why network adaptation as a result of referrals improves information retrieval in peer-to-peer systems.

While referrals have been shown to improve information retrieval in peer-to-peer systems, the topologies resulting from these network adaptation strategies are rarely studied.[5] Another problem with some of the work on adaptive peer-to-peer networks is the lack of concern for the network becoming disconnected (i.e., multiple disjoint graph clusters). However, all of the studies to date have demonstrated that communities of agents with similar interests emerge in referral networks, making information retrieval more efficient even if the global network is not connected (Yu & Singh 2003; Yu, Venkatraman, & Singh 2003;

[5]The one slight exception is the inclusion of measurements of clustering in the study of thresholded promotion of acquaintances (Yu, Venkatraman, & Singh 2003).

Yolum & Singh 2003; Ramanathan, Kalogeraki, & Pruyne 2002; Zhang *et al.* 2004a).

Referrals and network adaptation in peer-to-peer systems can be considered a form of organizational learning in a multi-agent system. In the later sections of this chapter, these ideas will be incorporated into a more general concept of agent-organized networks applicable to a wider variety of multi-agent systems. Subsequent chapters will demonstrate agent-organized networks in several, innately different multi-agent domains.

4.3 On The Complexity of Finding Optimal Network Structures

Before describing in detail the notion of agent-organized networks, I consider the complexity of finding optimal, or near-optimal, organizational network structures. First, consider the peer-to-peer networks described in the previous section. In these systems, the interests of all of the agents are known and it is reasonable to assume that these interests do not change quickly. In such an environment, it is easy to imagine a centralized algorithm that computes the "interest" distance between all pairs of agents and then greedily assigns a predetermined number of connections among the agents, ensuring that each agent receives one connection, then two, and so on. The result of this algorithm would be a collection of communities of agents, possibly globally connected, where the agents within a community would share common interests. This organizational structure is similar to that found by the distributed network adaptation techniques based on referrals described in the previous section. Clearly, the centralized algorithm proposed here is polynomial in the number of agents.

While it is possible to describe a polynomial-time algorithm for finding a well-structured peer-to-peer system, the situation is more difficult for finding optimal network structures in arbitrary multi-agent domains. In this section, evidence will be presented that suggests that finding optimal network structures is computationally complex.

The argument begins by considering evidence given in previous work. Recall the process of coalition formation described above. It is well known that finding optimal coalition structures is NP-complete (Sandholm *et al.* 1998; Abdallah & Lesser 2004; Sandholm & Lesser 1997). While the problems of finding a network topology and finding a coalition structure are qualitatively different, there is an intriguing analogy. In finding an optimal coalition structure, an exponential number of combinations of agents must be considered, since all possible combinations of agents are searched. For finding an optimal network topology, all possible combinations of agents must also be considered, with the further complication of considering the interconnectivity of the agents within each combination. While this is not a proof, it is at least suggestive of the challenge of finding optimal network structures.

Further evidence that finding optimal network structures is hard is demonstrated by the methods used to find good network structures. Consider searching the space of all possible network structures of size n. This space is $O(2^{\frac{n(n-1)}{2}})$, or exponential in a polynomial of the number of agents.[6] Heuristic approaches such as simulated annealing (Carley 1998) and genetic algorithms (Miller 2001; DeCanio *et al.* 2000) have been used to evolve organizations in this very large search space. In two of the studies on evolving organizational network structures, it is suggested that finding optimal organizational networks is NP-complete, but no proofs of these claims are provided (Carley 1998; DeCanio *et al.* 2000).

Recall the Symmetric Connections Model discussed in Chapter 3, where the value given to agent i in the network G is

$$Y_i(G) = \sum_j \delta^{d(i,j)} - \sum_{j:ij \in G} c. \tag{4.1}$$

In this model, agents receive a benefit (e.g., access to information) for connections to other agents, discounted based on geodesic distance in the network G. While agents receive a benefit for all direct and indirect connections, they only suffer a cost for maintaining their direct connections (i.e., their incident edges). This model will be used to provide further evidence of the computational complexity of finding "optimal" network structures for multi-agent domains. The argument begins by defining an agent's *best response* to a fixed network structure.

Definition 4 *Given an arbitrary, fixed network structure, an agent's* **best response** *to the*

[6] A given network can be represented by a bit string where each bit corresponds to an edge either being in the network (1) or not being in the network (0). Since there are $\frac{n(n-1)}{2}$ possible connections in a network, there are $2^{\frac{n(n-1)}{2}}$ possible network structures.

existing network structure in the Symmetric Connections Model is the set of direct connections that will maximize the agent's value.

Now, consider the structure of an agent's response under certain parameters in the SCM.

Lemma 1 *In the Symmetric Connection Model with $\delta - \delta^2 < c < \delta$, the best response of an agent i to an arbitrary, fixed network structure G is the minimum number of connections such that $\forall j \in G : d(i, j) \leq 2$.*

Proof. Let agent i be the agent joining the fixed network. The value of a three-hop indirect connection in the SCM is δ^3. The value of a direct connection is $\delta - c$. For an agent's best response to contain a three-hop indirect connection, the value of a three-hop connection must be greater than the value of a one-hop connection, requiring

$$\delta^3 > \delta - c, \tag{4.2}$$

and therefore,

$$0 > \delta - c - \delta^3. \tag{4.3}$$

The parameters stated in the lemma require that $c > \delta - \delta^2$. By substitution, this would require

$$0 > \delta - (\delta - \delta^2) - \delta^3, \tag{4.4}$$

or

$$0 > \delta^2 - \delta^3, \tag{4.5}$$

which is a contradiction since $0 < \delta < 1$. Therefore, an agent joining a fixed network will always establish a direct connection whenever a three-hop indirect connection (or greater) exists. From the stated parameters, it is clear that an agent prefers a two-hop connection to a one-hop connection. Furthermore, because of the direct cost associated with a direct connection, the agent will minimize the number of direct connections. Therefore, the best response of an agent i to a fixed network topology is the minimum number of connections such that $\forall j \in G : d(i, j) \leq 2$. ■

As an example of Lemma 1, consider an existing network that is fixed in a star topology. A new agent's best response is to establish a single connection with the hub of the star network. This single connection to the hub guarantees that the distance to all other agents in the network is less than or equal to two.

The proof of Lemma 1 leads to the following theorem.

Theorem 1 *Computing the best response of an agent to an arbitrary, fixed network structure in the Symmetric Connections Model with $\delta - \delta^2 < c < \delta$ is NP-complete.*

Proof. The proof is by reduction to the minimum dominating set. The minimum dominating set D of a fixed graph G is the smallest set of nodes such that $\forall i \in G$ either $i \in D$ or $\exists j \in D : ij \in G$. That is, all nodes in a graph are either in the dominating set or adjacent to a node in the dominating set. By Lemma 1, finding the best response in the SCM when $\delta - \delta^2 < c < \delta$ is equivalent to finding the dominating set in a graph. The minimum dominating set problem is known to be NP-complete (Garey & Johnson 1979).

A solution to the minimum dominating set, and therefore, the best response of an agent to a fixed network structure, can be checked in polynomial time. The first step is to verify that all nodes are either part of the minimum dominating set or adjacent to a node in the set, which can be done in $O(n)$ time, where n is the number of nodes in G. Then, the solution can be verified to be minimal, by sequentially checking to see if any of the nodes in the solution can be removed from the set while still covering the graph. This second step can be computed in $O(sn)$ time where s is the size of the solution set with $s \leq n$. Therefore, finding the best response of an agent entering the network in the SCM when $\delta - \delta^2 < c < \delta$ is NP-complete. ■

The theorem shows that it is computationally complex to compute an agent's best response to a fixed network structure for a very simple domain, even when the entire network is completely observable. I conjecture that as the domain becomes more complicated and the observability of the agents is decreased, the difficulty of finding optimal, or even near-optimal, network structures is even harder.

4.4 Agent-Organized Networks

In this section, I introduce, define, and develop the concept of agent-organized networks (AONs). The section concludes with a proposed, general, learning-based framework for AONs.

Recall that the motivation for developing the concept of AONs is as a mechanism for organizational learning.

Definition 5 *An* **agent-organized network** *(AON) is an organizational network structure, or agent-to-agent interaction topology, that changes as a result of local network adaptation decisions made by the individual agents in a networked multi-agent system.*

In this definition, *networked multi-agent system* refers to a multi-agent system where there is an explicit agent social network that governs the interactions among the agents in the system. Note that although the definition of AON is not restricted to bottom-up, top-down, resource-constrained, or explicit-cost network formation, the remainder of this dissertation will focus on bottom-up, resource-constrained AONs.

The two major considerations in the development of AON strategies are design questions and evaluation measures. The design questions include when and how the agents will adapt their local connectivity structure. Evaluation measures include the examination of network structures as they evolve under certain AON strategies and measuring performance changes as a result of AONs. Before proposing a general AON framework, I discuss these considerations in more detail in the following subsections.

4.4.1 When to Adapt Local Connectivity?

Agents operating within an AON must be able to determine when to adapt their local connectivity. Deterministic, stochastic, and learned policies are all possible. An example policy may be to randomly decide to adapt local connectivity based on a prespecified probability.

There are many alternatives to random network adaptation decisions. For example, a *performance-based* policy uses accumulated historical performance information to decide when to adapt. An agent using such a strategy may decide to adapt when its performance drops prespecified threshold. An alternative would be for the agent to adapt when its local performance drops below its neighbors' performance levels. In the general AON framework presented below, performance is the key value in determining whether an agent wishes to change its connectivity or maintain its current position in the agent social network.

4.4.2 How to Adapt Local Connectivity?

There are many possibilities for determining how an agent will change its local connectivity in an agent social network. In some contexts, agents may choose to remove or create connections. However, because the emphasis here is on the bootstrap model of network adaptation, this section will focus on the rewiring of connections as the primary method of changing local connectivity. I start with several definitions.

Definition 6 *The* **neighborhood** *of agent i in network G, denoted $N_i(G)$, is the set of agents $\{j|ij \in G\}$. The agents in the set $N_i(G)$ are called the* **neighbors** *of agent i.*

Definition 7 *A* **rewiring** *by agent i in network G is the replacement of a single agent $j \in N_i(G)$ with an agent $k \notin N_i(G)$. Initially, $ij \in G$ and $ik \notin G$. The rewiring by i from*

j to k, denoted $ij \to ik$, is $G - ij + ik$. Agent j is considered the **losing agent***. Agent k is considered the* **gaining agent***.*

Note that a rewiring affects the neighborhood of the agent that is performing the rewiring as well as the neighborhoods of both the losing agent and the gaining agent. Clearly, rewiring does not change the total number of connections in an agent social network.[7]

One form of rewiring is *random rewiring*, where an agent i randomly selects a current neighbor $j \in N_i(G)$ with whom to sever its connection. Agent i then randomly selects a non-neighbor agent $k \notin N_i(G)$ with whom to establish a new connection. Random rewiring seems like an unrealistic solution in many multi-agent environments, since it implies the existence of a centralized agent registry or central broker. However, random rewiring is a useful baseline for the experimental evaluations of AONs present in later chapters.

Another, more realistic, candidate for a rewiring strategy is referrals. The following definition is a generic definition for a referral.

Definition 8 *A* **referral** *rewiring by agent i in network G is the establishment of a new connection between agent i and an agent*

$$k \in \bigcup_{j:ij \in G} N_j(G) - N_i(G), \tag{4.6}$$

and the subsequent removal of a connection between agent i and an agent in $N_i(G) - k$.

[7]Maintaining a constant number of connections preserves the resource, cognitive, and communications constraints imposed by the initial network topology in the bootstrap model.

The order of operations in the definition of referral is important, since the referral may come from the losing agent. In practice, referrals are taken from a neighbor with particular properties, who will nominate one of its neighbors based on particular properties. As a naive example, one type of referral might be based on agent degree (i.e., the agent's number of connections). In such a referral, an agent deciding to adapt may remove its connection with its lowest-degree neighbor, and take a referral from its highest-degree neighbor, based also on degree. The criteria for selecting agents for referrals will almost always be tailored to a specific multi-agent environment. While referrals are both intuitive and realistic in large, open multi-agent systems, they should be used with caution.

Proposition 4 *An AON operating solely under referral rewirings can result in the network becoming permanently disconnected.*

Proof. Consider a network in which the agents can be divided into two disjoint groups. The agents in each group are only connected to other agents in their respective group, with the exception of a single pair of agents, one from each group, that share a single connection. This network is connected. Let the pair of agents with the single connection across groups be i and j. If i removes its connection and takes a referral from an agent other than j, the network is divided into two disjoint subnetworks. This proves that it is possible for referrals to result in disconnected networks. Once a referral disconnects a network, it is impossible to reconnect, since referrals are based on connectivity. Agents taking referrals from other agents in their group will never establish a connection with an agent from the other group;

therefore, once a network is disconnected under referrals, it will remain disconnected. ■

This proposition suggests that one must be careful when designing agent adaptation strategies for AONs. Agents that adapt based only on referrals may lead to disconnected agent social networks, which may be a detrimental occurrence in various multi-agent environments. For example, disconnected networks are viable in peer-to-peer informational retrieval domains, as long as the disconnected components share common interests.

Although referrals can lead to permanently disconnected networks, refinements of the referral concept can guarantee that a connected network will remain connected.

Definition 9 *Let $j \in N_i(G)$. A* **push referral** *by agent i in network G is the establishment of a connection with an agent $k \in N_j(G) - N_i(G)$ and the subsequent removal of the connection between agent i and agent j.*

A push referral is a special case of referral. To illustrate push referrals, imagine going to a store to buy a certain product. Upon learning that the store is out of the product of interest, a reasonable action is to ask the store clerk for a recommendation of a nearby store that might have the product of interest in stock.

Proposition 5 *An AON that begins with a connected network and evolves as a result of push referral rewirings alone will always remain connected.*

Proof. Consider a single push referral where agent i is rewiring its connection with agent j. Two cases exist: the removal of the connection between i and j (1) does not disconnect the network, or (2) disconnects the network. The first case is trivial. In the latter case, the removal of the connection from i to j breaks the network into two components (one containing i and the other containing j and all of the agents in $N_j(G)$). Therefore, when i makes the new connection with one of the agents $k \in N_j(G)$, the network is again connected. Finally, since a single push referral cannot disconnect the network, neither can a sequence of push referrals. ■

Referral-based adaptation strategies are appealing because they only require local information. Agents are not required to have knowledge of other agents beyond their immediate neighbors and there is no requirement for a central broker or manager. Additionally, agents are not required to maintain a memory of past interactions, although maintaining such a memory is not prohibited.

Another alternative type of adaptation is based on memory, or acquired knowledge, of other agents in the system. Agents may “remember” other agents with whom they have successfully interacted with in the past and adapt their network structure according to this acquired knowledge. As an example, consider agents forming teams in a network, where the team must be a connected component of the larger network. In such a setting, agents participate in teams with agents beyond their set of direct connections. This situation would allow the agents to keep track of successful “indirect” interactions through working together on teams. A possible network adaptation approach, given this team formation scenario, would be for agents to change their connectivity based on successful teamwork. This model of team formation will be considered in the next chapter.

Finally, in a *hybrid rewiring* strategy, an agent would mix any of the approaches described above (and possibly other strategies). One example of a hybrid strategy would use general referrals 90% of the time and random rewirings 10% or the time. Recall the proposition that states that general referrals can lead to permanently disconnected networks. Interestingly, the simple 90/10 hybrid strategy described above would not suffer from the potential for perpetual disconnectedness. The random component of this hybrid strategy would allow a disconnected network to eventually become reconnected.

4.4.3 Stability and Performance

The primary goal of AONs is increased organizational performance. In most cases, measures of organizational performance will be tailored to the specific domain of the AON. When measuring organizational performance, it is important to distinguish between, or control for, performance gains from the internal dynamics of the agents in the system and performance gains from network adaptation. In many networked environments, agents can learn or improve performance without adapting the network structure. Finally, in some situations, measuring the performance of individual agents or the fairness across the agents can help assess the utility of AONs.

Another issue associated with AON performance is stability. Taking from the economic theory of network formation, a *stable* network is one that stops evolving after reaching some operating point. That is, in a stable network, no agent desires to make any changes to its local connectivity. The concept of stability raises several interesting questions related to AONs:

- Is stability a desirable property of AONs?[8]
- Are stable AONs optimal, locally optimal, or neither?
- What are the stable networks under certain rewiring regimes in AONs?

General answers to these questions are beyond the scope of this dissertation, but partial answers to these questions in the context of specific multi-agent domains will be considered in Chapters 5, 6, and 7.

4.4.4 A General Learning-based AON Framework

In this section, I present a general learning-based AON framework. Later, in Chapters 5, 6, and 7, I apply the general framework to develop domain-specific AONs.

The general AON framework is based on stateless Q-learning (Claus & Boutilier 1998), in which the agents learn the value of taking actions online as they experience the results of their actions. The function for updating the action value function for each agent is

$$Q(a) \leftarrow (1-\alpha)Q(a) + \alpha R_t, \tag{4.7}$$

[8]It may be more desirable for an AON not to be stable, but rather to be continuously flexible.

or equivalently,

$$Q(a) \leftarrow Q(a) + \alpha[R_t - Q(a)], \tag{4.8}$$

where R_t is the "immediate" reward for taking the action a and α is the learning rate. When α is large, the agent learns fast; when α is small, the agent learns slowly. The basic premise is that agents will always choose the action that has the maximum value over all actions. The design of reward R_t is domain-specific and should be tailored to reflect the behavior as a result of the most recent action.

Extending stateless Q-learning to AONs requires defining the action set. If the AON will adapt using rewirings alone, the action set is $A_R = \{rewire, nothing\}$. Including the *nothing* action makes the general AON framework a multi-agent learning framework, since the *nothing* action allows an agent to monitor performance changes that result from the actions of other agents. Although it appears simple, this general AON framework can lead to a rich variety of behaviors.

By updating the value of doing nothing (i.e., taking no action), agents can "free ride" on the behavior of other agents in the network. If agents in other parts of the network (i.e., recall the discussion of long-range correlations in Section 4.1.1) are making changes that are beneficial, an agent using the general AON framework will continue to allow these beneficial changes to occur. At the same time, if decisions by other agents in the network are detrimental to an agent, the value of doing nothing will decrease, therefore triggering local network adaptation. While "doing nothing" seems like a useful approach, it can also be of great concern in complicated or excessively competitive domains. It is not difficult to imagine a scenario where network adaptation improves the local performance of the agent performing the adaptation, but decreases the performance of all other agents in the network (i.e., a form of the tragedy of the commons). In this scenario, one agent's changes cause another agent's changes and a ripple effect, or infinite regress, of adaptation ensues (see Section 2.1.2).

As long as the "do nothing" action remains, it is possible to extend the action set to include other types of network adaptations. If an AON allows for the removal and addition (i.e., more than just rewiring) of network connections, the action set can be extended to $A_S = \{add, delete, nothing\}$. This is the action set for agents in the Symmetric Connections Model described in Chapter 3.

Another extension of this framework is to include state information, allowing the agents to keep track of the value of taking certain actions *in certain states*. Because the state space of networks is large (i.e., 2^n states, where n is the number of agents), it may not be possible to have state-action values for all possible combinations of states and actions. Because of this, state abstractions are likely to be necessary when extending the stateless Q-learning approach. An example of abstraction for AONs is for the agents to represent their state to be their current number of connections. In the context of the Symmetric Connections Model, an agent may want to remove connections when it has many, but may not want to remove connections when it has few. In this setting, the agent's actions are selected based on its current "state."

Another extension to the basic framework is to allow for an adaptive learning rate. With a fixed learning rate, agents might over-compensate by adding more connections, or rewiring, more often than is desirable. This is especially true for small learning rates. To counter this behavior, one approach, based on the *Win or Lose Fast (WoLF)* (Bowling & Veloso 2002) concept, is to have two learning rates: one for when the agent receives positive reinforcement and one for when the agent receives negative reinforcement. That is, the agent learns slowly and cautiously when improving its performance, so as not to over-improve. Alternatively, the agent learns rapidly when performance is decreasing, in order to recover quickly from poor action selection.

Finally, a similar approach to assigning values to actions can be used to determine the value of specific connections in an agent's local neighborhood. An agent may use an exponentially weighted moving average, such as

$$V_{ij} \leftarrow V_{ij} + \beta[W_{ij} - V_{ij}], \tag{4.9}$$

where β is a smoothing, or learning, rate and W_{ij} is the current value of the connection from i to j. The design of W_{ij} must be domain-specific. If an agent uses this approach for tracking the value of its connections, it may use a heuristic to select connections for removal or rewiring.

The general AON framework is meant to serve as a template for designing AONs for specific multi-agent environments. The focus of the framework is on determining *when* an agent should adapt. While many generic strategies exist for changing the local connectivity, it is likely that domain-specific network adaptation rules will be required. These rules can be coupled with the generic framework to provide AONs for specific environments.

4.5 Concluding Remarks

The need for organizational learning in multi-agent systems is clear. One form of organizational learning is distributed network formation based on local decisions of the agents in an organization. This is motivated by the observation that the structure of an organization of agents can have a dramatic effect on its collective performance.

In this chapter, after reviewing the literature on organizational learning, the concept of agent-organized networks was introduced and developed. AONs are a proposed mechanism for supporting organizational learning through distributed, decentralized network formation. The two major considerations in the design of agent strategies for AONs are deciding when to adapt connectivity and deciding which connections to change. The design of AONs will be domain-specific for most multi-agent systems.

Finally, a general framework for the design of agent strategies for agent-organized network was presented. The framework is based on stateless Q-learning from multi-agent learning. The key to the framework is endowing the agents with the ability to detect when other agents are making network connectivity decisions that are beneficial (or detrimental). This allows agents to benefit from the actions of other agents. Additionally, a mechanism for approximating the expected utility of a connection was suggested.

In the next several chapters, AONs will be designed and evaluated in various multi-agent environments. First, two general multi-agent domains are considered, followed by the application of AONs to two, more-specific, networked multi-agent systems. For the general domains, several AONs with various levels of complexity will be considered.

Chapter 5

Networked Multi-Agent Team Formation

> *People tend to work in teams, in a collaborative way, in an informal network. If you create an environment like that, it's much more effective and much more efficient.*
>
> Jim Mitchell

Teams and teamwork are core problems in the study of multi-agent systems. There are many areas of application for the theory of multi-agent teams and teamwork, ranging from multi-robot systems, such as robotic soccer teams (Dias, Browning, & Veloso 2005), to agent-mediated e-commerce (He, Jennings, & Leung 2003; Norman *et al.* 2004). In much of the work on multi-agent teams, it is assumed that all of the agents in a system know about and can interact with one another. With multi-agent systems moving to increasingly large and complex domains, enabled by technologies such as the Semantic Web, peer-to-peer networks, and Grid technologies, the agents in such systems will be unable to continuously interact with all other agents in the system. In such systems, the agent-to-agent interactions will be governed by an implicit or explicit social network within which the agents will need to dynamically form teams and collectively achieve joint tasks.

In this chapter, I apply agent-organized networks (AONs) to a general, distributed team formation environment in which the agents are organized in an explicit social network. The agents in the system must dynamically form teams without the intervention of a centralized broker or decision maker. I begin the chapter by briefly reviewing the literature on teamwork and team formation. In Section 5.2, a generic model of team formation for joint task completion is discussed, and the effects of network structure on dynamic, distributed team formation are considered. Several AON strategies are developed for the team formation environment and experimentally evaluated in Section 5.3.

5.1 Overview of Multi-Agent Team Formation

Work on multi-agent teams can be divided into the process of team formation and the dynamics of teamwork. Studies of multi-agent teamwork dynamics focus on the behavior of multi-agent teams after they have formed. These studies address coordination and cooperation protocols among the team members and are largely focused on individual and joint cognitive capacities and behaviors (Cohen, Levesque, & Smith 1997; Grosz 1996; Tambe 1997). On the other hand, studies of the formation process focus on the agents' reasoning about how or when to join coalitions. Many methods have been applied to this problem, including optimization and approximation methods for identifying near-optimal coalition structures for a set of tasks coupled with an agent organization (Abdallah & Lesser 2004; Caillou, Aknine, & Pinson 2002; Sandholm & Lesser 1997).

There has been a significant amount of research by the multi-agent systems community on team formation and self-organization. Task allocation, coalition formation, team formation, and self-organization are different terms that have been used in studying how tasks can be allocated to agents. Much of the work on team formation focuses on the mental states of the agents and their willingness to form teams and collaborate (Cohen, Levesque, & Smith 1997; Wooldridge & Jennings 1999). Several researchers have applied these theories to implement frameworks in which teams coordinate closely to develop and execute distributed plans (Tambe 1997; Durfee & Lesser 1991). In this work, there is no explicit concern for the connectivity of the agent societies within which team formation takes place.

In distributed task allocation, or role allocation, many methods have been applied to assign tasks to individual agents in multi-agent teams. These methods include distributed constraint satisfaction (Modi *et al.* 2001; Nair, Tambe, & Marsella 2003), combinatorial auctions (Hunsberger & Grosz 2000; Sandholm 1999), market-oriented methods (Walsh & Wellman 1998), and distributed communicating partially observable Markov Decision Processes (POMDPs) (Pynadath & Tambe 2002; Nair, Tambe, & Marsella 2002). Applications of teamwork theories have been extended recently to address very large agent teams (Liao, Scerri, & Sycara 2004). This work on very large teams includes an information exchange network among the agents for effective coordination and information diffusion, but does not address how teams are formed in the large agent society (Scerri *et al.* 2004). The work on role allocation in multi-agent teams primarily focuses on assigning subtasks to agents within already formed teams. While role allocation within agent teams is closely related to our work, agents must first form teams in order to allocate roles. The process of team formation, and re-formation under agent failure, has been addressed from a formal perspective (Nair, Tambe, & Marsella 2002) without explicit concern for agent social network structures.

As agents are embedded in larger and more diverse environments, the practicalities of limited communication and spatial orientation may prohibit all agents from interacting with one another at all times, therefore creating a much more sparse social structure among the agents and a challenging distributed team formation environment. So and Durfee present theoretical results indicating that organizational structure can affect the ability of a collection of agents to form teams (So & Durfee 1996). Here, I demonstrate the effects of the network structure on organizational team formation performance and I show that agents with the ability to adapt their local network connectivity can improve organizational performance.

5.2 A Model of Team Formation in Agent Networks

To explore the effects of network structures and AONs on team formation processes, I use a simple multi-agent organizational model motivated by previous work on agent team formation (Nair, Tambe, & Marsella 2002; Abdallah & Lesser 2004; Kraus, Shehory, & Taase 2003; Gaston & desJardins 2003; Gaston, Simmons, & desJardins 2004; Gaston & desJardins 2005). In this model, tasks are generated periodically and globally advertised to the organization. Agents attempt to form teams to accomplish these tasks. The agents in the organization are embedded in an artificial social network that restricts the set of possible agent teams: specifically, for an agent to be on a team, the agent must have a social connection (i.e., an edge in the social network) with at least one other agent on the team. Since I am only concerned with the formation process, tasks are generic in that they only require that a team of agents with the necessary skills form to accomplish the specific task.

The goals of the model are to provide a dynamic team formation environment in which agent teams form spontaneously in a completely decentralized manner and the agents' decision making is based solely on local information. In the team formation model:

- agents are not subject to failures (Nair, Tambe, & Marsella 2002), since I am focused on the dynamics of team formation,
- teams must form in real time, as opposed to episodic optimization for task alloca-

tion (Abdallah & Lesser 2004),

- the agents are embedded within an artificial social network that can be very large and configurable; it is not limited to a specific hierarchical structure (Abdallah & Lesser 2004), and

- the model does not restrict agent team joining strategies to any particular bidding or negotiation protocol (Kraus, Shehory, & Taase 2003).

The model is only concerned with the dynamic formation of teams and does not address teamwork mechanisms or protocols, for which there is a large body of previous work (Modi *et al.* 2001; Nair, Tambe, & Marsella 2003; Hunsberger & Grosz 2000; Sandholm 1999; Walsh & Wellman 1998; Pynadath & Tambe 2002; Nair, Tambe, & Marsella 2002; Tambe 1997).

5.2.1 The Model

In the team formation model, the organization consists of n agents, where each agent can be considered as a unique node in an agent social network. The social network is modeled as an adjacency matrix E, where an element of the adjacency matrix $e_{ij} = 1$ if there is an edge between agent i and j and $e_{ij} = 0$ otherwise. The social relationships among the agents are undirected, so $e_{ij} = e_{ji}$. The number of connections in the agent social network is denoted $e = |E|$. In the model, every agent is connected to itself (i.e., $e_{ii} = 1$ for all agents). Each agent is also assigned a single fixed skill, $\sigma_i \in [1, \sigma]$, where σ is the number of different types of skills that are present in the organization.

During the team formation process, each agent can be in one of three states: UNCOMMITTED, COMMITTED, or ACTIVE. An agent in the UNCOMMITTED state is available and not assigned to any task. An agent in the COMMITTED state has selected a task, but the full team to work on the task has not yet formed. Finally, an agent in the ACTIVE state is a member of a team that has fulfilled all of the skill requirements for a task and is actively working on that task. Only uncommitted agents can commit to a new or partially filled task.[1] Committed agents cannot decommit from a given task. Upon task completion, agents in the active state return to the uncommitted state. The state of agent i is s_i.

Tasks are introduced at fixed task introduction intervals, where the length of the interval between tasks is given by the model parameter, μ. Tasks are globally advertised (i.e., announced to all agents). Each task T_k has an associated size requirement, $|T_k|$, and a $|T_k|$-dimensional vector of required skills, R_{T_k}. The skills required for a given task T_k are chosen uniformly from $[1, \sigma]$. Each task is advertised for a finite number of time steps $\gamma|T_k|$, ensuring that the resources (i.e., agents) committed to the tasks are freed if the full

[1]While it is a strong assumption to assume that agents can only be either active or committed to a single task at any time, it is realistic to assume that the agents have some resource constraints limiting their ability to be on any number of tasks. The model could be easily extended to allow agents to simultaneously participate on multiple teams. One such method would be to duplicate the agents, including their network connectivity in the organization.

requirements of the task cannot be met. Similarly, teams that successfully form to fill the requirements of a given task are only active for a finite number of time steps $\alpha|T_k|$.

The agent social network explicitly restricts the sets of agents that can form teams.

Definition 10 *A* **valid team** *is a set of agents $M = \{a_i\}$ that induce a connected subgraph of the agent social network and whose skill set $\{\sigma_i\}$ fulfills the skill requirements for a given task T_k.*

The requirement of a team to induce a connected subgraph of the agent social network means that for some agent in the team, $i \in M_k$, there must exist at least one other agent, $j \in M_k, i \neq j$, such that $e_{ij} = 1$. This implies that an uncommitted agent is only eligible to commit to a task in two situations: (1) *team initiation*, when no other agents are committed to the task, and (2) *team joining*, when at least one neighbor of the agent is already committed to the task. There are many possible heuristics for initiating and joining teams, but in order to focus on network adaptation, two simple strategies are used.

During each iteration of the model, the agents are selected in a random order to update. Each agent in the UNCOMMITTED state in turn considers each task in a random order. If a task currently has no other agents committed to it, an agent can choose to initiate a team, and does so with a probability equal to the proportion of the agent's immediate neighbors that are currently in the UNCOMMITTED state. The probability that an agent i initiates a team for a task to which no agents are currently committed is the initiation probability IP_i, where

$$IP_i = \frac{\sum_{j=1}^{n} e_{ij} I(s_i, \text{UNCOMMITTED})}{\sum_{j=1}^{n} e_{ij}}, \tag{5.1}$$

and $I(x, y)$ is an indicator function that returns 1 if $x = y$ and 0 otherwise. If an agent is eligible for a team, it joins a team with a joining probability, JP_i, equal to the ratio of filled positions, including itself, on the team:

$$JP_i = \frac{|M_k| + 1}{|T_k|}. \tag{5.2}$$

Note that agents can only be committed to, or active on, one team at a time. Figure 5.1 gives the pseudocode for the *JoinTeam* algorithm used for each agent. The algorithm combines team initiation and team joining. I have selected a simple team joining strategy to ensure that the benefits of network adaptation are truly a result of network adaptation. The problem of developing or learning effective team initiating and team joining policies is also important (Bulka, Gaston, & desJardins 2005), but is beyond the scope of this dissertation.

The team formation performance of the agent organization is measured as the ratio of number of teams successfully formed to the total number of tasks introduced to the system:

$$\text{organizational performance} = \frac{\text{\# of teams successfully formed}}{\text{\# of tasks introduced}} \tag{5.3}$$

This measure of performance provides a global measure of how effective the agent organization is at forming teams to execute the advertised tasks. Each agent's local performance is

$$Y(a_i) = \frac{\text{\# of successful teams joined}}{\text{\# of teams joined}}, \tag{5.4}$$

which is an estimate of organizational performance. This estimate will generally be positively correlated with true global performance, but for a particular agent, the correlation may be low, or even negatively correlated.

Extending the model to include network adaptation, during each iteration, the agents either attempt to join teams *or* adapt their local network structure (but not both). I present the details of the network adaptation strategies below.

Next, three important phenomena are introduced to aid in understanding the behavior of the team formation process embedded in various network structures. One is an emergent property of the dynamics of the model; the other two are characteristics of the network structures within which the agent organizations are embedded.

Blocking. [2] A phenomenon I call *blocking* can occur in the team formation model when multiple tasks are introduced into the agent organizations within close temporal proximity. This phenomenon is illustrated in Figure 5.2 for a simple graph. In the figure, the list on the right corresponds to tasks and their skill requirements. Each node is labeled with its corresponding skill. Notice that if the shaded nodes form a team to meet the requirements of the first task, teams attempting to form to complete the second and third tasks are blocked by the agents working on the first team. This phenomenon of blocking is central to understanding the performance of the various networked organizations.

[2] This is different, but similar to, the social pathologies (Jensen & Lesser 2002) discussed in Chapter 2.

Algorithm 1: *JoinTeam*

input:
i: an agent with $s_i = \text{UNCOMMITTED}$,
$T = \{T_1, T_2, \ldots\}$: the set of current tasks,
$M = \{M_1, M_2, \ldots\}$: the set of teams, where
  M_k is associated with T_k,
$R = \{R_{T_1}, R_{T_2}, \ldots\}$: the skill requirements for tasks in T
E: the adjacency matrix of the agent social network
begin
  for all $T_k \in T$ in random order
    if $|M_k| = 0$ **and** $s_i = \text{UNCOMMITTED}$
      with probability IP_i // see equation (5.1)
        if $\exists r \in R_{T_k} : r = \sigma_i$
          $M_k \leftarrow M_k \cup \{i\}$
          $s_i \leftarrow \text{COMMITTED}$
        end if
    else if $\exists j : e_{ij} = 1, j \in M_k$ **and** $s_i = \text{UNCOMMITTED}$
      if $\exists r \in R_{T_k} : r = \sigma_i$ **and** r is unfilled
        with probability JP_i // see equation (5.2)
          $M_k \leftarrow M_k \cup \{i\}$
          $s_i \leftarrow \text{COMMITTED}$
      end if
    end else if
  end for all
end

Figure 5.1: The algorithm used for each agent to decide which teams to initiate and which teams to join.

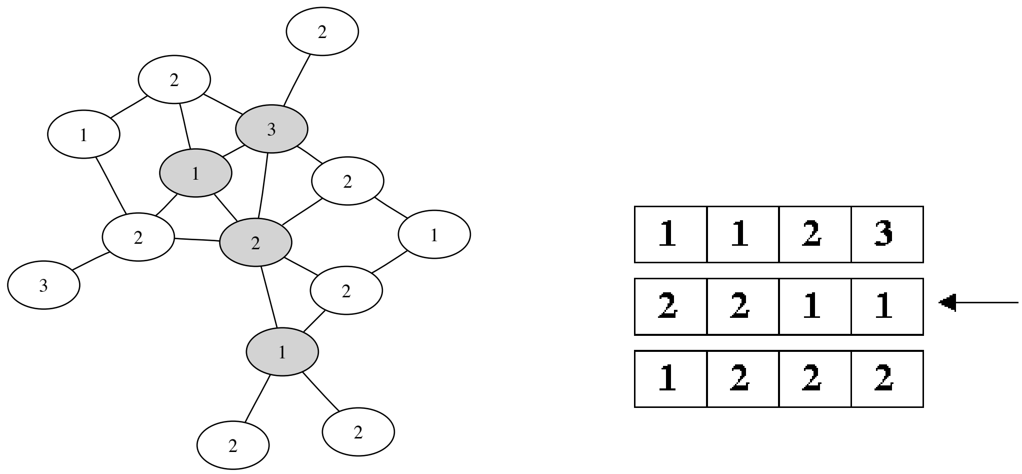


Figure 5.2: This figure shows an example of blocking in this team formation model for a notional graph. The list on the right represents tasks and their respective skill requirements.

Blocking is a general phenomenon that can be the result of either an insufficient number of skills in the organization or the “unreachability” of skills in the network. Figure 5.2 illustrates the latter case. The skills to simultaneously support the three tasks exist in the organization, but the position of the team for the first task prohibits teams from forming for the other two tasks. While both types of blocking are important, any discussion of blocking will focus on the notion of unreachability to understand the dynamics of team formation in networked systems.

Carrying Capacity. The number of simultaneous teams supported by a given network structure affects organizational performance. Although it is difficult to compute carrying capacity directly for many network structures, the concept of carrying capacity can help to illustrate and analyze the behavior of the team formation process, particularly in simple graphs.

Definition 11 *The* **carrying capacity** *of an organization of agents is the maximum number of simultaneously active teams that can be supported by the agent organization.*

The carrying capacity of networks ranges from 0 (e.g., a graph with no edges) to $n/|T|$, where n is the number of agents in the organization and $|T|$ is the size of tasks (assuming the task size is fixed). Carrying capacity is directly related to blocking: once the carrying capacity of a network is met, blocking will occur for all additional tasks until some active team completes a task. This type of blocking is a result of the organization having a finite size. Blocking as a result of unreachability occurs when the carrying capacity has not been met.

Diversity Support. The number of different skill combinations supported by an agent organization also has a direct impact on organizational performance. The number of different skill combinations is based on the network structure and on the skill assignments of the agents. In the experiments below, each agent's skill is selected from a uniform random distribution, as is each skill needed for the generated tasks.

Definition 12 *The* **diversity support** *of an organization of agents is the percentage of all possible skill combinations that is supported by the agent organization.*

The diversity support is the ratio of skill combinations supported by an agent organization to the number of all possible skill combinations that could be required by a task. This percentage depends directly on the number of skills in the model (σ) and the size of teams ($|T|$). Note that tasks are generated by creating a combination of skills *with replacement*, meaning that a given task can require more than one agent with a particular skill. Diversity support is related to the likelihood of a team forming for an arbitrary task. Like carrying capacity, diversity support is difficult to compute for many network structures. In the next section, carrying capacity and diversity support are used to analyze two basic network structures and to emphasize the fact that network structure dramatically affects organizational performance in networked multi-agent team formation.

5.2.2 The Effects of Network Structure

In many multi-agent systems, it is assumed that all agents are capable of interacting with one another. While this is a valid assumption for many domains, as multi-agent systems move toward open and large environments, it will quickly become impossible for all agents to interact with all other agents all of the time. In such scenarios, there will be an agent social network that governs the direct interactions of the agents. In this section, I examine the effects of organizational network structure on multi-agent team formation. The section begins by considering various characteristics of simple, static organizational structures, then I present an empirical study of how organizational team formation performance is dependent upon organizational network structure.

Simple Static Structures

While it is difficult to formally analyze the behavior of the team formation process for agent organizations embedded in complex network structures, it is possible to do so for simple networks. Here, analytical results for two simple networks—the star and the ring—are presented in order to understand the behavior of the team formation process and to support the hypothesis that the social network structure of an agent organization has a significant impact on its team formation efficiency.

The two networks considered are the star and the ring, each with n nodes and $n - 1$ edges. Figure 5.3 shows examples of these two graph topologies for $n = 10$ nodes. The ring is a one-dimensional lattice with $K = 1$. The star topology consists of a single hub node with edges connecting the hub to each of the $n - 1$ remaining nodes. Part of the motivation for selecting these two types of structures for analysis is that they have the same number of nodes and edges and therefore do not introduce performance differences due to different densities. The choice is further motivated by the fact that these two networks represent the boundary cases of the networks considered in the empirical studies to follow. The ring is a simple one-dimensional lattice. The star is an extreme form of a scale-free graph, with a skewed degree distribution and a hub structure.

The total number of possible tasks that can be generated serves as the starting place for

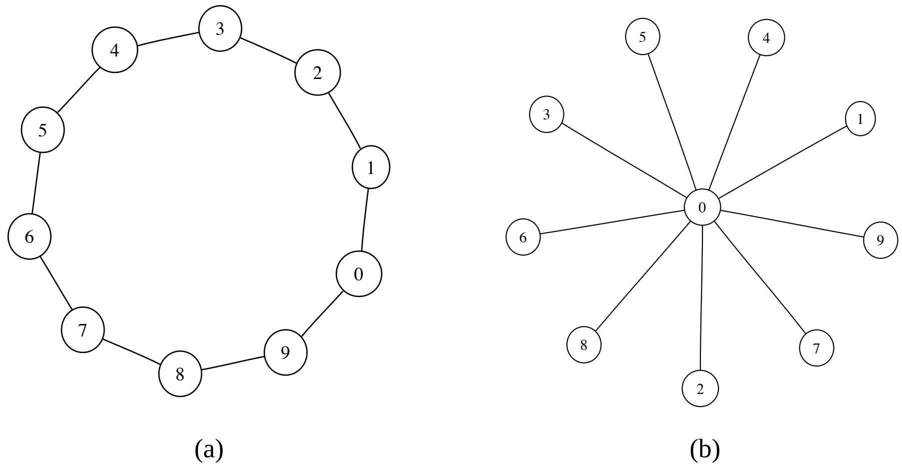


Figure 5.3: Examples of ring and star networks.

analyzing the simple network structures.

Proposition 6 *Let σ be the number of possible skills and let t be the task size $|T|$ (i.e., the number of skills for each task). Then the total number of distinct skill combinations that can be associated with a task is:*

$$\Omega(t, \sigma) = \begin{pmatrix} t + \sigma - 1 \\ t \end{pmatrix}.$$

Proof. The skills for a task are chosen uniformly at random from the interval $[1, \sigma]$ with repetition. Therefore, the number of distinct skill assignments is the number of t-combinations of σ skills with repetition. ■

Given that each of these tasks is equally likely, it is possible to compute bounds on the diversity support (i.e., on the probability that a random task can be supported by each of the simple networked agent organizations).

Proposition 7 *Let $t < n$ be the fixed task size for the team formation process. Then, an upper bound on the diversity support of the ring of n nodes is n/Ω.*

Proof. The diversity support is the percentage of all tasks that are supported by the agent organization, restricted by the network topology. Consider the team formed by starting with any agent and following a path of $t-1$ edges in one direction around the ring. This construction leads to n unique teams of t agents (one team for each starting agent). Some of these teams may not have distinct skill sets. (Consider the extreme case where all agents have the same skill: in this case, the n teams only have one distinct skill set.) However, if the skills are optimally assigned for greatest diversity, then there are at most n different combinations of skills that the organization can support. It follows that n/Ω is an upper bound on the diversity support of the ring of n agents. ■

Proposition 8 *Let t be the fixed task size and σ be the number of skills. If $n \geq \sigma t$, then the diversity support of the star with n nodes is*

$$\frac{t}{t+\sigma-1}.$$

Proof. When $n \geq \sigma t$, there may be up to t of each of the σ skills in the agent organization. For any team in the star topology, the hub (or central) agent must be a member of the team. For the remaining team members, any of the remaining agents can be selected. In the set of these agents, each of the skills is represented at least $t-1$ times. Therefore, to count the number of different skill combinations that are supported by the star, I count the number of $t-1$ combinations of σ skills with repetition. The percentage of all possible skill combinations supported by the star is this number divided by Ω. Therefore, the diversity support of the star with $n \geq \sigma t$ is

$$\begin{aligned} \frac{\binom{(t-1)+\sigma-1}{t-1}}{\binom{t+\sigma-1}{t}} &= \frac{\frac{(t+\sigma-2)!}{(\sigma-1)!\,(t-1)!}}{\frac{(t+\sigma-1)!}{t!\,(\sigma-1)!}} \\ &= \frac{(t+\sigma-2)!\,t!\,(\sigma-1)!}{(\sigma-1)!\,(t-1)!\,(t+\sigma-1)!} \\ &= \frac{t}{t+\sigma-1}. \end{aligned}$$

■

The diversity support of a network structure can be interpreted as the probability that an agent organization of uncommitted agents *can form* a valid team to accomplish a given task based on its requisite skills. Obviously, diversity support has a direct impact on the team formation performance of an agent organization. Carrying capacity also affects an agent organization's team formation performance. Recall that carrying capacity is the total number of simultaneous teams that the agent organization can support.

Proposition 9 *Let $1 < t < n$ be the fixed task size. Then the carrying capacity of the ring of n nodes is $\lfloor n/t \rfloor$.*

Proof. Teams must form such that the agents (nodes) in a team induce a connected subgraph of the network. The carrying capacity is the maximum number of simultaneous teams supported by the network. Consider the first task of size t requiring a team to form. Since I am interested in the maximum number of simultaneous teams, I select an arbitrary (independent of skill) initial node to initiate the team and allow the team to form along the edges going in one direction from the initial node. Once the team has formed, there remains a connected set of $n - t$ agents that are uncommitted. For the next task, I repeat the above process, starting from the node adjacent to the last member of the previous (and now active) team, leaving $n - 2t$ connected uncommitted agents. This process continues until there are $\lfloor n/t \rfloor$ active teams leaving fewer than t remaining uncommitted agents. ■

Proposition 10 *Let $t < n$ be the fixed task size for the team formation process. Then the carrying capacity of the star with n nodes is 1.*

Proof. For a single team to form to accomplish a task, the team must induce a connected subgraph. In the star topology, the single hub node must be on any team in order for the team to induce a connected subgraph. If the hub node is active, no other nodes can form a team because they cannot induce a connected subgraph without the hub. Therefore, the maximum number of simultaneous tasks for the star with n nodes is 1. ■

Consider a simple example of these two networks and a team formation environment with $n = 100$ and $t = \sigma = 10$. In such an environment, the number of skill combinations is $\Omega(10, 10) = \binom{19}{10} = 92,378$. Table 5.1 shows the calculated values of diversity support and carrying capacity for the star and ring networks in the team formation environment.

With a slight modification to the model, it is possible to assume that 10 tasks arrive at the same time and any of those tasks that acquire a complete team will be completed before the next 10 tasks arrive. This simplifying assumption is made in order to calculate

	carrying capacity	diversity support
ring	10	$\frac{100}{92{,}378} = 0.001$
star	1	$\frac{10}{19} = 0.526$

Table 5.1: The values of carrying capacity and diversity support for the static ring and star network topologies in the team formation model when $n = 100$ and $t = \sigma = 10$.

the performance of the two simple static network structures. A secondary simplifying assumption is that teams form instantaneously and that the diversity support of the ring is constant, even if the ring has active or committed agents. In a dynamic scenario, this assumption does not hold, but this ideal case will be used to show the significant difference in performance over various network structures of the same size. Note that an analysis using this assumption will overestimate the performance of the ring.

First, consider diversity support. Diversity support can be interpreted as the probability that a given task can be fulfilled by the agents in a specific organizational structure. It can therefore be used to calculate the expected number of tasks that will be completed in any batch of ten tasks in the example. Since the diversity support of the star is 0.526, the expected number of satisfiable tasks (i.e., skill combinations) that can be fulfilled in a batch of 10 is 5.26. For the ring, the diversity support is much lower, yielding an expected number of supported tasks of 0.01.

Carrying capacity also affects organizational performance. Although the star can support on average 5.26 tasks in any batch of ten, its carrying capacity limits the number of tasks that can be completed to one, the maximum number of simultaneous tasks. Because of its diversity support, the star is able to support on average more than half of the tasks, but only one out of every batch of ten will be successfully completed. On the other hand, the carrying capacity of the ring does not limit the number of simultaneous tasks that can be completed in this example; all ten tasks in a batch would “fit” in the ring organization. The limiting factor for the ring is its diversity support, which results in 0.01 tasks out of every batch of 10 being completed.

Now, the number of tasks successfully completed can be used to compute an estimate of organizational performance. Assume that a total of 100 tasks, 10 batches of 10, are introduced to both of the organizations. Following the analysis above, the star will successfully complete one task out of each batch and 10 tasks total out of the 100 that are introduced. The ring-structured organization will complete on average 0.01 out of every batch of ten tasks. Therefore, summing over the 10 batches of 10, the ring will successfully complete 0.1 tasks out of 100 introduced. Using the formula for organizational performance (i.e., number of successfully completed tasks divided by the total number of tasks introduced), the organizational performances for the star and the ring are 0.1 and 0.001, respectively.

In this simple example, the organizational performance of the star dominates that of the ring. The two networks are the extreme cases: the ring is limited by the lack of diversity support and the star is limited by its carrying capacity, although the limitations of the ring’s lack of diversity support is far greater. Clearly, modifications to the ring to increase its diversity support would be beneficial. Likewise, the star would gain significant benefits from additional carrying capacity, especially to leverage its high diversity support. Next, I experimentally examine the behavior of several complex organizational network structures in the team formation model.

Complex Networks

The previous section provided theoretical evidence that the dynamics of team formation can vary greatly over different network structures, even simple network structures. Here, the results of computational experiments provide further evidence that network structures have a dramatic effect on the team formation performance of agent organizations.

Figure 5.4 shows the average organizational performance over 50 simulations of the networked team formation model for four different network structures as a function of the task introduction interval μ. The networks are parameterized so that they all have the same number of agents and connections, $n = 100$ and $e = 400$, ensuring that all of the networks make use of the same number of resources. The four network types are:

- regular two-dimensional lattices, where each agent is connected to the agents immediately to its north, south, east, and west,

- small-world networks based on the two-dimensional lattice, with rewiring probability $p = 0.05$,

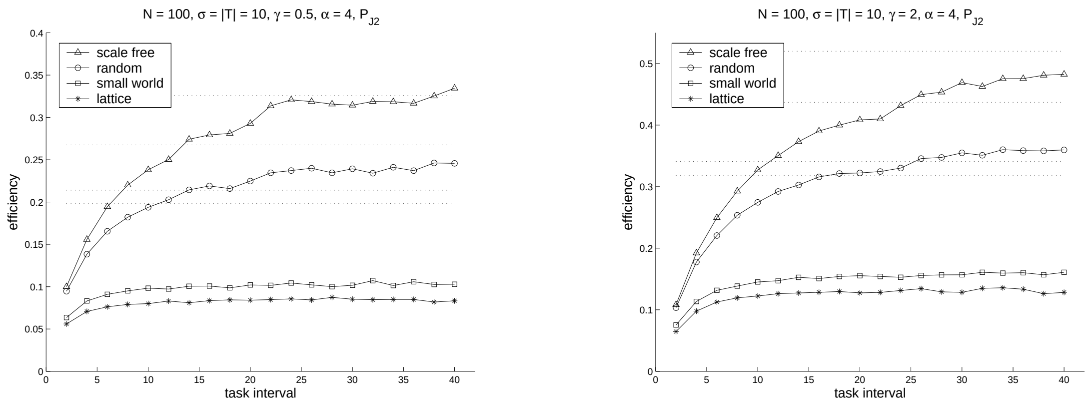


Figure 5.4: Organizational efficiency as a function of task introduction interval μ for the team formation model with $n = 100, |T| = \sigma = 10, \gamma = \{0.5\text{(left)}, 2.0\text{(right)}\}, \alpha = 4$. The horizontal lines represent the average efficiency of the network structures assuming that agents can be active or committed to any number of teams (i.e, no blocking), ordered from the top down as scale-free, random, small-world, and lattice.

- random graphs where connectivity (i.e., a single component) is verified, and
- scale-free graphs.[3]

The figure is a sample of the experimental results that support the claim that network structure has a dramatic impact on team formation. The results of these experiments are in line with the simple example considered above. The scale-free network is most like the star topology in the example, and the lattice network is most like the ring topology. The other two networks included in this section are in the region between the two extremes cases. This suggests that scale-free networks have higher diversity support and that lat-

[3]For more details, see Chapter 3.

tice structures, although possessing high carrying capacities, fail to capitalize on it because of a lack of diversity support. The dramatic effect of network structure on organizational performance in multi-agent team formation directly motivates the use of agent-organized networks in team formation domains.

5.3 Agent-Organized Networks for Team Formation

This section demonstrates the use of the general AON framework discussed in the previous chapter for the team formation environment. This section begins by incrementally improving on a purely random AON, arriving at an intelligent and effective learning-based AON. Following an experimental demonstration of these AONs, alternative AON strategies are discussed along with additional experimental results.[4] All of the AONs discussed in the context of multi-agent team formation are *resource-constrained* (i.e., the networks are limited to a fixed number of connections and only rewiring adaptations are allowed).

5.3.1 From Random to Intelligent AONs

Recall that the three parts of an AON strategy are

1. deciding *when* to adapt connectivity,
2. deciding *which* connection(s) to remove (rewire), and

[4]Previous work on rule-based AON strategies that use performance and structural information contains information on additional alternatives (Gaston & desJardins 2005).

3. deciding *where* to make new connection(s).

Since the multi-agent team formation model has no explicit cost of connectivity, only rewiring adaptations are considered. For simplicity, the assumption is made that each agent can only rewire one connection during any single iteration of the model, although the framework does not require this assumption. Additionally, only agents in the UNCOMMITTED state are allowed to rewire and an agent that decides to rewire during an iteration is prohibited from either initiating or joining teams during that iteration. An AON strategy will be referred to using the shorthand *when/which/where*, following the three parts of any AON strategy.

An obvious baseline for AONs is a purely random strategy. In such a strategy, there is no inherent intelligence, and at a minimum, a more intelligent AON should outperform the purely random AON. In a random AON strategy, an agent decides to adapt connectivity based on a prespecified probability, randomly selects a connection to rewire (without regard for the agent on the other end of the connection), and establishes a new connection with an agent randomly selected from the agent organization, with the exception of prohibiting multiple connections between the same two agents. This strategy will be referred to as *random/random/random*. Obviously, this strategy assumes that every agent "knows of" all other agents or that there is a capability to randomly generate the name, location, or address of all of the agents in the organization. While these assumptions are questionable in many multi-agent system environments, particularly large and open environments, the *random/random/random* strategy will be used as a baseline AON.

In order to understand how adding intelligence to AON strategies improves performance, several incremental modifications can be made to the random strategy in order to arrive at a more realistic AON strategy. First, following the general AON framework, stateless Q-learning with an adaptive learning rate replaces the random decision of when to adapt. For team formation, the action set is simply $\{rewire, nothing\}$ and the agent updates its Q values by

$$Q(a) \leftarrow Q(a) + \alpha[R_t - Q(a)], \quad (5.5)$$

where R_t is the **change** in local performance over a fixed number of iterations after taking a certain action. For the team formation model, in order to accumulate enough local information about performance, the Q values are updated after 100 iterations. As such, no actions (including the *nothing* action) are taken during the 100 iterations. The value of 100 iterations was established through trial-and-error experimentation. Finally, this strategy employs a variable learning rate with $\alpha = \alpha_{\text{max}} = 0.4$ when R_t is negative (i.e., learn quickly when performance is decreasing) and $\alpha = \alpha_{\text{min}} = 0.05$ when R_t is positive (i.e., learn slowly when performance is increasing).[5] Using this decision strategy, an agent decides to adapt its connectivity if $Q(rewire) > Q(nothing)$ and otherwise maintains its

[5]Other values for α_{min} and α_{max} were considered. The behaviors were similar as long as the agents learned sufficiently quickly when performance was decreasing and sufficiently slowly when performance was increasing. Setting $\alpha_{\text{min}} = \alpha_{\text{max}}$ generally resulted in poor AON performance, since the agents were unable to learn quickly enough to stop rewiring after it was no longer valuable.

current network connectivity. This strategy will be referred to as *Q/random/random*, since decisions about which connection to remove and which connection to add remain random.

The next incremental improvement is in the way an agent decides which connection to rewire. Perhaps the most obvious approach, and the approach taken here, is for each agent to maintain a value for each of its connections. Then, upon deciding to adapt, the agent selects the minimum-valued connection for rewiring. The value of a connection from agent i to agent j is maintained using an exponentially weighted moving average:

$$V_{ij} \leftarrow V_{ij} + \beta[W_{ij} - V_{ij}], \tag{5.6}$$

where W_{ij} is one if i and j are on a successful team completed during the current iteration and zero otherwise. The values of connections are updated at every iteration. Using the exponential weighted moving average for updating the values of connections, when an agent decides to adapt, it can rewire the existing connection with lowest value. This prevents an agent from continuously rewiring the same connection and from dropping connections that are highly beneficial. The value of a new connection (after rewiring) is set to one. The AON strategy that uses the stateless Q-learning approach for deciding when to adapt, the exponentially weighted moving average for removing minimally valuable connections, and the random policy for establishing a new connection is referred to as *Q/minNeighbor/random*.

Continuing with the incremental improvements from the basic random strategy, the final step is to add intelligence to the way new connections are established. There are many possibilities, several of which will be discussed later. Here, an intuitive approach is adopted as a first pass at creating an intelligent AON strategy. First, the agent will use a push referral to determine an agent with whom to establish a new connection. Recall that a push referral from agent i to agent j requires that j has decided to adapt and has decided to rewire its connection with i. Subsequently, agent i tells agent j where to establish the new connection. As described in the previous chapter, there are several benefits to push referrals including the lack of a need for a central broker and the guarantee of organizational connectivity.

As a starting point for this strategy, the agents will give push referrals based on the maximum-performance of their neighbors. That is, an agent providing a push referral will deterministically refer its neighbor with the largest local performance estimate.[6] Therefore, the AON strategy that employs maximum performance push referrals and the other incremental improvements is *Q/minNeighbor/pushMax*.

Complexity and Information Requirements

The complexity and information requirements vary over the four AON strategies proposed above. The purely random AON strategy requires no information to be stored about other agents or actions. As AONs are incrementally improved, the information requirements increase. These increased information requirements include the values of actions and the values of connections. Additionally, the more complex AONs assume some level of communication of values among directly connected agents.

[6] Of course, this assumes that the agents can and do communicate their own estimates of local performance.

The search complexity of the various AONs follows the opposite ordering. Push referral-based AONs only search over the neighborhood of a single node. Referral-based AONs search over the neighborhood of all neighbors. Finally, the random AONs can be interpreted as searching the entire set of agents in the organization.

As previously stated, there are many possibilities for determining how agents will establish new connections, and there are even many possibilities for how to provide push referrals. Additional approaches will be considered in Sections 5.3.3, 5.3.4, and 5.3.5.

Now that AONs with various levels of complexity and intelligence have been presented, the next step is to experimentally evaluate the behavior of these AONs and the structures that the organizations tend towards when the agents employ the various AON strategies.

5.3.2 Experimental Results

The experimental analysis of AONs in the team formation domain begins with comparing and contrasting the performance and structural trends of the four strategies described in the previous section. The basic experimental setup for understanding the behavior of AONs is to fix a set of parameters for the team formation environment and compare various AON strategies in this fixed-parameter setting. The result presented in this section are representative of the behavior of AONs under other parameter regimes, although no claim about the general applicability of specific AON strategies is implied.

Static Baseline

The experimental evaluation begins by measuring the average team formation performance of a multi-agent organization on a sample of **fixed** network structures from a particular class of networks. The parameters of the team formation model used in these experiments are: $n = 200, e = 300, \mu = 2, |T| = \sigma = 10, \gamma = 1, \alpha = 2$. This parameter regime represents a heavily loaded task environment intended to be a challenge for networked, distributed team formation (i.e., high potential for blocking, etc.) (Gaston & desJardins 2005). The findings presented below were found to be experimentally similar for other parameter regimes for heavily loaded team formation environments.[7]

The class of fixed network structures used in these experiments is random graphs. The type of random graph used is a slight variation on the Erdos-Renyi model of random graphs where the networks are guaranteed to be connected (i.e., have a single component). Random graphs were chosen as the base network type for experimentation since they are essentially structureless, but have properties that help in heavily loaded team formation environments. For example, the lack of locality in random graphs provides high diversity support (i.e., a wide variety of potential agent and skill mixes that can form valid teams in

[7] The claim of similar results for other parameter settings does not hold under varying network density (i.e., when the organizational network gains or loses connections). This is intuitive because the number of connections increase, the diversity support of the organization increases. The results here focus on sparse organizational structures since it seems particularly desirable to allow the agents to organize themselves when the network is significantly resource-constrained.

the network).

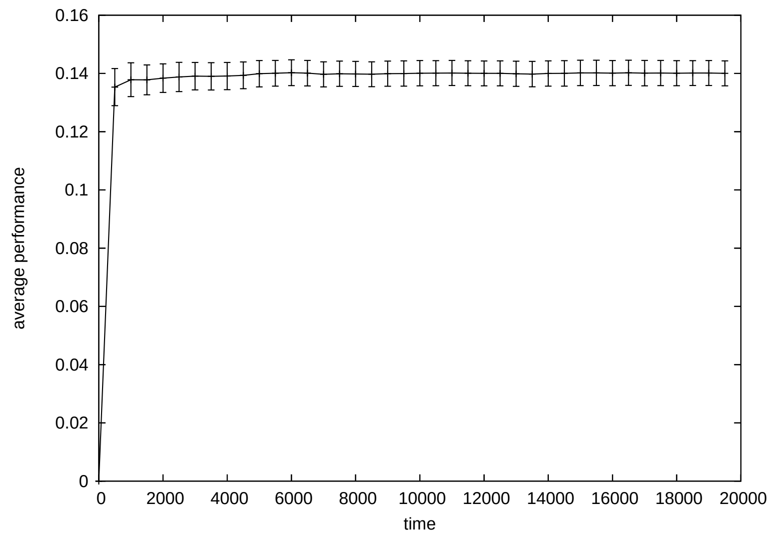


Figure 5.5: The average performance of the team formation model with $n = 200, e = 300, \mu = 2, |T| = \sigma = 10, \gamma = 1, \alpha = 2$ over 20000 iterations averaged over 50 samples of connected random networks. The value of the average performance over all time is 0.139 with a two-sided 95% confidence interval of approximately 0.004. The errors bars depict the 95% confidence intervals.

Figure 5.5 shows the average performance over 50 different connected random networks for 20000 iterations of the team formation model. Note that the performance for the specified set of parameters levels off quickly and that there is little deviation in the average organizational performance. Also note that the overall performance is low – less than 0.14. This implies that on average, a random network supports successful team formation for 14% of all tasks introduced into the system. This is a direct result of the heavily loaded task environment and the complexities of distributed team formation in networks (Gaston

statistic	mean	95% confidence
diameter	4.86	±0.025
clustering	0.009	±0.002
normalized std. dev. k	0.967	±0.009
degree correlation	-1.7×10^{-5}	$\pm 4.0 \times 10^{-6}$

Table 5.2: Mean and 95% confidence intervals for four of the structural statistics discussed in Chapter 3 averaged over the 50 sample random networks.

& desJardins 2003; Gaston, Simmons, & desJardins 2004). Additionally, Table 5.2 contains the average structural statistics averaged over the collection of 50 fixed, connected, random networks. The performance of the multi-agent organization in fixed networks and the structural statistics of these fixed networks will be used as baselines in the study of AON behaviors.

AON Results

The experimental results for the four AONs described above in an identically parameterized team formation environment are shown in Figure 5.6. The figure shows the relative increase in performance over the team formation performance in the static network structures. In the AON experiments, the same 50 connected random graphs as used above were the initial organizational structures. The experiments were organized such that the agents were prohibited from adapting their connectivity until the 1000th iteration of the model.

In order to understand the differences in performance between the AON strategies and the static structures, as well as the performance differences among the AON strategies, var-

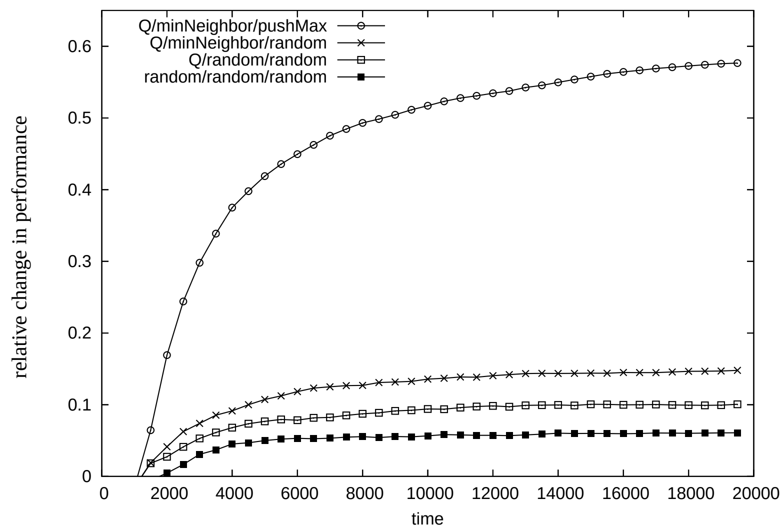


Figure 5.6: The relative performance of the multi-agent organizations when the agents employ the four increasingly complex AON strategies. The performance is relative to the average organizational performance over the 50 static, connected random graphs. All of the increases in performance are statistically significant over the static network performance. The large difference between the *Q/minNeighbor/maxQ* strategy and the rest of the AON strategies is statistically significant at the 95% confidence level for *Q/minNeighbor/maxQ* approximately equal to 0.034.

ious statistical measurements were taken as the networks evolved over time. The evolution of these statistics is shown in Figure 5.7. The four statistics shown are network diameter, clustering, the normalized standard deviation of degree, and degree correlation among neighboring agents. Details of these statistics is provided in Chapter 3.

The discussion of the effects of AONs is further supported by the network structures shown in Figure 5.8. The figure contains five networks: one of the initial random networks and samples of network structures resulting from the AON strategies after 20,000 iterations.

Note that the networks shown in Figure 5.8 (b), (c), (d), only show the largest component of each network. Details of this are in the caption.

AON Discussion

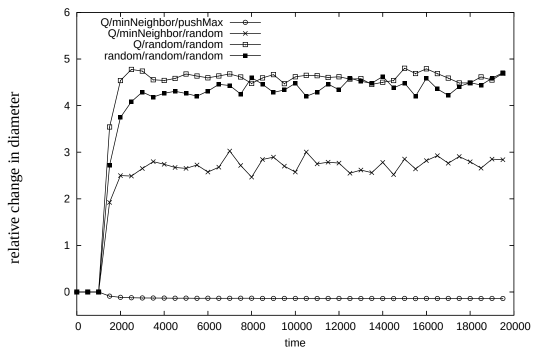
The main, and most obvious, result is that as the AON strategies increase in sophistication, the performance increase as a result of the AON is much greater. That is, the *Q/minNeighbor/pushMax* AON strategy results in the greatest improvement in performance: approximately a 55% increase in efficiency (see Figure 5.6). Note, howerver, that *all* of the AON strategies, including the purely random strategy, led to statistically significant increases in organizational performance. This implies a significant benefit of adaptive organizational structures in networked multi-agent team formation and supports the claim that distributed network adaptation is a form of organizational learning. While all of the AONs improve performance, agents that discriminate among their current connections and select the connection of least value to rewire outperform agents that rewire randomly. Furthermore, agents that take meaningful referrals from the agent that is losing the connection outperform agents that randomly decide where to establish new connections. This is suggestive that the benefit of decentralized network adaptation is largest when the agents intelligently adapt their local connectivity.

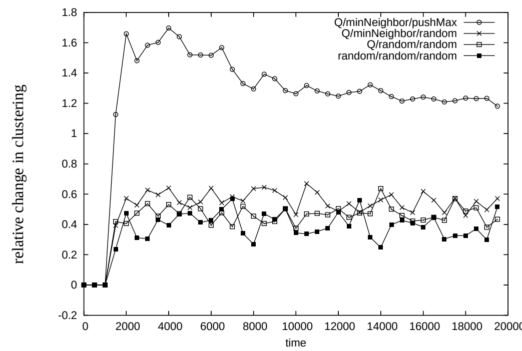
Figure 5.6 shows that the structural evolutions of the first three AON strategies (*random/random/random*, *Q/random/random*, and *Q/minNeighbor/random*) are quite similar.

This is largely a result of the random rewiring common to all three strategies. The one difference between the three is the diameter of the resulting networks. For all three strategies, the diameter of the networks drastically increases compared to the static, connected random graphs. This is a direct result of the strategies disconnecting the network structure (i.e., breaking the structure into disjoint components). The difference in diameter for the *Q/minNeighbor/random* strategy compared with the other two random strategies is that the agents using the *Q/minNeighbor/random* never drop a connection with an agent that only has one connection. That is, they do not abandon their neighbors that completely rely on them for connectivity into the rest of the network. This fact, along with the slight difference in the other structural characteristics of the networks, accounts for the difference in performance among the first three AON strategies.[8]

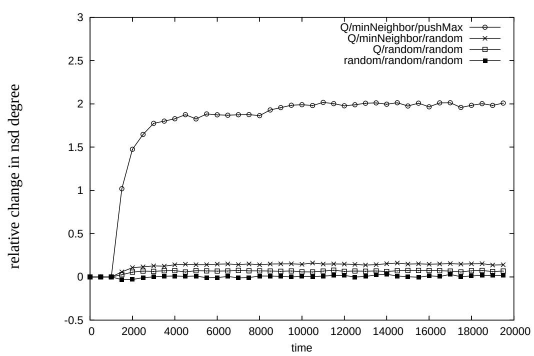
The fourth AON strategy, *Q/minNeighbor/pushMax*, produces the largest average increase in performance and results in very different network structures than the other strategies. In addition, the network structures that result from this AON strategy are significantly different than the initial random networks. Unlike the other strategies, because it is a push referral-based strategy, *Q/minNeighbor/pushMax* guarantees that the network will remain connected. In fact, the diameter shrinks. This phenomenon can be explained by considering some of the other network characteristics.

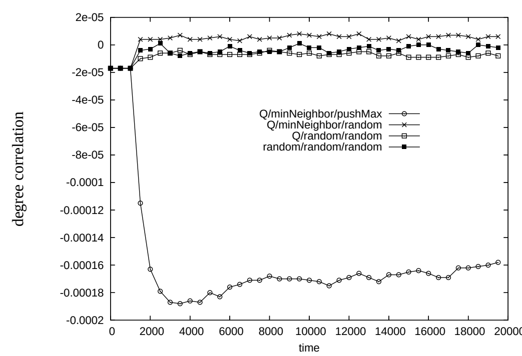
[8]This finding is repeated in all of the experiments: namely, that slight structural changes often lead to measurable changes in performance for networked multi-agent team formation.





(a) (b)





(c) (d)

Figure 5.7: The evolution of the network statistics for the four AON strategies averaged over 50 iterations. The four statistics shown are: (a) relative diameter, (b) relative clustering, (c) relative normalized standard deviation (nsd) degree, and (d) absolute degree correlation. The symbols used to represent the strategies in the figure were selected to match the symbols of the three strategies in Figure 5.6.

Under *Q/minNeighbor/pushMax*, the agents push the degree distribution of the network toward a more spread out (possibly, skewed) distribution. This is reflected by the dramatic increase in relative normalized standard deviation of degree (nsd degree) (Figure 5.7 (c)). An increase in normalized standard deviation in degree means that some agents in the network are increasing their connectivity, while others are decreasing their connectivity.[9] The observed decrease in degree correlation is consistent with the decrease in diameter. As with the star network studied above, decreasing the diameter of a network with a constant number of connections is directly proportional to increased diversity support. Diversity support is useful in a dynamic task environment in which there is a large number of possible skill mixes for the tasks (e.g., given the parameters of the environment used here, by Proposition 6 there are $\Omega(10, 10) = \binom{19}{10} = 92,378$ possible task combinations).

The AON strategies, and in particular *Q/minNeighbor/pushMax*, increase clustering. This is interesting for several reasons. First, increasing clustering and decreasing diameter are contradictory in that clustering and diameter tend to be negatively correlated. If agent i is already two hops away from agent k via a direct connection with agent j, and agent i wants to decrease its average distance to all agents in the network, agent i can do so most effectively by connecting to an agent other than k. Conversely, in order to increase clustering in this situation, i would want to connect to k directly, creating the triangle $\{i, j, k\}$. Second, I did not predict that increased performance would be correlated with

[9] Since the domain requires resource-constrained AONs, the number of total connections, and therefore the sum of degrees, and average degree in the network remains constant.

increased clustering. In hindsight, a plausible explanation may be that *increased clustering reduces competition and blocking*. Given the parameters of the team formation environment and the team joining and initiation strategies of the agents, a group of agents that are closely connected (i.e., that exhibit excess clustering) are more likely to all join the same team. Therefore, they reduce competition among various teams in their part of the organization and likely reduce the effects of blocking.

Figure 5.8 shows that the representative networks for static structure and the first two AON strategies are all similar. This changes when the agents use *Q/minNeighbor/random*, for which some changes in structure are apparent. There is an increase in the number of chains of nodes outside the core of the network. This is most obvious looking at the "tail" at the bottom of the network shown in Figure 5.8(d). Finally, the structure of the network resulting from the *Q/minNeighbor/pushMax* is qualitatively different from the other four. In this network, the hub and spoke structure is clearly discernible. This structure is consistent with the evolution of the structural statistics described above.

AON Summary

The results presented in this section suggest that decentralized network adaptation driven by the decisions of individual agents can significantly increase organizational performance in networked multi-agent team formation. This section is concluded by considering why the push referral-based *Q/minNeighbor/pushMax* strategy overwhelms the remainder of the

AON strategies. The “rich get richer” concept makes sense in considering the structure of the network that results from this strategy. Agents that are losing a connection in this strategy, refer high performing agents (i.e., agents that participate on a large portion of successful teams) to the agents that are rewiring. The accumulation of connectivity only helps the high performing agents perform better as their local diversity support increases with the addition of the new connection. That is, the high performing agents are the high degree agents and under this strategy, these agents accumulate more connections. Additionally, this creates a dependency of low degree agents on higher degree agents which reduces the amount of competition among different teams since the dependent agents are likely to join the teams that the agent they depend on joins.

In this section, the utility of AONs was demonstrated by describing and experimenting with four strategies that make incremental improvements from one to the next. The base strategy was a purely random strategy that provided improved performance over no adaptation, but was outperformed by more complex, more intelligent, agent adaptation strategies. In the next several section, I consider various alternatives to the strategies discussed in this section. These alternatives include other push referral methods, knowledge-based methods, and bilateral adaptation.

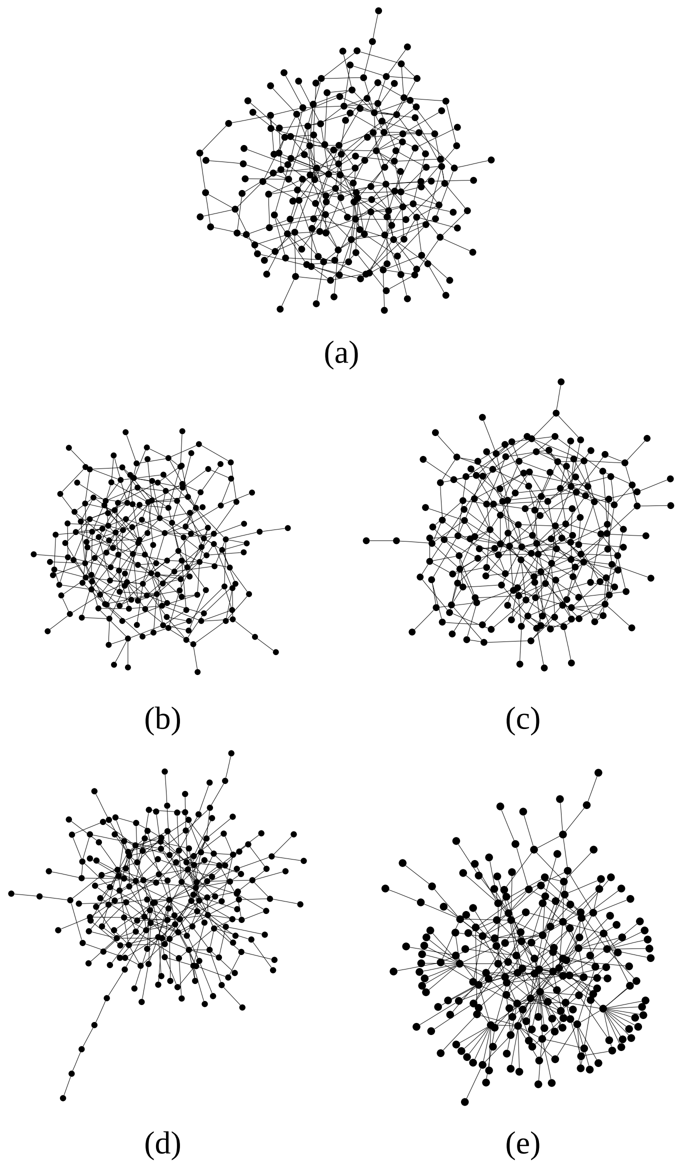
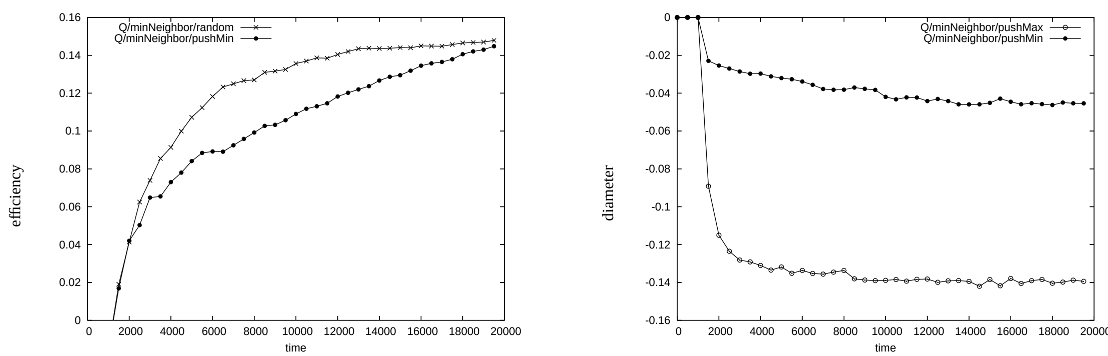


Figure 5.8: Representative network structures: (a) an initial, random, connected network; (b) a network after 20000 iterations where the agents used *random/random/random*; (c) a network resulting from *Q/random/random*; (d) a network resulting from *Q/minNeighbor/random*, and (e) a network resulting from *Q/minNeighbor/pushMax*. The networks shown in (b) and (c) show only the largest component of the network; each has 12 missing agents. Similarly, the network shown in (d) is only the largest component with four missing agents. The network in (e) contains all 200 agents (see text).

5.3.3 Are Push Referrals Always Best?

One conclusion that could be drawn from the previous section is that push referrals will always perform well since they are guaranteed to maintain connectivity. Here, this claim is refuted by considering an alternative push referral strategy, *Q/minNeighbor/pushMin*. In this strategy, agent i decides to adapt and decides to rewire its current connection with agent j. Agent j provides a push referral just as before, but now j refers its lowest performing neighbor. In this scenario, agent j could be considered a philanthropist, trying to improve things by connecting two agents having lower local performance.



(a) (b)

Figure 5.9: A comparison of the alternative push referral strategy, *Q/minNeighbor/pushMin*, with two of the previously discussed AONs for team formation. (a) compares the performance relative to no adaptation of the alternative strategy with *Q/minNeighbor/random*, and (b) compares the diameter relative to no adaptation of the alternative strategy to *Q/minNeighbor/pushMax*.

Figure 5.9 shows the experimental results for *Q/minNeighbor/pushMin*. To compare the behaviors of this strategy with those discussed above, the new push referral strategy's performance is compared to *Q/minNeighbor/random* and the only structural comparison is in diameter compared to *Q/minNeighbor/pushMax*. As with the other strategies, *Q/minNeighbor/pushMin* is able to increase performance, but on average much slower and less than *Q/minNeighbor/random* over the 20000 iterations in the experiments. This suggests that the added information in the new strategy is actually detrimental, since it performs worse than random rewiring (i.e., no information). Figure 5.9 (b) shows the relative change in diameter. When the agents use *Q/minNeighbor/pushMin*, the diameter decreases and the network remains a single component. However, the diameter does not decrease as much as that of the agents using *Q/minNeighbor/pushMax*.

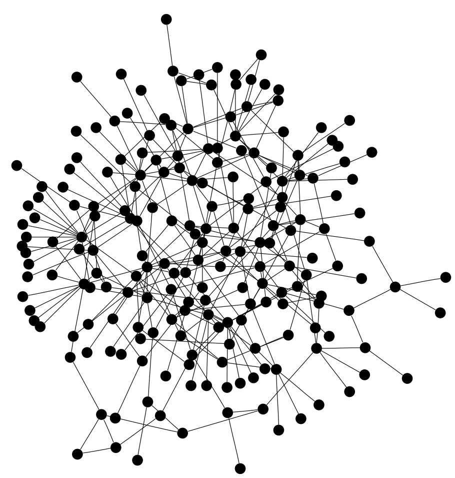


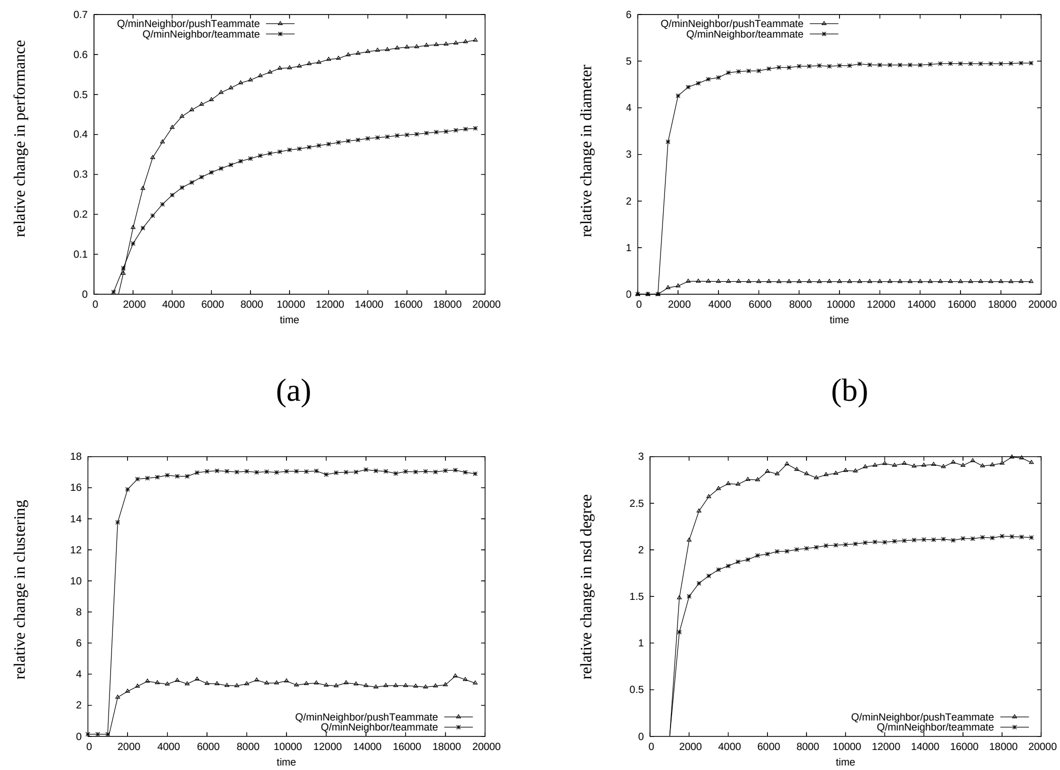
Figure 5.10: A network structure after 20000 iterations of the team formation model when the agents employed the AON strategy *Q/minNeighbor/pushMin*. There is some hub-and-spoke structure (left side of the network), but much less than seen in Figure 5.8 (e) for *Q/minNeighbor/pushMax*.

This simple experiment shows that while AONs with push referrals can increase performance and maintain a connected network, some push referral rewiring approaches are better than others. In addition, push-referral-based strategies will not always outperform non-push-referral-based strategies.

5.3.4 History-Based AONs for Team Formation

Another method for the agents to rewire is based on accumulated knowledge. The AON strategies considered so far have not used acquired knowledge for deciding where to make new connections; instead, the strategies discussed above either make random rewiring decisions or use information about the current network structure. The proposed alternative in the team formation environment is for each agent to maintain a list of frequency counts for the number of times it has been on successful teams with each of the other agents in the organization. Note that the collection of this historical information gives the agents the ability to see beyond their local network structure. The assumption is that the agents working on a team know something about each other, regardless of whether they have a direct connection or not.

There are at least two different ways that agents can use historical teammate information in order to adapt their local connectivity. First, once an agent has decided to adapt and has selected a connection to remove, the agent can choose an agent to connect to using the acquired historical knowledge of teammates. One method for doing so is for an agent i to select the agent j that has been on the most successful teams with i but is not currently directly connected to agent i. The AON strategy that uses the stateless Q-learning mechanism for deciding when to adapt, the exponentially weighted moving average for determining the connection to break, and the selection of the maximum unconnected agent from the teammate history for determining a new connection is called *Q/minNeighbor/teammate*.



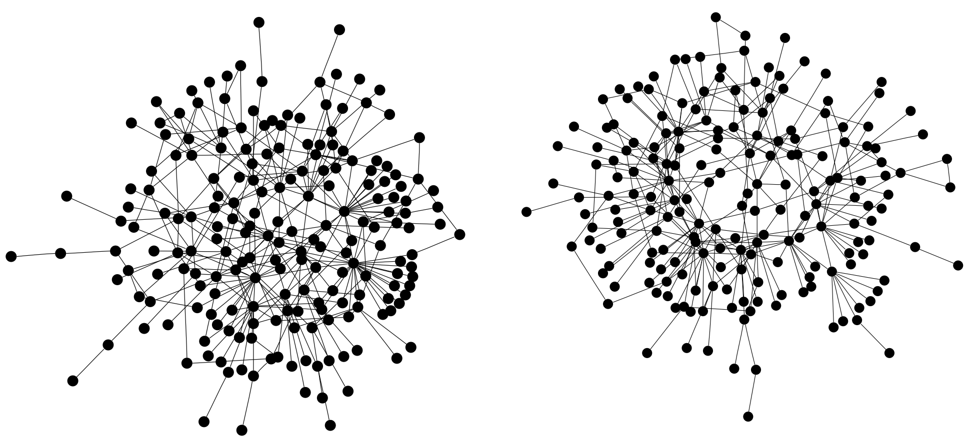
(a) (b)

(c) (d)

Figure 5.11: Experimental results for the history-based rewiring strategies averaged over 50 simulations of 20000 iterations of the team formation model: (a) relative average change in organizational performance, (b) relative change in diameter, (c) relative change in clustering, and (d) relative change in the normalized standard deviation of degree.

An alternative to the strategy above is to use a modified form of push referrals. In this situation, agent i decides to adapt and chooses to remove its connection with agent j.

Agent j then provides agent i with a push referral to the agent with whom j has been on the most successful teams, say agent k. Agent i then establishes a connection with agent k, provided that k is not i and i is not currently connected to k. This strategy, in a slight abuse of terminology, is referred to as *Q/minNeighbor/pushTeammate*. This is not a true push referral, and therefore does not guarantee that the network will stay connected. In fact, as the experiments show, the network can and does become disconnected when the agents operate under this strategy. This is due to the fact that the new connections are to an agent that *was* historically connected to the losing agent, while a true push referral guarantees that the losing agent and the gaining agent are *currently* connected.



(a) (b)

Figure 5.12: Representative resulting networks for the two teammate history-based AON strategies: (a) *Q/minNeighbor/pushTeammate* and (b) *Q/minNeighbor/teammate*. Note that (a) shows the entire network and (b) shows the largest component with nine agents missing.

Figures 5.11 and 5.12 show the result of the same experiments as above for these two history-based strategies. Interestingly, and surprisingly, the push referral teammate strategy performs very well, outperforming the best of the previously discussed strategies: *Q/minNeighbor/pushMax*. One reason for this boost in performance is that the agents are still giving referrals based on performance (i.e., the agent that has helped the most), but now the pool of candidates for referral can be much larger than just the losing agent's immediate neighbors. In fact, over time, these histories may contain the entire agent organization. This is realistic in some, but not all, multi-agent environments. An alternative to maintaining historical information about all agents may be to maintain a list of the top m teammates.

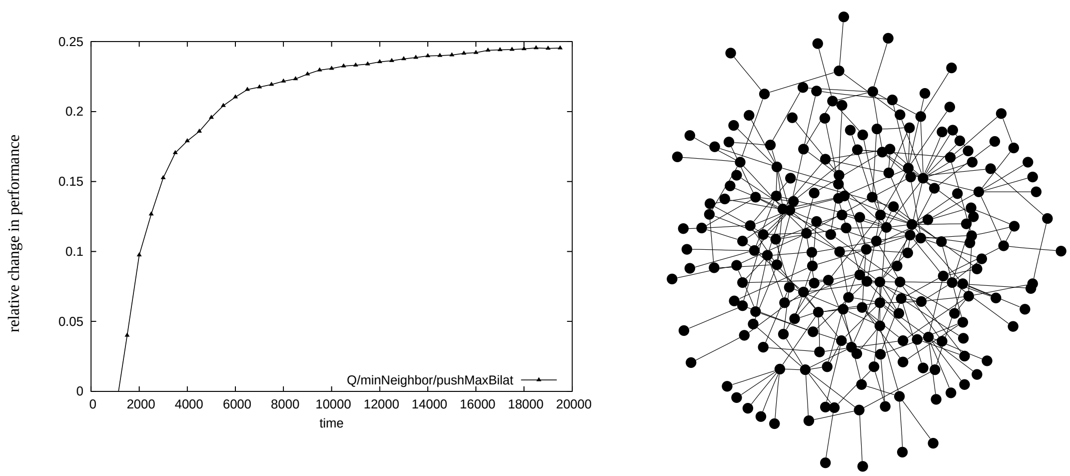
The main reason for the increased performance of the referral approaches over the non-referral history-based strategy is its ability to maintain a mostly connected network. Although this modified push referral does not guarantee connectivity of the network, it comes close to maintaining connectivity, as evidenced by the results shown in Figure 5.11 (b) and the representative resulting network in Figure 5.12. Additionally, the teammate referral strategy does not limit an agent to its own knowledge, but allows an agent to capitalize on its neighbor's knowledge. Finally, because the teammate strategies draw from a larger pool of top performers, the skew in the degree distribution over the agents grows quickly. That is, the "rich get richer" faster.

5.3.5 On Bilaterally Stable AONs for Team Formation

Not mentioned in the descriptions of the experiments and AON strategies is the amount of rewiring that occurs in each of the AONs. As it turns out, in the experiments above, the average cumulative number of rewirings for all of the various strategies fall within the standard deviations of each of the other strategies. That is, all of the strategies use approximately the same number of rewirings on average; for the *random/random/random* strategy, the probability of rewiring was calibrated appropriately. Another similarity of the strategies discussed above is that they do not converge. They continue rewiring over all 20000 iterations (and beyond).

As the quote in the beginning of this chapter suggests, an informal, adaptive, flexible organizational network structure may be very productive. While this is the case, it may also be desirable to achieve stability, or at least stability in the absence of shocks to the system. One method for attempting to achieve stability in decentralized adaptive organizational networks is to adopt the notion of bilateral network formation from the economic theory of network formation. To summarize, bilateral network formation only creates a connection if both agents involved in the connection agree to the connection, while either agent involved in a connection can elect to remove the connection. To translate this to resource-constrained AONs, an agent can unilaterally elect to attempt to rewire a connection, but the agent that is selected to establish the new connection must agree to the connection. Most of the AON strategies discussed previously in this chapter can be modified to provide bilateral network formation.

In order to test bilateral network formation for AONs in the team formation environment, the *Q/minNeighbor/pushMax* strategy was modified to guarantee bilateral connection establishment. The rules are the same, but the agent selected to establish the new connection must agree to establish the connection with the rewiring agent. For the gaining agent to agree to the new connection, the gaining agent must have $Q(rewire) > Q(nothing)$. If the gaining agent declines the new connection, the rewiring is aborted and the original connection (the one selected for rewiring) remains in place. This new strategy is referred to as *Q/minNeighbor/pushMaxBilat*.



(a) (b)

Figure 5.13: The experimental results for the bilateral AON strategy: (a) the average relative change in organizational performance and (b) a representative resulting network structure. For details on the structure of the resulting networks see the text.

The expectation was that the bilateral AON strategy would have far fewer network adaptations over the course of the team formation simulations. In fact, the organization of agents using *Q/minNeighbor/pushMaxBilat* performed only 154 ± 4 network rewirings on average and stopped rewiring after approximately 1500 iterations from the start of adaptation. This represents a 98.3% reduction in the number of rewirings, compared to the nearly 9000 rewirings over 20,000 iterations that occur on average in organizations of agents that employ the other AON strategies.

Figure 5.13 (a) shows the average relative performance curve for the organization of agents using the bilateral AON strategy. Even though the bilateral AON performs far fewer adaptations, it outperforms all of the AON strategies described above except the teammate history-based strategies and *Q/minNeighbor/pushMax*. The figure also depicts a representative network structure resulting from the bilateral AON strategy, within which can be seen evidence of a hub-and-spoke type structure, similar to the high-performing AONs discussed above. This is supported by an average relative decrease in diameter of approximately 7%. The brief period of adaptation, the small number of adaptations, and the rapid increase in organizational performance suggests that the bilateral AON strategy considered here allows the agents to learn quickly how and when to adapt and allows the agents to make smart decisions about how to adapt.

As mentioned in the beginning of this section, stability is not always a desirable property of an organization. However, bilateral, intelligent AON strategies may be useful when the cost of adaptation is excessive. As demonstrated here, bilateral AONs can allow the agents to quickly learn how to adapt their connectivity; leading to significant increases in organizational performance.

5.4 Concluding Remarks

Distributed, networked, multi-agent team formation in a heavily loaded task environment represents a challenging cooperative, collaborative, and competitive domain. There are many possibilities for organizational learning in such a domain. After demonstrating that the network structure of a multi-agent organization has a dramatic effect on organizational team formation performance through simple examples and computational experiments, this chapter proposed and discussed multiple AON strategies for agents embedded in an organization attempting to complete tasks. The utilities of the AON strategies were studied empirically through simulation and resulted in significant increases in performance over static network structures. The experimental results support the claim that intelligent, decentralized network adaptation is a method of organizational learning.

The network adaptation strategies in this chapter were built from the general AON framework presented in Chapter 4. In the next chapter, AONs are applied to another generic multi-agent environment: distributed production and exchange. Various AON strategies are considered, including several based on the general AON framework.

Chapter 6

Navigating Production and Exchange Networks

We're already on the way to an expanded economy full of new participants: agents, bots, objects, and machines, as well as several billion more humans.

Kevin Kelly

In a networked environment increasing returns are created and shared by the entire network. Although one networked entity may accrue more gains than another, the real value resides in the greater web of relations.

Angus Matthew

In his article "New Rules for the New Economy," Kevin Kelly emphasizes the role of networks in today's economy. Interestingly, Kelly includes agents and bots as key participants in the new, global, networked economy. As the number of agents and bots continues to increase in the open market environment of the Internet, the ability of these agents to maintain and dynamically update their economic social connections with other agents will become increasingly important. One factor will be limited cognitive and processing capacities hindering any single agent's ability to interact with *all* other agents. Also, in open market environments, agents will come and go without constraint, therefore requiring other agents to adapt to a changing interconnected market.

In this chapter, I begin the exploration of AONs in multi-agent market environments. After briefly reviewing some of the relevant literature, I present a model of multi-agent production and exchange and demonstrate through computational experiments that the network structure governing trading among the agents has a significant effect on organizational performance. I then develop several AON strategies for agents operating in the production and exchange environment and empirically demonstrate the utility of these AONs.

6.1 Overview of Multi-Agent Market Environments

One of the most promising, and potentially lucrative, application domains for multi-agent technologies is in the automation of commerce and trading (Wellman 2004). Enabled by global communications systems such as the Internet, electronic marketplaces are rapidly becoming part of the global culture. The many applications of agent technology to e-commerce include market clearing, automated matching of buyers and sellers, automated trading agents, automated formation of supply chains,and management of distribution systems (Wellman 2004; He, Jennings, & Leung 2003).

In the multi-agent systems community, the importance of agent-enabled ecommerce is demonstrated by increasing participation in the Trading Agent Competition (TAC) held in conjunction with the International Joint Conference on Autonomous Agents and Multi-Agent Systems (Wellman *et al.* 2003a; 2003b). Past competitions have included challenges in the travel agent and supply chain domains. In the travel agent scenario, agents manage the scheduling and purchasing of transportation and local arrangements for a set of specified trips. In the supply chain scenario, agents are challenged with managing an inventory, including taking orders, placing orders, and distributing goods in a multi-tiered supply chain.

There is a large literature on multi-agent markets that is beyond the scope of this dissertation. Instead of surveying this large literature, I briefly survey some of the literature that is most relevant to *networked* multi-agent market environments (i.e., distributed markets where all of the agents do not necessarily know about all other agents in the market).

One of the central problems in distributed market environments is effectively pairing buyers and sellers or, more generally, traders (Walsh & Wellman 2000). A common mechanism for matching buyers and sellers is auctions. Auctions are effective for pairing buyers and sellers, but have certain drawbacks.

> “In many large-scale economies, the problem of who to buy or sell from is a significant one. Mechanisms that pair buyers and sellers, such as auctions, can be used to do this when the number of agents is relatively small, but if the population is large or the auction is combinatorial, both the allocation of goods

> and the selection of bids becomes a difficult computational problem.” (Brooks & Durfee 2002)

Combinatorial auctions are auctions in which collections of goods can be bid on as bundles (Cramton, Shoham, & Steinberg 2006; Sandholm 1999). A classic example comes from the shipping industry. Consider a company bidding for the delivery of goods between cities. If routes can only be bid upon one at a time, a company is likely to bid a relatively high price for all routes. On the other hand, if routes can be bid upon in bundles (i.e. combinatorially), then a company can bid a lower average price per route for two routes, one of which is a return trip to the starting city. When bidding on combinations of goods is permitted, the complexity of market clearing algorithms greatly increases.

The solution to the “difficult computational problem” offered by Brooks and Durfee is a variant on coalition formation called congregation formation (2002). In congregation formation, agents move among a fixed set of smaller markets until they arrive in a market where the interests of the agents are all similar. Once the agents organize into the smaller markets, the computational problem of clearing combinatorial auctions is diminished due to the smaller number of agents in any local market. Although there is no explicit representation of an organizational network in congregation formation, congregation formation serves as a motivational example to emphasize the importance of organizational structure in multi-agent markets.

Until recently, there has been little work on the role of networks and their formation in economic situations. In the economics literature, the process of network formation has been studied from a game-theoretic perspective (Jackson & Wolinsky 1996; Watts 2001; Jackson & Watts 2002; Jackson 2003). One of the most widely studied models from the economic network formation literature is the Connections Model (Jackson & Wolinsky 1996), in which there is no buying and selling; instead, there are costs for direct connections and implied financial benefits for direct and indirect connections. For a detailed description of this model and related theory, see Chapter 3.

At the intersection of economics and computer science, recent research has shown that networks play a critical role in understanding and affecting the dynamics of market economies. The existence of equilibria have been studied in economic networks, including real-world international trade networks (Kakade *et al.* 2004) and in generalizations of the classic Arrow-Debreu economics to networks of consumers and economies (Kakade, Kearns, & Ortiz 2004). Additionally, the structure of networks has been shown to affect price convergence in a distributed production and exchange economy (Wilhite 2001).

Finally, in work most closely related to what I present in this chapter, partner selection has been discussed within the fields of computational economics and multi-agent systems. Studies of a trade network game have demonstrated that distributed partner selection can improve efficiency in small-scale trade networks (Tesfatsion 1997). More recently, reinforcement learning has been proposed as a way to learn effective agent interactions based on reputation in multi-agent market environments (Tran & Cohen 2003).

In the remainder of this chapter, I present an adaptation of a model of a multi-agent production and exchange economy, discuss the effects of various static network structures, develop several AON strategies for the agents to adapt local connectivity in the production and exchange networks, and present empirical results on the performance of organizations of agents using the various AON strategies. The work presented in this chapter differs from the related literature in two important ways: (1) the network formation process is bottom-up: that is, the agents must reorganize an existing network structure with limited resources, and (2) direct attention is given to the network structures that arise from the various AON strategies and the correlations between structure and performance.

6.2 Modeling Decentralized Multi-Agent Markets

In order to study mechanisms for AONs in multi-agent economies, I selected a simple, generic, yet realistic model of a production and exchange economy. In this section, I describe the model and discuss the effects of agent social structures on the organization's ability to distribute goods effectively.

6.2.1 A Model of Production and Exchange

The basis for the model was first presented by Wilhite (2001; 2003). Each agent is given an initial endowment of two distinct goods, and has a fixed production capacity. At each time step, each agent is allowed to choose whether to trade or to produce. The goal of the individual agents is to maximize their utility. The model assumes that agents' trading behaviors are perfectly rational (i.e., they always select the action that maximizes their utility) and completely truthful (i.e., they always provide perfect information during negotiation and trade).

Let there be n agents in the economy and two goods, g_1 and g_2, where g_2 is infinitely divisible and g_1 must be traded in whole units.[1] The utility of agent i is defined to be

$$U^i = g_1^i g_2^i. \tag{6.1}$$

It follows that if an agent possesses a total of $G = g_1 + g_2$ goods, then the optimal allocation is $g_1 = g_2$.

In the original model, the agents were given the ability to produce a set amount of both goods. In order to promote trading among the agents and focus on the efficiency of the trading structure, I restrict the agents to being able to produce only one of the two goods. This restriction requires that agents trade to maximize utility. Assuming that agent i is a producer of g_1, $\Delta g_1^i \in [1, q]$ and $\Delta g_2^i = 0$ (and likewise for producers of g_2), where q is a model parameter. The exact production capacity of an agent is drawn uniformly at random from the interval. This allows for a society of heterogeneous agents, in which some are effective producers and others are poor producers. The latter must rely more heavily on trade to increase their utility.

[1] Forcing g_1 to be traded in whole units is to simplifies the price formation and trading process. It is claimed that this adds realism to the model (Wilhite 2003).

In the original model, during each iteration, the agents are selected in random order and are allowed to negotiate (i.e., determine the price of trading) with m other agents. The selected agent then chooses the action—trade with one of the m agents or produce—that maximizes its utility. In negotiation, each agent truthfully reveals its *marginal rate of substitution*,

$$mrs_i = \frac{\delta U^i/\delta g_1^i}{\delta U^i/\delta g_2^i} = \frac{g_2^i}{g_1^i}. \tag{6.2}$$

When the two negotiating agents' marginal rates of substitution differ, there is an opportunity for mutually beneficial trade between the two agents (Wilhite 2003). Assuming that agent i is negotiating with agent j, the next step in the negotiation process is to calculate the exchange price:

$$p_{i,j} = \frac{g_2^i + g_2^j}{g_1^i + g_1^j}. \tag{6.3}$$

The price, as determined by Equation (6.3), is the amount of good g_2 that an agent is willing to exchange for one unit of g_1. When agents i and j are negotiating, if $mrs_i > mrs_j$, then agent i will be willing to trade $p_{i,j}$ units of g_2 for one unit of g_1. The agents must also take into account a trading tax τ, which is given as a model parameter. Assuming that agent i is trading one unit of g_1 for $p_{i,j}$ units of g_2 with agent j, the tax is applied to the transaction such that

$$g_1^i = g_1^i - (1.0 + \tau) \quad \text{and} \quad g_2^j = g_2^j - (1.0 + \tau)p_{i,j}. \tag{6.4}$$

The agents repeatedly trade in this manner until the exchange no longer increases the utility of either agent.

During negotiation, the agents do not actually exchange goods; rather, the active agent computes the change in utility $\Delta U^i(j)$ that would result from trading with each of the m agents with whom it has negotiated. The agent also calculates its change in utility after production:

$$\Delta U^i(g_1) = (g_1^i + \Delta g_1^i)(g_2^i + \Delta g_2^i). \tag{6.5}$$

Finally, the agent selects the action that results in the largest ΔU^i and then all of the goods are either exchanged or produced accordingly. To reiterate, an agent only takes an action if it increases the agent's utility (and similarly, a trade only occurs when it simultaneously increases the utility of both agents).

This model was originally studied in the context of global price conversion and the roles that individual agents adopted: heavy traders, heavy producers, and specialized producers. The nature of the interactions in the initial study was based on each agent selecting m other agents at random to interact with at each time step (Wilhite 2003). I refer to this interaction paradigm as *random mixture*.

An alternative to random mixture is to embed the agents in a fixed network topology and only allow agents that are directly connected to negotiate and trade. It has been shown that network structure has a direct impact on the rate at which the economy converges on a global price (Wilhite 2001) when the agents only trade (i.e., have no production capacity). In the next section, I demonstrate the dependence of organizational performance on network structure using both production and trading dynamics.

6.2.2 The Effects of Network Structure

In order to evaluate the efficiency of the trading network, I extend the model to *clear* a variable amount of goods from the system at the end of each iteration. In essence, I introduce consumption into the model. At the end of each iteration, every agent "consumes" an amount of g_1 and g_2 equal to the lesser amount of the two goods. That is, after all agents have finished either initiating trade or producing, each agent i updates its amounts of the two goods such that

$$g_1^i = g_1^i - \min(g_1^i, g_2^i) \qquad \text{and} \qquad g_2^i = g_2^i - \min(g_1^i, g_2^i), \tag{6.6}$$

where $c_i = \min(g_1^i, g_2^i)$ is the *clear amount* of agent i during the current iteration. This clearing mechanism can be interpreted as the agents requiring one unit of each of the two goods in order to accomplish one unit of some other task. An example of this may be a collection of agents where some have processing capacity and others have storage capacity, both of which are required in order to complete a large-scale data processing task. In this example, agents are able to share (i.e., trade) their respective resources in order to accomplish the collective data processing task.

The notion of an individual clear amount lends itself directly to a measure of organizational performance. The *average clear amount per agent*,

$$C = \frac{1}{n} \sum_i \min(g_1^i, g_2^i), \tag{6.7}$$

is an intuitive measure of organizational performance, since agents must trade in order to clear goods. Trading structures that support more effective trade among the agents lead to a larger average clear amount per agent.

To motivate the need for AONs, I measured the average clear amount per agent after 10000 iterations of the model for each of five static network structures:

- **one-dimensional lattices –** agents organized in a ring, connected only to their nearby neighbors,
- **two-dimensional lattices –** agents organized in a two-dimensional toroidal grid,
- **one-dimensional small-worlds –** one-dimensional lattice with a small percentage of connections randomly rewired to provide “short-cut links” (Watts & Strogatz 1998),
- **random graphs –** connections exists between agents i and j with probability p (Erdos & Renyi 1959), and
- **star topologies –** hub agents are connected to all other agents in the system, approximating a scale-free network (Albert & Barabási 2002).

Figure 6.1 shows the average clear amount per agent over 200 simulations for each of the five network structures as a function of the tax τ. (The error bars for the 95% confidence intervals are within the size of the points.) All of the network structures were parameterized to have the same number of nodes (i.e., agents) and connections as the two-dimensional lattice (200 agents and 400 undirected connections).

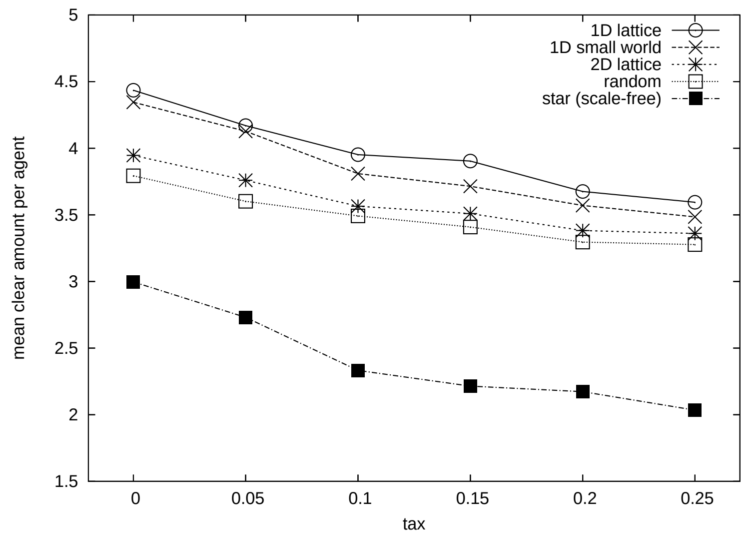


Figure 6.1: The effects of various network structures on the average clear amount per agent after 10000 iterations over 200 simulations. The network structures were appropriately parameterized to all have exactly 200 agents and 400 connections. The model parameters follow those used by Wilhite in the original study: $q = 30$ and the initial endowment of each of the goods for each of the agents was drawn uniformly at random from [1,60].

As expected, there is a statistically significant difference in the resulting average clear amount for the different network structures in line with previous results (Wilhite 2001). There are several interesting observations based on the results. First, and most obvious, network structure directly affects organizational performance in the production and exchange network. Interestingly, the one-dimensional lattice structures perform the best. It is conjectured that this is a result of less competition for goods within local regions of the network, which is in turn a direct result of no (or few in the case of the small-world network) long-range connections. The other observation is in the way the various network structures are affected by increases in the trade tax. The organization of agents embedded within the star topology suffers the most as a result of increased trade tax. This is a direct result of all trades having to go through a small set of hub agents that serve as clearinghouses for trades. When the trade tax increases, these agents can no longer "afford" to serve as clearinghouses for as many trades. On the other hand, the random network structure is most resilient to increases in the trade tax, because of the lack of locality in the network structure.

The significant effect that the network structure has on the performance of the production and exchange economy motivates the need for decentralized mechanisms that allow individual agents to adapt their local network connectivity in order to improve organizational performance.

6.3 AONs for the Production and Exchange Economy

In this section, I develop and evaluate various AON strategies for the agents embedded in the production and exchange model. The section begins with an incremental development of an intelligent AON strategy based on the general framework presented in Chapter 4. Subsequent sections offer alternative, more specialized AON strategies and compare their performance.

6.3.1 Increasingly Intelligent AONs

Similar to the chapter on multi-agent team formation, I begin the discussion of AONs for the multi-agent production and exchange environment by introducing a succession of increasingly intelligent AON strategies. The AONs described in this section are named using the *when/which/where* notation introduced in Chapter 5.

A purely random strategy serves as the base strategy for comparison. An agent using the purely random strategy, denoted *random/random/random*, decides to adapt at random based on a specified probability, randomly selects one of its connection to rewire, and randomly selects a new agent with whom to establish a new connection. This “zero-intellignece” strategy provides a benchmark for comparing the performance of more intelligent AON strategies. While there is no intelligence in the *random/random/random* strategy, it is possible that the organization will benefit from agents using this strategy. Clearly, agents adapting their local connectivity according to this strategy will interact with a larger (and changing) set of other agents, increasing the likelihood of finding high-payoff trades (i.e., mutually beneficial trades in which a large number of goods are exchanged).

Starting with the *random/random/random* AON strategy, several successive changes are made in order to arrive at an effective AON strategy. While this approach is similar to that presented in Chapter 5, the trajectory is different. The first change to the purely random strategy is the rule for determining which connection to rewire. Employing the general AON framework, each agent maintains a set of values for each of its connections using an exponentially weighted moving average:

$$V_{ij} \leftarrow V_{ij} + \beta[W_{ij} - V_{ij}], \tag{6.8}$$

where V_{ij} is the value of the connection from agent i to agent j, $W_{ij} \in [0, 2]$ is the number of trades between i and j during the current iteration, and β is the learning, or "age-off," parameter. The value of W_{ij} ranges from zero to two, since each agent can initiate trading during a single iteration. W_{ij} can be interpreted as the usefulness of a connection. Using these values, an agent selects the connection to rewire that currently has the lowest value. The AON strategy based on *random/random/random* that selects the minimum-valued connection to rewire is denoted *random/minNeighbor/random*. In all of the experiments below, $\beta = 0.1$.

From *random/minNeighbor/random*, the next level of intelligence is to adopt a stateless Q-learning strategy for learning when to adapt local connectivity. The basis for the Q-learning approach was presented in Chapters 4 and 5. The action set is $\{nothing, rewire\}$ and the values of the actions are updated using

$$Q(a) \leftarrow Q(a) + \alpha[R_t - Q(a)], \tag{6.9}$$

where α is the learning rate and R_t is the immediate reward. Following the general framework, I use an adaptive learning rate with $\alpha_{min} = 0.05$ and $alpha_{max} = 0.4$ to avoid the pitfalls of learning too slowly when decreasing performance and too quickly when increasing performance.[2] Here,

$$R_t = \Delta c_i = c_i^t - c_i^{t-1}, \tag{6.10}$$

which is the change in the cleared amount. Additionally, when an agent takes an action using this strategy, the agent waits ten iterations before taking another action and accumulates reward over the ten iterations for the action taken at the beginning of the ten iterations. The value of ten iterations was determined experimentally. This strategy is referred to as *Q/minNeighbor/random*.

The last step is to change how an agent decides to establish a new connection. Since the agents are already keeping track of the values of each of their connections, a logical strategy is to allow the agents to pass referrals based on these values. For the final incremental change, the agents will use push referrals where the losing agent refers the agent on the other side of its maximal-valued connection. More precisely, if agent i chooses to rewire its connection with agent j, agent i then establishes a new connection with agent k, where

$$k = \max_{l \in N_j(G)} V_{lj}. \tag{6.11}$$

An organization of agents using this strategy, denoted *Q/minNeighbor/pushMax*, is guaranteed to remain connected.

Next, I present experimental results for the four AON strategies and discuss the corresponding evolutions of the network structures. The base networks used in these experiments are the same as those used for the team formation model in the previous chapter.

[2]See Chapter 4, Section 4.4.4 for details.

statistic	mean	95% confidence
diameter	4.87	±0.010
clustering	0.011	±0.001
normalized std. dev. k	0.950	±0.004
degree correlation	-1.2×10^{-5}	$\pm 2.0 \times 10^{-6}$

Table 6.1: Mean and 95% confidence intervals for four of the structural statistics discussed in Chapter 3, averaged over the 200 sample random networks.

All of the base networks are randomly generated graphs with $n = 200$ nodes and $e = 300$ connections. The base networks are a slight modification of random graphs to guarantee connectedness. Structural characteristics of the base networks are given in Table 6.1. The performance of the various AONs will be presented relative to the average performance of the static base networks, which is 3.77 with a 95% confidence interval of 0.007, averaged over 200 simulations of 20,000 iterations each. The parameters of the model for all experiments, following Wilhite (2001), are $q = 30$, initial endowments drawn uniformly from [1,60], and $\tau = 0.05$.

Figure 6.2 shows the experimental results for organizations of agents using the four AON strategies discussed above. As expected, with each additional level of intelligence added to the AON being used, performance increases. The number of network adaptations is approximately the same, at 52,000 adaptations, for all of the AON strategies.[3] The number is so large because in all cases, agents continue to adapt in an attempt to increase local performance. The increases in performance for all of the AONs are statistically significant

[3]The probability of adapting in the random strategies was calibrated so that the number of adaptations was within the range of that of the Q-learning strategies.

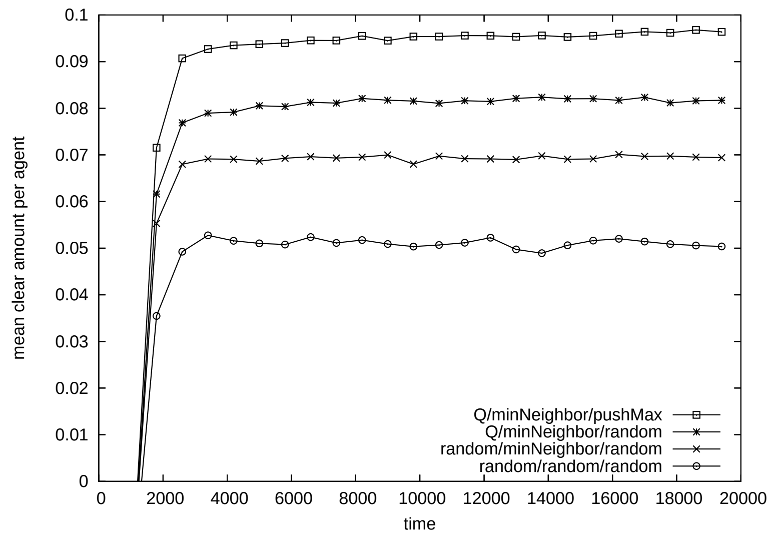


Figure 6.2: The relative performance of organizations of agents using the four AON strategies. The 95% confidence interval for *random/random/random* is 0.008 and the confidence intervals for the other three strategies are approximately 0.006.

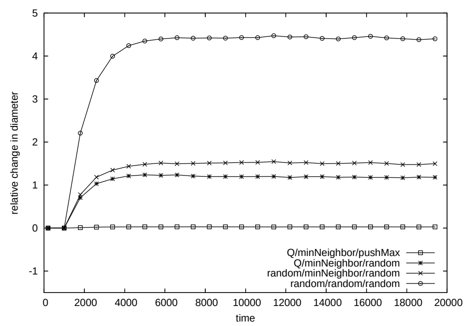
compared to the performance of the static structures. Similarly, the differences in performance among the AON strategies are also statistically significant except, for the difference between *Q/minNeighbor/pushMax* and *Q/minNeighbor/random*. See the caption for details on confidence intervals.

The increases in performance are intuitive. First, *random/random/random* increases performance as a result of the agents interacting with a larger number of other agents over time. In the static structure, agents are restricted to interacting with a fixed set of trading partners. The agents using *random/minNeighbor/random* have the added benefit of being able to discriminate among the connections they rewire. These agents, based on experience, select the connection that is of least value to rewire. Even though the rewiring remains random, the selection of the lowest-valued connection prevents an agent from rewiring a high-valued connection, hence the increase in performance over the purely random strategy.

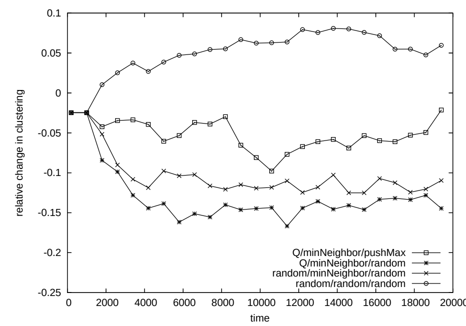
Next, the agents add the ability to learn when to make changes to local connectivity. The random adaptation policy requires that the agents adapt based on a fixed probability, regardless of current position or past performance. The simple Q-learning mechanism allows the agents to learn from experience and estimate when rewiring is more beneficial than doing nothing. Finally, in the most intelligent of the four AON strategies, the agents are also given the ability to discriminate among the agents when establishing a new connection through referrals. In the *Q/minNeighbor/pushMax* strategy, the losing agent gives a push referral to the adapting agent. This referral is the agent on the other end of the losing agent's highest-valued connection. This strategy performs better than randomly selecting a new connection because an agent that is trading frequently[4] is more likely than random to have a high production capacity.

In addition to measuring changes in performance, I measured various structural properties of the networks as they evolved as a result of the four AON strategies. Figure 6.3 shows these measurements as a function of time. Other than the push referral-based strategy, the AONs have a tendency to disconnect the network, evidenced by the large increases in diameter shown in Figure 6.3 (a). Interestingly, *Q/minNeighbor/pushMax* also increases

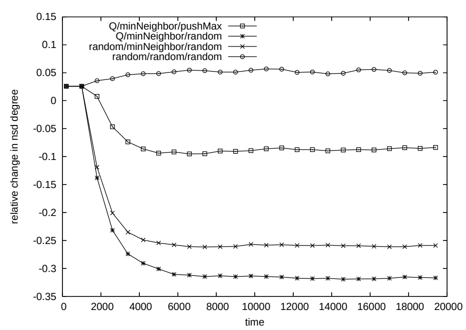
[4]Recall that the value of connections is based on frequency of trading.



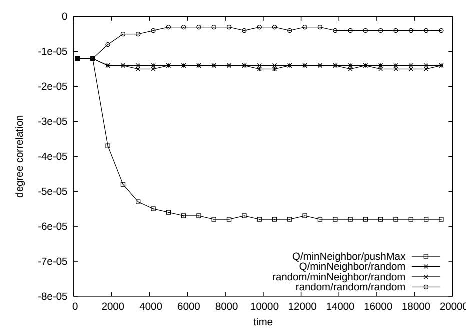
(a)



(b)



(c)



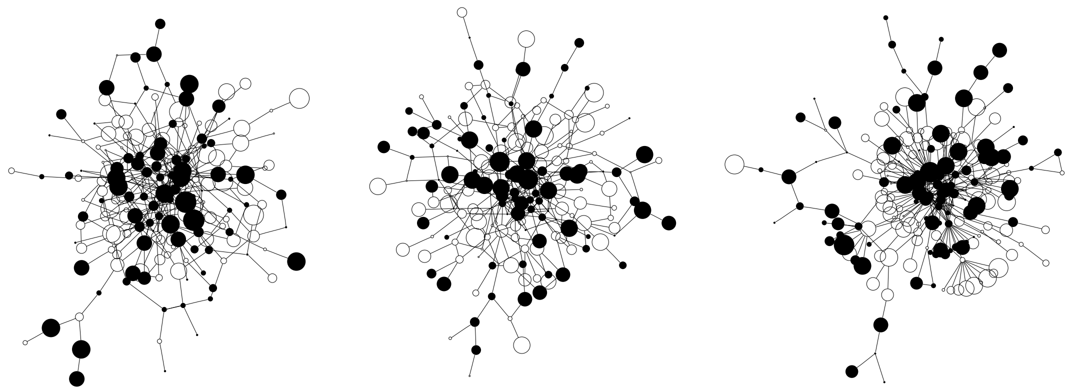
(d)

Figure 6.3: The evolution of the network statistics for the four AON strategies averaged over 200 iterations. The four statistics shown are: (a) relative diameter, (b) relative clustering, (c) relative normalized standard deviation (nsd) degree, and (d) absolute degree correlation. The symbols used to represent the strategies in the figure were selected to match the symbols of the three strategies in Figure 6.2.

the diameter of the organizational network even though it is guaranteed to maintain connectivity. This increase in diameter can be interpreted as a “spreading out” of the network. Presumably this spreading effect reduces trading competition outside of local areas of the network.

For the three AON strategies other than *random/random/random*, there is an apparent downward trend in clustering. While decreased clustering may be indicative of increased performance, the results are inconclusive, since the trends are not smooth even after being averaged over 200 simulations. A stronger trend is observable in the degree distributions and the degree correlations among the agents.

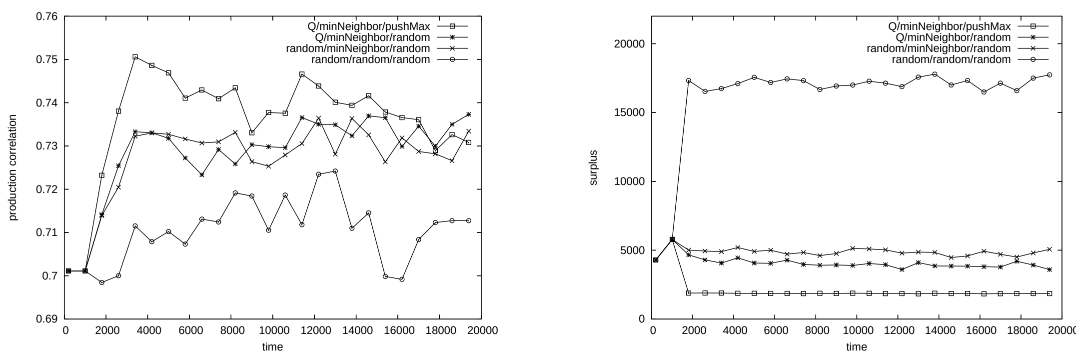
The strategies with higher performance decrease the normalized standard deviation of degree, moving the organizational network toward a uniform degree distribution. This effect is weaker when the agents use *Q/minNeighbor/pushMax*. Finally, the degree correlation among the agents decreases when the agents use the *Q/minNeighbor/pushMax* strategy. This suggests that, while the organization is moving toward a flatter degree distribution, more lower-degree nodes are connected to more higher-degree nodes. This supports the claim of a localizing, or buffering, effect. That is, pockets of trading activity are localized with few connections to the rest of the organization. The spreading and localizing effects of the *Q/minNeighbor/pushMax* strategy are further emphasized by the representative resulting network shown in Figure 6.4 (c). In Figure 6.4, there is little discernible structure in the networks resulting from *Q/minNeighbor/random* and *random/minNeighbor/random*, largely due to the random policy for establishing a new connection. By contrast, there is a discernible hierarchical structure in the network resulting from *Q/minNeighbor/pushMax*, consistent with the localizing and spreading effects discussed above.



(a) (b) (c)

Figure 6.4: Resulting networks from three of the four AON strategies: (a) *random/minNeighbor/random*, (b) *Q/minNeighbor/random*, and (c) *Q/minNeighbor/pushMax*. The network resulting from *random/random/random* is not shown, since it is indistinguishable from the static structure shown in Figure 6.1. The fill of the nodes in the networks represents which good is produced by the agent at that node; the size of the node corresponds to the agent's production capacity.

Before moving on to alternative AON strategies, there are two model characteristics that support the discussion of performance increases resulting from AONs. Figure 6.5 shows the evolution of two model-centric statistics over time, averaged over 200 simulations. The first statistic is production correlation, which is calculated in the same way that degree correlation is calculated. Production correlation measures the amount of complementary production capacity among neighboring agents. That is, production correlation is high if agents that produce good one are neighbors with agents that produce similar amounts of good two. Although the trends are not smooth, there is a noticeable increasing trend in production correlation for the AON strategies other than *random/random/random*. This result suggests that the set of connections that result from the more intelligent AON strategies may streamline the trading process. When production correlation is higher, there is more opportunity for direct trading among the agents.



(a) (b)

Figure 6.5: Two model-specific characteristics as they evolve as a result of the four increasingly intelligent AONs over time: (a) the production correlation between connected agents and (b) the average surplus. See the text for details.

The second statistic is surplus (i.e., the number of goods that remain after the market is cleared). Surplus corresponds to the amount of unused goods that remain in the organization after each agent clears its goods. Surplus can be used to support the performance results of the AONs. Clearly, the higher-performing AONs are able to reduce the amount of surplus in the organization, further suggestive of the trading efficiency of the resulting network structures.

I have shown that AONs based on the general AON framework increase the collective performance of the multi-agent production and exchange economy. In the next two sections, I will develop and discuss additional AON strategies more tailored to the production and exchange environment.

6.3.2 Using Agent Production Capacities

One of the properties of successful AONs discussed in the previous section was the ability to increase the production correlation among the agents in the network, that is, the ability to pair agents with complementary production capacities. Given this finding, it is easy to design an AON strategy that exploits this behavior.

Building on the best strategy from the previous section, the *Q/minNeighbor/pushProd* AON strategy uses Q-learning to determine when to adapt; removes its minimum-valued connection; and takes a push referral from the losing agent based on the losing agent's neighbors' production capacities. More specifically, if agent i, a producer of good one, is rewiring its connection with agent j, then agent j refers its neighbor k such that

$$k = \max_{l \in N_j(G)} \Delta g_2. \tag{6.12}$$

The performance of the organization of agents using this AON strategy is shown in Figure 6.6. Note that *Q/minNeighbor/pushProd* yields approximately a 4% increase in performance over *Q/minNeighbor/pushMax*. This increase is due directly to the fact that this AON is designed specifically for the production and exchange environment.

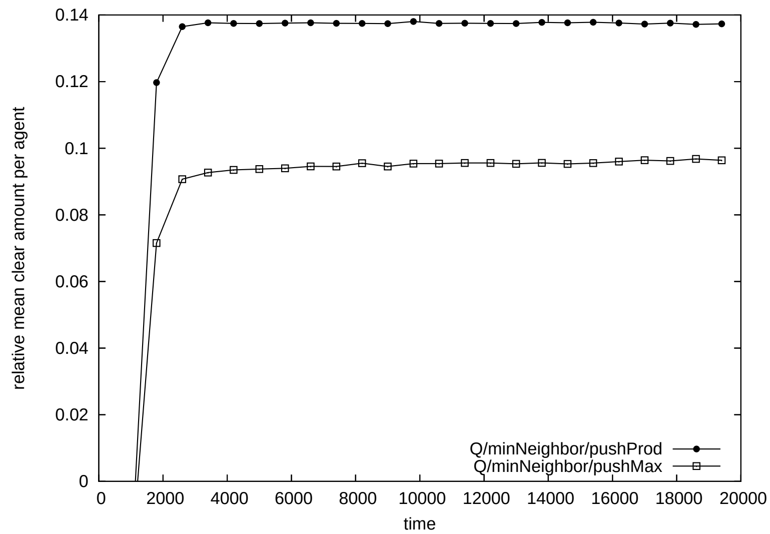


Figure 6.6: The relative performance of the organization of agents using *Q/minNeighbor/pushProd*. The 95% confidence interval for *Q/minNeighbor/pushProd* is 0.007. The results for *Q/minNeighbor/pushMax* are replicated for convenience.

Table 6.2 shows the structural statistics of the networks resulting from the agents using the *Q/minNeighbor/pushProd* AON strategy. The most significant structural property observed is the large relative decrease in clustering. The 90% decrease in clustering means that the *Q/minNeighbor/pushProd* strategy removes almost all of the clustering in the network. The removal of clustering from the network also corresponds with the slight decrease in diameter and the large increase in normalized standard deviation of degree. All of these measurements suggest that this strategy is removing local competition and moving the network structure toward one with many long range correlations. The value of long range correlations is the ability to shift goods around the network quickly since all of the network

statistic	*Q/minNeighbor/pushProd*	*Q/minNeighbor/pushMax*
relative diameter	-0.012	0.027
relative clustering	**-0.904**	-0.021
relative nsd k	0.535	-0.084
degree correlation	-9.2×10^{-5}	-5.8×10^{-5}

Table 6.2: Structural statistics for the networks resulting from agents using *Q/minNeighbor/pushProd*. The statistics for *Q/minNeighbor/pushMax* are shown for comparison. The bold value for clustering is for emphasis.

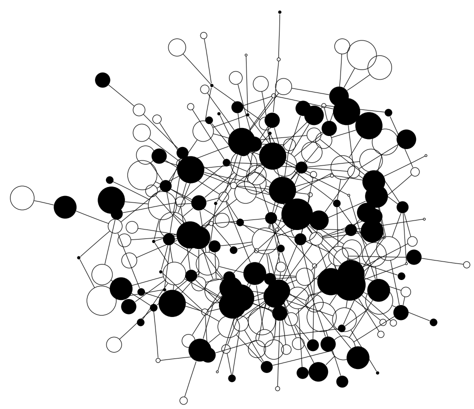


Figure 6.7: A sample resulting network from the *Q/minNeighbor/pushProd* AON strategy. Note the absence of a closely connected core, compared to the networks shown in Figure 6.4. The fill of the nodes in the network represents which good is produced by the agent at that node; the size of the node corresponds to the agent's production capacity.

is "local." This structure is seen in Figure 6.7, in which there is no closely connected core of the network. For comparison, see Figure 6.4.

6.3.3 A Threshold-Based Alternative

In this last section on AONs for the production and exchange economy, I offer an alternative to the stateless Q-learning mechanism that agents use for deciding when to adapt. Similar to the strategy used in the previous section for determining the agent with which to establish a new connection, the alternative method for deciding when to adapt uses the values of the current connections. Recall that the values of an agent's current set of connections are updated using an exponentially weighted moving average, where

$$V_{ij} \leftarrow V_{ij} + \beta[W_{ij} - V_{ij}], \tag{6.13}$$

with W_{ij} being the number of trades across the connection from i to j in the current time step and β being the learning, or smoothing, rate.

Intuitively, an agent is likely to be productive (i.e., to regularly clear a large amount of goods) if the agent trades with its neighbors frequently. When an agent trades with its neighbors frequently, the values for the connections with these neighbors are high. Given this, an agent can use a threshold on the values of connections to determine when to rewire. That is, an agent i decides to adapt when $\exists j \in N_i(G) : V_{ij} < \Theta$, where Θ is a threshold parameter. In essence, an agent using this strategy decides to adapt its connectivity when one of its connections becomes effectively useless. The obvious method for subsequently deciding which connection to rewire is the minimum-valued connection. Finally, because of its high performance in the previous experiments, the adapting agent takes a push referral from the losing agent where the losing agent refers its maximal-valued neighbor. This combined strategy is denoted *threshold/minNeighbor/pushMax*.

Figure 6.8 shows the performance of an organization of agents using the threshold-based AON strategy. The figure replicates the results for *Q/minNeighbor/pushMax* and

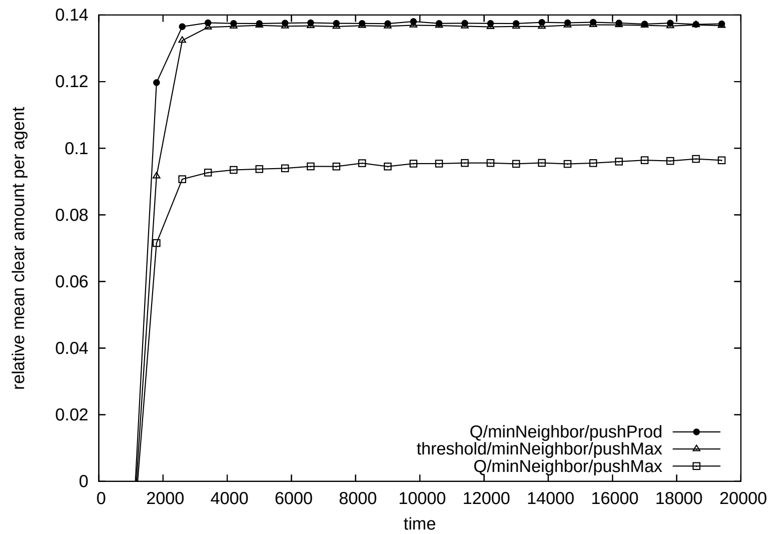


Figure 6.8: The relative performance of the organization of agents using *threshold/minNeighbor/pushMax*. The 95% confidence interval for *threshold/minNeighbor/pushMax* is 0.006. The results for *Q/minNeighbor/pushMax* and *Q/minNeighbor/pushProd* are replicated for comparison.

Q/minNeighbor/pushProd for comparison. The threshold-based AON strategy outperforms the Q-learning based strategy that uses the same criteria for selecting a connection to rewire and determining where to establish a new connection. In hindsight, this is an obvious result, because the agents using the threshold strategy are only required to learn the value of connections, while agents using *Q/minNeighbor/pushMax* are required to learn both the value of connections and when to adapt.

The major structural difference in the networks from *threshold/minNeighbor/pushMax* compared with the networks from *Q/minNeighbor/pushMax* is the much larger decrease in

statistic	*threshold/minNeighbor/pushMax*
relative diameter	0.010
relative clustering	-0.465
relative nsd k	0.037
degree correlation	-6.0×10^{-5}

Table 6.3: Structural Statistics for the networks resulting from agents using *threshold/minNeighbor/pushMax*.

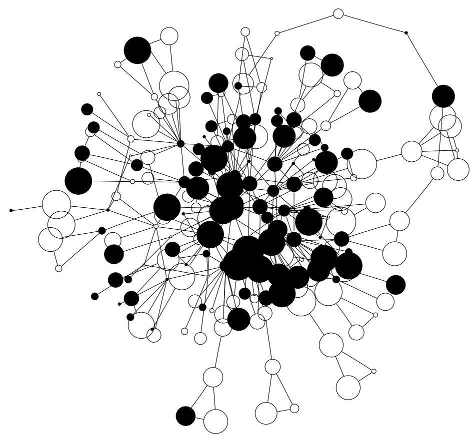


Figure 6.9: A sample resulting network from the *threshold/minNeighbor/pushMax* AON strategy. Note the absence of a closely connected core, compared to the network shown in Figure 6.4. The fill of the nodes in the network represents which good is produced by the agent at that node; the size of the node corresponds to the agent's production capacity.

clustering. This can be seen by comparing the values in Tables 6.3 and 6.2. The decrease in clustering can also be seen in the representative network structure shown in Figure 6.9, where there is a clear lack of a highly connected core (i.e., the "center" of the network is not filled with nodes). Notice that the hierarchical structure observed in the networks that result from *Q/minNeighbor/pushMax* is preserved in the networks that result from *threshold/minNeighbor/pushMax*.

By being direct about when to rewire, the *threshold/minNeighbor/pushMax* AON strategy is able to match the performance of the highest-performing strategy discussed above: *Q/minNeighbor/pushProd*. This is suggestive of the fact that domain-specific AON strategies – strategies that exploit the inherent structure of the environment – are likely to be more successful than AON strategies that are more general. At the same time, the generalized approach leads to a statistically significant increase in organizational performance.

6.4 Concluding Remarks

In this and the previous chapter, I have demonstrated the ability of AONs to significantly improve organizational performance in two general multi-agent environments: a team formation environment and a production and exchange market. In both environments, AON strategies based on the general AON framework presented in Chapter 4 provided significant gains in organizational performance. The results are suggestive of what is possible in many other multi-agent system domains, many of which are analogous to team formation or a market economy. In the next chapter, I develop and apply AONs in two specific application domains: supply chain management and sensor networks.

Chapter 7

Applications of Agent-Organized Networks

> *To achieve the oft-expressed visions of dynamically forming and dissolving business interactions requires automated support for* **supply chain formation***, the process of bottom-up assembly of complex production and exchange relationships.*
>
> Walsh & Wellman (2000)

> *The real goal is to explore the uses of intelligent sensors, a technology whose promise suddenly seems huge. The applications for this "embedded intelligence" are vast and profound. Eventually large swathes of the earth will communicate with the digital realm using millions of miniature sensors.*
>
> Benjamin Fulford

7.1 Supply Chain Networks

In recent years, supply chain management has enjoyed a renewed interest from researchers in economics, operations research, information science, and computer science. This is largely a result of large-scale communications systems, such as the Internet, that are redefining the way supply chains can be managed and the role of information in managing supply chains. In addition, increased computational power facilitates the automation of many supply chain management functions.

A central problem in improving supply chain management is mitigating the "bullwhip effect" (Lee, Padmanabhan, & Whang 1997), where distortions in supply and demand propagate and increase upstream in a supply chain. Various approaches have been proposed for mitigating the bullwhip effect, most based on methods for improving information sharing across the stages of a supply chain. A noteworthy example of how artificial intelligence has been applied to mitigating the bullwhip effect is in the use of genetic algorithms (GAs) for evolving ordering rules based on past performance in the MIT Beer Game (Kimbrough, Wu, & Zhong 2001). The GAs discover effective ordering policies for deterministic demand, stochastic demand, and stochastic lead time in a linear, four stage supply chain.

Supply-chain management is an ideal environment for the application of agents and multi-agent systems (Fox, Barbuceanu, & Teigen 2000). Agents can provide many functions in supply chains, including acquiring orders from customers, coordinating the internal logistics of a factory, assigning and scheduling transportation resources, and managing inventories and purchasing.

Further evidence for the utility of autonomous software agents in supply chains is provided by the Supply Chain Trading Agent Competition (Arunachalam *et al.* 2004). This competition, held annually over the past several years, provides a proving ground for agent technology. To date, the competition has focused on the design of a single agent to serve as an automated intermediate level agent in a supply chain.

Another widely studied problem related to supply chains is supply network formation. As mentioned above, research on multi-agent systems and supply chain management focuses on the behavior of individual agents and how they can better handle order processing, inventory management, and logistics at a single location in a supply chain. Supply network formation, and reformation, is the study of how collections of interconnected supply chains form, interoperate, and adapt. Motivating the need for understanding supply network formation are concerns for efficient distribution of goods in supply networks and survivability of supply networks that are subject to failures or unexpected changes (Thadakamalla *et al.* 2004).

In much of the literature on supply network formation, the network formation process is top-down. That is, most studies assume that all of the agents in two adjacent stages, or levels, of the supply network are connected, and the use of these connections is refined over time. Another trend in the literature is the study of relatively small supply networks. After surveying the existing literature on supply network formation, the remainder of this section will focus on the bottom-up formation of relatively large supply networks.

Related Work

One approach to supply network formation is termed supply chain configuration (Emerson & Piramuthu 2004). In this approach, agents use a classifier system[1] to select an agent in the adjacent upstream stage with whom to place orders. The experiments demonstrated the approach on a two-stage supply network with three-agents and a three stage supply network with six agents. While the networks examined were small in scale, the study motivates adaptive configuration of supply networks.

System dynamics have been used to study emergent supply networks, where suppliers and customers have heterogeneous preferences for short- or long-term performance (Akkermans 2001). This is one of the largest studies of supply networks, with 1000 nodes and four stages. The model successfully replicates the bullwhip effect. The major finding is that preferences for both short- and long-term performance result in stable relationships. Again, this study focused on top-down supply network formation by specifying preferences for performance and refining the set of all possible connections over time.

The effects of various order-filling policies were examined for refining a full set of connections among the agents in adjacent stages of a small-scale supply network (11 agents and five stages) (Schieritz & Grobler 2003). The results are similar to many other studies in that the various order-filling policies led to refined sets of connections over time.

In two studies closely related to supply network formation, specific algorithms were

[1]The decision tree classifier C5.0 was used in this particular study (Emerson & Piramuthu 2004).

applied for determining the interconnectivity among the agents in supply networks. In one, algorithms for selecting among suppliers based on *a priori* lead times demonstrate improved performance for single agents, but no network-wide studies were conducted (Zeng & Sycara 1999). In the other, *hierarchical task decomposition* was applied to determine the participants and structure of a supply chain given an initial set of supply and demand requirements (Walsh & Wellman 2003). The two methods provide strong theoretical foundations for supply chain formation and hold promise for application to large-scale supply network formation.

7.1.1 Modeling Multi-Agent Supply Chain Networks

To evaluate the usefulness of AONs for bottom-up supply network formation, I synthesized several models from the recent literature (Zeng & Sycara 1999; Akkermans 2001; Schieritz & Grobler 2003; Kimbrough, Wu, & Zhong 2001). For various reasons, no single model supports the study of AONs for supply chain formation. The reasons include the assumption of top-down supply chain formation; overly complicated price formation, bidding, or ordering dynamics; and scale. My goal is to demonstrate that the efficiency of a supply network can increase as a result of individual agents autonomously adapting their connectivity online in a large supply network.

The model consists of L levels, or layers, of a supply network, where each layer contains some number of agents. In all of the experiments described below, $L = 4$, with the

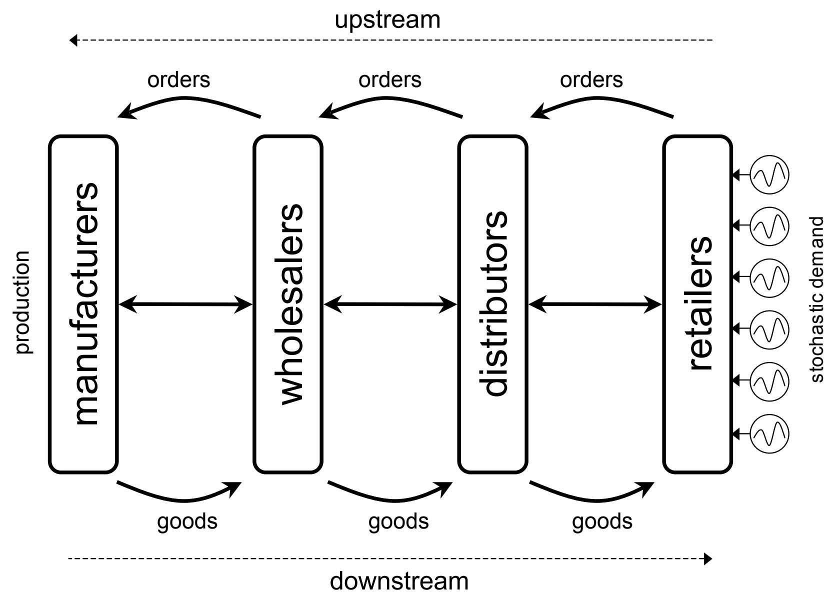


Figure 7.1: A characterization of a large-scale supply network. The network has four stages, or levels. Orders flow upstream from retailers to manufacturers. Goods, based on orders, flow downstream.

layers comprising retailers, wholesalers, distributors, and manufacturers (Kimbrough, Wu, & Zhong 2001). Each agent in each layer except the final layer, manufacturers, is connected to E agents in the next layer of the supply chain. Direct interactions among the agents only occur across these direct connections.

The demand in the model is generated by the retailers, where the demand for each retailer is stochastic (i.e., a random walk, with a “step” taken at each iteration, capped by a maximum value) (Kimbrough, Wu, & Zhong 2001). This demand is passed through the network in the form of orders, which can be aggregated at intermediate levels (i.e.,

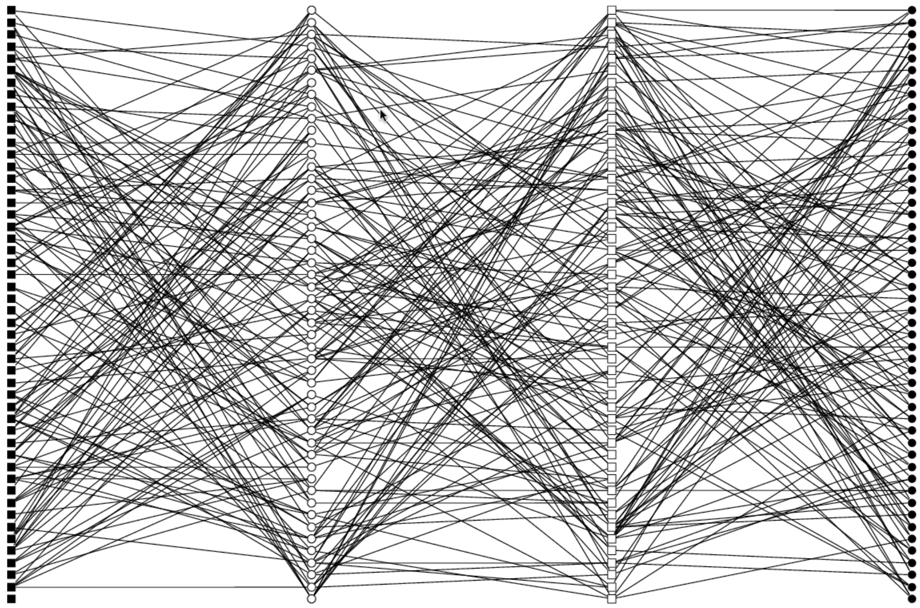


Figure 7.2: A rendering of a relatively large-scale supply network. At each level, or stage, there are 50 agents. The connections among the agents in adjacent levels are assigned randomly.

wholesalers and distributors). That is, a retailer agent generates an order based on its current demand and passes it along to one of its wholesalers. For simplicity, orders are not split, but switching among upstream agents based on performance is included (Zeng & Sycara 1999). Orders can only be sent from one agent to another if there exists a direct connection between the two agents. As depicted in Figure 7.1, *upstream* is the direction in which orders flow and *downstream* is the direction in which goods flow.

During each iteration of the model, the agents in the retailers level update, followed by the agents in the distributors level, and so forth upstream. Within each level, the agents update in a random order. When the retailer agents update, they simply generate orders based on their current demand. Upstream, the update process is more complicated.

The agents in all of the upstream levels maintain an inventory of goods. An agent's inventory is directly related to the way in which orders are processed and generated. For all agents, orders are processed in "first in, first out" (FIFO) order (Schieritz & Grobler 2003). All upstream agents keep track of the total quantity of the goods ordered from them during the current time step. I will call this value $o_{in}(t)$. After receiving all incoming orders for the current time step (recall that all agents downstream have already updated), orders are processed using the existing inventory until there is not enough inventory to fill an order. Once inventory is depleted during a time step, the agent orders goods from one of its upstream connections. The amount an agent orders is based on an "exponential smoothing of recent customer orders" (Akkermans 2001):

$$o(t) = o(t-1) \cdot \theta + o_{in}(t) \cdot (1-\theta). \tag{7.1}$$

The assumption here is that an agent should order enough goods to meet its expected demand. When an agent has an inventory i after processing orders, the amount of the inventory is subtracted from the expected demand to arrive at the size of the outgoing order to avoid ordering more goods than are necessary:

$$o_{out}(t) = o(t) - i. \tag{7.2}$$

Another component of ordering is the way in which agents select among their providers for placing orders (recall that orders are not split). Following other models (Akkermans

2001; Schieritz & Grobler 2003), the agents decide which agents to order from based on reputation. In my model, this reputation is measured as *lag time* or the time it takes an agent to fill an order. In this model, lag time is highly dynamic (especially when the topology of the network is changing); as a result, the agents require only a very short memory of the lag times of their suppliers. In particular, the agents keep track of the most recent "time to fill" an order for each of their suppliers.[2] The agents place an order with one of their connected suppliers by selecting the supplier with the smallest lag time. Ties are broken randomly.

In considering the method just described for selecting suppliers with whom to place orders, outstanding orders (i.e., orders that have not yet been filled) cause a problem. Suppose an agent's chosen supplier is able to fill its first order immediately. The agent will select that same supplier again, assuming that the other suppliers are at least slightly slower in filling orders. Now suppose that the "quick" supplier is selected again, but that that supplier is no longer able to fill orders at all (perhaps because of failure or disconnection). If the ordering agent does not factor outstanding orders, the ordering agent will continue to order from the same "quick" supplier based on the one order that the supplier was able to fill quickly. To alleviate this problem, the lag times tracked by ordering agents are incremented at every time step for every supplier with whom there is an outstanding order.

The last component of the model is production by the manufacturers. The manufacturers, or more generically, the agents in the most upstream level of the supply network,

[2] At first, this may seem like a large assumption, but a large number of experiments support this as an effective method for placing orders.

have no agent with whom to place orders. As a result, these agents are endowed with the ability to directly produce goods. These agents produce goods using the same method as the intermediate-level agents use to order goods (Equation (7.2)). The two differences are that each manufacturer has a maximum production capacity per time step X and produced goods are available after only one iteration. This is in contrast to the orders placed by agents in the middle levels of the supply network. Those agents can order any number of goods during an iteration, but the time to fill those orders is indefinite.

The Bullwhip Effect

As mentioned in the introduction to this section, the "bullwhip effect" is a commonly observed phenomenon in supply networks. The bullwhip effect is a phenomenon where variances in orders increase upstream in supply chains (Lee, Padmanabhan, & Whang 1997). To qualitatively validate my synthesized model of a supply network, I conducted experiments to determine if the bullwhip effect was observable in the model.

Figure 7.3 shows evidence of the presence of the bullwhip effect when the model is set in motion. The demand curve represents the sum of the actual demand (i.e., the sum over all of the demands of the individual retailers) which are determined by a random walk. This demand curve is equivalent to the orders of the retailers.

Clearly, the orders of the wholesalers are more variable than the orders of the retailers. Similarly, the orders of the distributors are more variable than the orders of both the

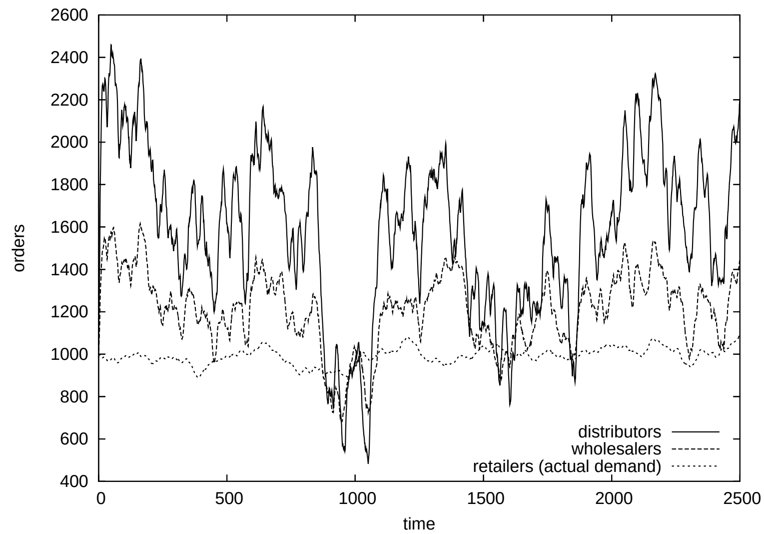


Figure 7.3: An observation of the bullwhip effect for a supply chain with four layers and 50 agents in each layer. The topology of the supply network was generated randomly (see Figure 7.2).

wholesalers and the retailers. The production of the manufacturers is omitted, since their production capacities are limited. The result suggests, qualitatively, that the model reflects behavior of real supply networks.

Measuring Performance

Many different measurements could be used for evaluating performance in supply chain formation. In the experiments below, because of the focus on the efficiency of supply network structures, I use **the ratio of cumulative consumption to cumulative demand**. This ratio directly measures how effective a supply network is at distributing goods and alleviating bottlenecks or choke points (i.e., agents where many incoming orders are received but few outgoing orders are filled).

When the ratio of total consumption to total demand is one, all of the demand at the retailer level has been satisfied by goods flowing downstream from the manufacturers. Deviations from unity in the ratio of consumption to demand result from the various dynamics at work in the supply network model. First, the stochastic nature of demand introduces variability in demand over time, with increasing variance at upstream levels of the network (e.g., the bullwhip effect). Additionally, the topology of the supply network in conjunction with the decisions that agents make about where to send orders also introduce variability. If many wholesalers all attempt to order from the same distributor, this can cause increased lag time if the distributor's inventory is low. Furthermore, the distributor is reliant upon manufacturers that may be ill prepared for increased demand, or overwhelmed with current orders.

7.1.2 An AON for Supply Network Formation

The supply network model moves from a static supply network with interesting dynamics to a supply network **formation** environment when the agents are given the ability to autonomously adapt their upstream connectivity. This extension to the supply network model above lends itself to the general AON framework described in Chapter 4. In fact, I apply the general AON framework directly to allow the agents to learn and to adapt their connectivity based on what they learn over time.

The general AON framework described in Chapter 4 requires additional information to be useful in any specific domain. In particular, the reward function used for updating the action value function must be specified. As with several other environments examined in this dissertation, the action set for the supply chain formation model is $A = \{rewire, nothing\}$. Given this action set, the agent employs stateless Q-learning to maintain values for taking each action. The reward function used in the application of the general framework to supply network formation is simply the minimum lag time for any upstream provider. Therefore, an agent adapts when rewiring historically results in lower lag time (for the "best" provider) than the lowest historical lag time while the agent is not rewiring.

An additional detail required to understand the AON learning and rewiring is the duration of time during which the agents give credit to the rewiring action. To reiterate, an agent that decides to rewire must adjust its value for rewiring for some period of time following the rewiring, after which the agent switches back to the "nothing" action until another rewiring action takes place. This duration D_r is a free parameter. Through extensive experimentation and exploration of the model, I arrived at $D_r = 10$ for all agents. Note that many other values result in effective AON strategies.[3]

In addition to the learning and rewiring methods, it is also necessary to specify how an agent selects connections to remove and establish. To prevent the development of and need

[3]This is an area left for future work: developing a systematic methodology for determining when the value of a rewiring action is no longer the value of rewiring, but rather the value of doing nothing.

for additional information, a rewiring agent removes its connection that has the longest lag time. This information is already being acquired and used for placing orders. Therefore, using this information for the AON adaptation policy requires no extra effort or computation on the part of the agents.

Finally, agents select agents for new connections randomly. There are various reason for choosing this simple method. First and foremost, the agents within any level of the supply network do not interact or know about one another in the supply network model. This prevents any sort of referrals for network adaptation. While other methods are possible, such as using information about old suppliers, the simple random strategy is employed here to demonstrate the utility of AONs.

7.1.3 Experimental Results and Discussion

In this section I describe experiments and present several results for applying AONs to the supply network formation model described above. All of the results presented in this section are the average over 25 simulations of a parameterized version of the supply network formation model. Due to the special, layered, structure of supply networks, the network structural statistics used to to understand the behavior of AONs in previous chapters cannot be applied to understanding AONs for supply network formation. This is discussed further in the presentation of the experimental results.

To understand the utility of individual agents using the learning-based AON strategy, two benchmarks were included in the experiments. The first benchmark is the network with no adaptation. When there is no adaptation, the agents do not adapt their connectivity structure, but they do select among their existing connection for order placement. The second benchmark is a purely random AON. As was done for previous experiments presented in earlier chapters, the random adaptation policy was calibrated to cause approximately the same number of adaptation as the learning-based AON. Recall that an agent employing the random AON strategy adapts with a specified probability and rewires a randomly select existing connection to a randomly selected, unconnected agent.

Another control function included in the experiments was to start the networks in "equilibrium." That is, the network is initialized so that the sum of the inventory at a level in the network equals the sum of the demand of the downstream level. This initial equilibrium was validated by creating a set of independent, path unique supply chains (i.e., one retailer connected to one wholesaler connected to one distributor connected to one manufacturer with no overlapping connections). When such a network was initialized in equilibrium, the performance never deviated from perfect (i.e., supply always equaled demand). While the networks in the experiments start in equilibrium, the decisions of individual agents for placing orders in more interconnected networks quickly move the system out of equilibrium.

The first of the two representative experiments begins with a network configuration similar to that shown in Figure 7.2. In this experiment, there are equal numbers of retailers, wholesalers, distributors, and manufacturers, and every agent has the same number of upstream connections, assigned at random, initially. Table 7.1 gives the values of the various model parameters for each of the two experiments discussed in this section.

description	symbol	value (Ex. 1)	value (Ex. 2)
network levels	L	4	4
agents per level	N	50	50/60/80/100
upstream connections per agent	E	3	4
initial demand	D_0	20	5
maximum demand	D_{max}	40	10
(uniform) initial inventory	i_0	20	10
expected demand weight	θ	0.9	0.9
(uniform) max production capacity	X	20	10
AON learning rate (min)	α_{min}	0.05	0.05
AON learning rate (max)	α_{max}	0.4	0.40

Table 7.1: The values of the various model parameters for each of the two experiments discussed in this chapter. The main difference between the two experiments is the structural constraints of the networks.

Recall that the agents that do not adapt their network connectivity are still able to learn. In particular, they can learn which of their connections to rely on for the timely delivery of goods. This is observable in the experimental results shown in Figure 7.4. There is an initial increase in the ratio of consumption to demand, but this increase is quickly overrun by order backlogs. That is, as time goes on, the static supply network is not able to keep up with the demand generated by the retailers. This is evidenced by the decreasing trajectory of the performance curve over time.

Unlike the network of agents that do not adapt their network connectivity, the networks

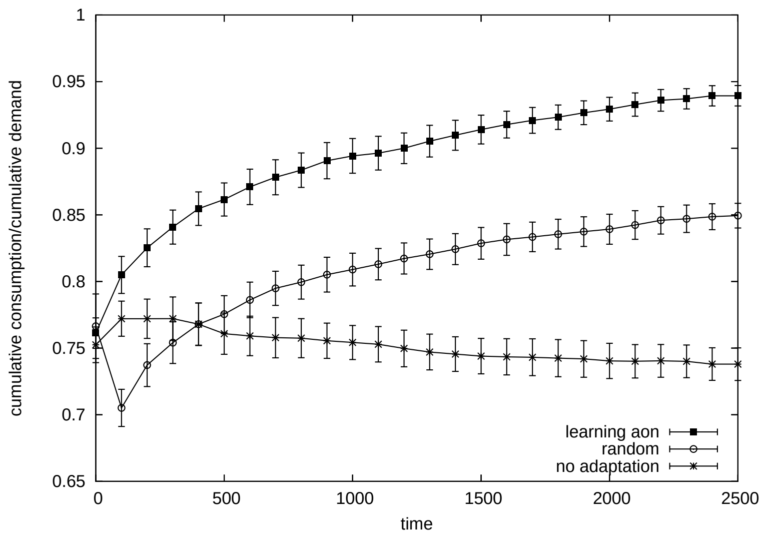


Figure 7.4: The experimental results comparing networks of agents using AONs for adapting connectivity to networks of agents that do not adapt connectivity. These results are for a model with equal numbers of retailers, wholesalers, distributors, and suppliers. The parameters for the underlying model are given in Table 7.1. Each point represents the average of 25 simulations with the errorbars showing 95% confidence intervals.

of agents that do adapt their network connectivity are able to increase performance over time. This is obvious from the results shown in Figure 7.4. Notably, agents using the random AON strategy are able to provide a statistically significant increase in performance over the static networks. This suggests that flexibility, or fluidity, in the network is more beneficial than static long-term relationships among the agents. I conjecture that this is a result of the spreading around of demand by random adaptation.

While the random AON strategy allows the agents to improve the collective efficiency of the supply network, the learning-based AON strategy provides an even greater increase in performance. As seen in Figure 7.4, the network of agents using the learning-based AON yields a 27.2% increase in performance over the static network. This is a significant increase in its own right, but may be even more significant given the economic implications of supply networks.

As mentioned above, the structural statistics used to understand the behavior of the AONs in previous chapters are not applicable in the supply network formation model. The one measure that had some promise was the normalized standard deviation of degree for incoming connections. I hypothesized that the random initial configurations were creating skew in the distribution of the number of incoming connections for agents in the upstream levels of the supply network. As it turns out, measurements of normalized standard deviation refuted this hypothesis. Furthermore, the measurements taken provided no evidence of structural trends for the supply networks that were adapted by the agents. Therefore, at this point, little can be said about the structural properties of the supply networks with increased performance. The results for the random AON provide evidence of the fact that increased performance may be more of a result of adaptation than of the structure that results from adaptation. This is an area that is left as future work.

In conducting experiments with the supply network formation model, I thought it important to include experiments on supply networks with variable numbers of agents in each level. Figure 7.5 shows the result of one such experiment where the number of agents in each level decreases moving upstream. The results are qualitatively similar to those of the

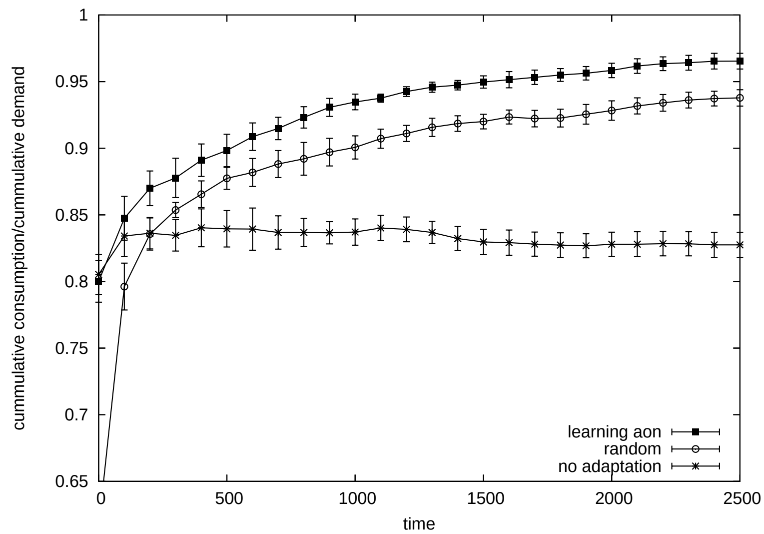


Figure 7.5: The experimental results comparing networks of agents using AONs for adapting connectivity to networks of agents that do not adapt connectivity. These results are for a model with decreasing numbers of agents in each upstream level of the supply network. The parameters for the underlying model are given in Table 7.1. Each point represents the average of 25 simulations with the errorbars showing 95% confidence intervals.

first experiment discussed above. One observable difference between the two experiments is that overall performance levels are higher in the second experiment. This suggests that the structure where the number of agents decreases with each upstream level of the supply network has simpler dynamics. This makes intuitive sense, because the demand is aggregated moving upstream in the network and there are fewer choices in the upstream levels.

The experimental results for AONs applied to the supply network formation domain demonstrate the benefit of agents adapting their connectivity in dynamic supply networks. While these experiments support the use of AONs for supply networks, the results are not surprising, since the model is in some ways similar to the production and exchange model considered in Chapter 6. In the next section, I apply the AON concept to the problem of topology control in wireless ad hoc networks, an environment that is significantly different from any environment considered thus far.

7.2 Topology Control in Sensor Networks

Sensor networks are rapidly becoming an important area of research for the multi-agent systems community. Applications of sensor networks include environmental monitoring, structural modeling, disaster management, health care, manufacturing, and vehicle monitoring (Culler, Estrin, & Srivastava 2004; Horling, Mailler, & Lesser 2004). Sensor networks can be either wired or wireless. Wireless sensor networks present several unique challenges, including network connectivity among sensors (i.e., agents) situated in some physical space. Although the physical space largely determines the connectivity of a sensor network, there are aspects of connectivity over which the agents have control, such as transmission power.

Some sensor networks are comprised of homogeneous agents, but more realistic environments are made up of heterogeneous agents. Many different types of agents can be included in sensor networks, including sector managers, data collectors, data routers, and end point sensors. In these situations, the role that an agent takes on and the interconnectivity of the agents is important for the overall efficiency of the network. Understanding the effects of interconnectivity will be essential to designing and ensuring the effectiveness of multi-agent sensor networks. Furthermore, strategies and policies for adjusting network connectivity among heterogeneous agents in a sensor network may lead to improved performance, fault tolerance, and extended network lifetime (Horling, Mailler, & Lesser 2004; Culler, Estrin, & Srivastava 2004).

There are many aspects of wireless sensor networks that can benefit from agent-based approaches. One such area is agent-based routing (Gan, Liu, & Jin 2004), where agents manage the flow of data through the wireless topology. Another aspect of sensor networks that can benefit from agent-based approaches is topology control. For the remainder of this section, I will focus on the problem of distributed topology control. Topology control is the problem of distributed management of the connectivity among the sensors, or agents, in an ad hoc network

7.2.1 Overview of Topology Control

There are several important goals in the design of wireless ad hoc sensor networks, including ad hoc deployment, energy constraints, and unattended operation (Cerpa & Estrin 2004). Each of these design goals implies constraints on the behavior of individual sensors, or agents, in wireless networks.

First, the way in which the sensors, or nodes, are placed in physical space is assumed to be random. That is, in many applications, the designers of a wireless sensor network do not have direct control over the locations of the individual sensors (Cerpa & Estrin 2004; Rodoplu & Meng 1999). Additionally, the sensors randomly deployed into a physical space must also be equipped to deal with environmental factors that may result from their positioning or changes to the environment.

Another major concern with wireless sensor networks is energy efficiency (Jones *et al.* 2001; Rajaraman 2002). Energy efficiency is desirable primarily because lower consumption of power allows for extended network lifetime without outside intervention. The sensors deployed in a wireless network have limited battery power; the longer they can preserve that power, the longer they can function as members of the network.

Energy efficiency is closely related to the third design goal, unattended operation. Typical applications of sensor networks are for monitoring areas where it is difficult for humans to stay or frequently visit. Therefore, it is highly desirable that sensors in wireless networks are able to autonomously adapt to changing conditions such as failures or environmental changes.

In addition to the design goals described above, connectivity is an important property in many applications of wireless sensor networks (Rajaraman 2002; Ghosh *et al.* 2004; Pishro-Nik, Chan, & Fekri 2004). Connectivity is necessary in many applications because the sensors need to exchange information or need to forward information out of the sensor network to other locations. The task of distributed vehicle monitoring (Horling, Mailler, & Lesser 2004) is an example of a situation in which the agents must exchange information for success.

The problem of distributed topology control in wireless sensor networks is primarily focused on providing energy efficiency while maintaining connectivity in the network. Distributed topology control is also constrained by unattended operation and the arbitrary positioning of the sensors in the network. The problem of controlling topologies can be further compounded in heterogeneous networks where different sensors have different maximum transmission powers (Li & Hou 2004). Distributed topology control algorithms provide sensors in wireless networks with methods for managing their power consumption based on their perceived relative position in the network.

One class of topology control techniques is position-based algorithms (Rodoplu & Meng 1999; Li & Hou 2004; Song *et al.* 2004). The position information for each node is generally assumed to be acquired by the Global Positioning System (GPS), but other methods can provide position information (Song *et al.* 2004). One position-based approach is CTBC(α) (Li *et al.* 2001), a two-phase cone-based approach in which the region around a sensor is broken into cones. Power is increased until a neighbor sensor is within communication range in each of the cones. The second phase removes redundancy from the initial increase in power. CTBC(α) is representative of a class of topology control methods that use local distance information for controlling the transmission power and the topology of wireless networks. The basic premise in distance-based algorithms is for the sensors to initially transmit at maximum power and exchange position information. The sensors then use this position information to establish their transmission power to guarantee some level of connectivity and minimize power consumption (Wattenhofer & Zollinger 2004; Ghosh *et al.* 2004).

While position-based topology control is viable in many application domains, GPS or other positional information is not always available or reliable (e.g., inside buildings, in dense forests, or underwater) (Song *et al.* 2004). More general topology control techniques do not assume, or rely upon, positional information. The assumption of no positional information is the approach I will take later in this section in describing a topology control model and in applying AONs to general, distributed topology control.

Another approach to topology control is to allow sensors to move between states of being asleep (passive) and awake (active). So called "stochastic" sensor networks make a trade-off between connectivity and energy efficiency by allowing sensors to randomly move between the two states (Zhang *et al.* 2004b). In a similar approach, schedules for moving between the asleep and awake states are designed into the sensors (Erramilli, Matta, & Bestavros 2004). Another algorithm, ASCENT, prescribes methods for agents to move between passive and active routing regimes based on perceived information flow (i.e., packet throughput) in the network (Cerpa & Estrin 2004). Finally, Geographic Adaptive Fidelity (GAF) combines position-based topology control techniques with the asleep/awake state switching methods (Xu, Heidemann, & Estrin 2001).

As with position-based methods, the state switching techniques are useful in many applications. These application environment do not typically include highly dynamic environments and environments that require all agents to potentially pass information to one another regularly. In this light, for the remainder of this chapter, I focus on a general version of the topology control problem. This general model assumes that no positional information is available and that all agents must remain in active mode at all times.

7.2.2 The Topology Control Model

In order to study the application of AONs to topology control in wireless (sensor) networks, I have selected a generic model of multihop wireless networks. In this section, I describe the model. In subsequent sections, I describe one centralized algorithm and one distributed algorithm that will be used for comparison.

The basis for the model of topology control in multihop wireless networks is a graph where two nodes are connected if and only if the corresponding nodes can communicate. Following Ramanathan and Rosales-Hain (2000), I extend this base model to also include properties of the radios and an environmental propagation function. This is a widely adopted model in the sensor network and wireless network research community, although most other studies include the ability of the nodes to know and share positional information (Wattenhofer *et al.* 2001; Rodoplu & Meng 1999; Li & Hou 2004; Song *et al.* 2004). In the model used here, in order to preserve generality, there is no positional information available to the individual nodes in the network (Ramanathan & Rosales-Hain

2000). Additionally, the model used here for topology control is independent of the routing protocol (Rajaraman 2002) used for communication over long distances as that protocol has no impact on the topology control algorithms discussed below. Finally, I use the terms node, sensor, and agent interchangeably to refer to the computational components of the networks.

Definition 13 *(Ramanathan & Rosales-Hain 2000) A multihop wireless network is represented as* $M = (N, L)$*, where* N *is a set of nodes and* $L : N \to ([0, 1], [0, 1])$ *is a set of coordinates on the unit square denoting the location of the nodes.*[4]

The nodes, or agents, in the network also have an adjustable parameter for transmit power, where the transmit power of agent i is denoted p_i. The transmit powers of a pair of agents along with the propagation model for the environment determines if two agents can directly communicate in the wireless network.

Definition 14 *(Ramanathan & Rosales-Hain 2000) The propagation function is represented as* $\gamma : L \times L \to \mathbb{R}$*, where* L *is the set of location coordinates on the unit square.* $\gamma(l_i, l_j)$ *gives the loss due to propagation at location* $l_j \in L$*, when a packet is originated from location* $l_i \in L$*.*

Direct communication between two agents in the network depends on their corresponding transmit powers, the propagation function, and the receiver sensitivity S. That is, two

[4] In the original model, the coordinate space was $(\mathbb{Z}_0^+, \mathbb{Z}_0^+)$. The change is for convenience with no consequence as either coordinate space can easily be mapped to the other.

agents, i and j, can communicate directly if and only if

$$p_i - \gamma(l_i, l_j) \geq S \qquad \text{and} \qquad p_j - \gamma(l_i, l_j) \geq S. \tag{7.3}$$

Without loss of generality, I assume perfect receiver sensitivity (i.e., $S = 0.0$) in all of the experiments described below.

In the model, it is assumed that γ is a monotonically increasing function of the geographical distance, $d(l_i, l_j)$ between two points. This assumption holds true in the "free space propagation model" or when clutter in the environment results in a uniform loss in all directions (Ramanathan & Rosales-Hain 2000; Rappaport 1996). In all of the experiments below, I use the propagation function based on the "well-known generic model of propagation" (Rappaport 1996):

$$\gamma(d) = \begin{cases} 0.0, \quad \text{if} \quad d < d_{thr}, \quad \text{and} \\ 10 \cdot \varepsilon \cdot \log_{10}(\frac{d}{d_{thr}}), \quad \text{if} \quad d \geq d_{thr}, \end{cases} \tag{7.4}$$

where d_{thr} is a threshold distance, and there is no loss for smaller distances (Ramanathan & Rosales-Hain 2000). In my experiments, $d_{thr} = 0.01$. Using this equation for loss due to propagation, the loss varies as the power ε of distance (i.e., $\frac{1}{d^{\varepsilon}}$). In all of the experiments presented below, $\varepsilon = 2$.

Definition 15 *(Ramanathan & Rosales-Hain 2000) Given a multihop wireless network $M = (N, L)$, a set of transmit powers $P = \{p_i\}$, and a propagation function γ, the* **induced graph** *is represented as $G = (V, E)$, where V is the set of vertices corresponding* *to nodes in* N*, and* E *is a set of undirected edges such that* $(u, v) \in E$ *if and only if* $p(u) \geq \gamma(d(u, v))$*, and* $p(v) \geq \gamma(d(u, v))$.[5]

Finally, the topology control problem in multihop wireless networks is the problem of finding a set of transmit powers $P = p_i$ that optimizes a cost metric over the set of transmit powers and that satisfies some constraint on the induced graph. There are several possibilities for the cost metric over the set of transmit powers, from which I have selected *average transmit power*. Minimizing average transmit power means maximizing energy efficiency and average lifespan of the individual agents in the network. The obvious constraint in topology control is *connectivity*. For a network to be connected, there must exist a (possibly multihop) path from every node to every other node in the network. As will be seen in the experiments presented below, guaranteeing a connected network in distributed topology control is difficult. Therefore, the constraint will be relaxed to a performance measure where the number of agents in the largest component of the network will be used to assess performance. Obviously, in a connected network, all of the agents are in the largest component.

[5]This is a slight modification of the definition given by Ramanthan (2000) with no change in meaning. It is also a slight abuse of notation.

7.2.3 A Centralized Algorithm: CONNECT

Before considering distributed topology, I present a centralized topology algorithm known as CONNECT (Ramanathan & Rosales-Hain 2000). I use this algorithm to aid in understanding the topology control problem and to serve as a baseline in the experiments below.

The CONNECT algorithm essentially computes a minimum spanning tree over all of the nodes in a wireless network, where the cost of a connection is the power needed to establish the connection. The algorithm starts by sorting all pairs of nodes based on distance. Initially, each node is considered its own cluster. Then, iteratively following the sorted order of node pairs, the power of the nodes that are closest is increased to establish a connection if the two nodes under consideration are not in the same cluster. When there is only one cluster, the algorithm stops. Figure 7.7 gives pseudocode for CONNECT.

One complication of the greedy approach of CONNECT is so-called *side-effect* edges (Ramanathan & Rosales-Hain 2000). Side-effect edges occur when a node i increases its power in order to connect with a particular node j and as a consequence establishs a connection with a third node k (which may already be in the same cluster as i). An obvious approach to removing side-effect edges is to post-process the network, and reduce the power of nodes involved in side-effect edges. In CONNECT, this is done with the post-processing method *perNodeMinimalize*, shown in Figure 7.7. This post-processing step guarantees that the topology is *per-node-minimal* and that the CONNECT algorithm finds an optimal solution to the minimum average power topology control problem and guarantees connectivity (Ra-

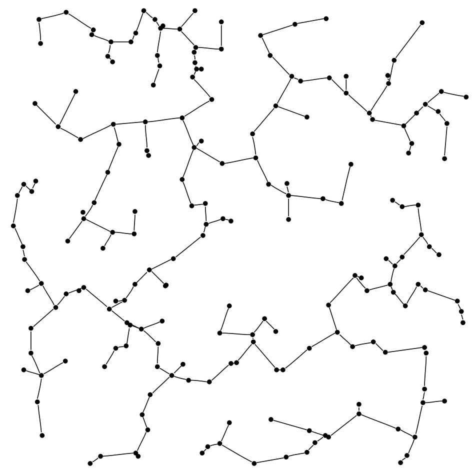


Figure 7.6: A network resulting from applying the centralized topology control algorithm CONNECT. It is clear that the network is a minimum spanning tree.

manathan & Rosales-Hain 2000).

Figure 7.6 shows an example topology resulting from the CONNECT algorithm with $N = 200$ sensors randomly distributed in the unit square. In this example, $\varepsilon = 2$ and $d_{thr} = 0.01$. The figure demonstrates that CONNECT finds a minimum spanning tree over the set of nodes.

While CONNECT is an exemplar of a centralized topology control algorithm, it is unrealistic and does not apply to many real-world applications. Due to its centralized nature, all information about all agents in the network must be located, or aggregated, at a single location, including perfect information about the locations of each of the agents. In ad hoc, mobile, or faulty networks, aggregating information is difficult and is further complicated by the fact that the information is constantly changing. Additionally, CONNECT results

Algorithm: *CONNECT*

```
input:
   M = (N, L): a multihop wireless network
   γ: a propagation function
output:
   P = {p_u}: a set of transmit powers
begin
   sort node pairs in non-decreasing order of distance
   initialize |N| clusters, one per node
   for each (u, v) in sorted order
      if cluster(u) ≠ cluster(v)
         p_u = p_v = γ(d(u, v))
         merge cluster(u) with cluster(v)
         if number of clusters is 1
            end for
   perNodeMinimalize(M, γ, P)
end


procedure: PerNodeMinimize(M,γ,P)
   begin
      for each node u
         for each v where p(u) > γ(d(u, v))
            if graph with p(u) = γ(d(u, v)) is not connected
               end for // end inner loop
            else p(u) = γ(d(u, v))
   end
```

Figure 7.7: Pseudocode for the CONNECT algorithm (Ramanathan & Rosales-Hain 2000). The *PerNodeMinimize* subroutine removes redundant connectivity.

in a fragile topology, since a single node failure will result in the network becoming disconnected. This may also happen when a node moves (i.e., agent mobility). One way of alleviating this effect is to require *biconnectivity* – where all pairs of agents have at least two edge-disjoint paths connecting them (Ramanathan & Rosales-Hain 2000).

7.2.4 A Decentralized Algorithm: LINT

The CONNECT algorithm guarantees an optimal (i.e., maximum energy conservation that guarantees connectivity) topology, but the centralized nature of the algorithm is unrealistic. More realistic topology control algorithms for ad hoc wireless networks are decentralized. In decentralized algorithms, individual nodes make decisions about their transmit power based on locally observable information.

Whereas most distributed topology control algorithms use positional information, LINT (Ramanathan & Rosales-Hain 2000) is a distributed algorithm that does not rely on GPS or some other positioning system.[6] For this reason, I selected LINT as a baseline for comparing AONs for topology control. The basic premise of LINT is that a designer specifies a desired degree, as well as an upper threshold and lower threshold on degree. If a node's degree is higher than the upper threshold, it reduces power. If a node's degree is lower than the lower threshold, it increases its power. The complexity of the LINT algorithm is in how the nodes change their power.

[6]Even CONNECT uses distances to calculate transmission powers.

LINT assumes a "generic model of propagation by which the loss function varies as some ε power of distance" (Ramanathan & Rosales-Hain 2000). Letting d be the distance between two nodes, recall that the loss function is

$$\gamma(d) = 10 \cdot \varepsilon \cdot \log_{10}(\frac{d}{d_{thr}}). \tag{7.5}$$

Now, let k_d be the desired, designer-specified, degree of each node in the network. Likewise, let k_l and k_h be the high and low thresholds for degree respectively. Following the same convention, let k_c be the current degree of a node and let p_c be the current power of a node. LINT provides a formula for computing p_d, the desired power, given p_c, k_c, and k_d. The main assumption in the derivation of the formula for adjusting power is that the D nodes are distributed uniformly at random in the plane. Under this assumption, the degree k of a node is related to its range r, the maximum distance over which it can communicate, by the equation

$$k = D\pi r^2. \tag{7.6}$$

Recall that S is the receiver sensitivity, and for a node to communicate with another node at a current distance (i.e., current range) of r_c

$$p_c - 10 \cdot \varepsilon \log_{10}(\frac{r_c}{r_{thr}}) = S, \tag{7.7}$$

and, likewise for desired degree:

$$p_d - 10 \cdot \varepsilon \log_{10}(\frac{r_d}{r_{thr}}) = S. \tag{7.8}$$

Notice that I am using r_{thr} as equivalent to d_{thr} from above, which is the threshold for which there is no loss for smaller distances.

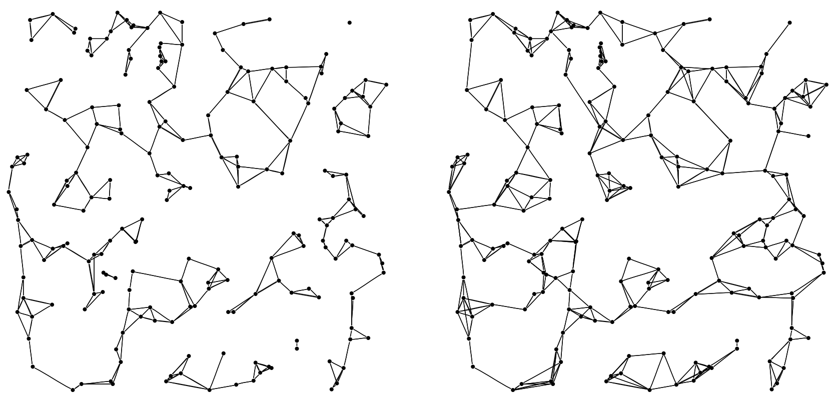
Finally, by equating Equation (7.7) and Equation (7.8) and substituting Equation (7.6), a node can use the following equation to adjust its power:

$$p_d = p_c - 5 \cdot \varepsilon \cdot \log_{10}(\frac{k_d}{k_c}). \tag{7.9}$$

Given this formula for adjusting power, the LINT algorithm is simple. In a wireless network using LINT for topology control, if a node senses that its degree is larger than k_h or smaller than k_l, it adjusts its power using Equation (7.9) for both increasing and decreasing its power (Ramanathan & Rosales-Hain 2000).

Of course, the order and nature of the node updates is important in the LINT algorithm. In the original presentation of the algorithm, updates were synchronous: a node is selected at random during each iteration to update its power. This method for updating is preserved in the experiments conducted below. The LINT topology control algorithm runs continuously until there are no changes in the transmit powers of the nodes in the network.

The benefit of LINT is that it is a complete decentralized algorithm. Each of the nodes in a wireless network using LINT for topology control individually adjusts its power based on locally sensed information about other nodes in the network. A major downfall of LINT is that it does not guarantee connectivity, as can be seen in Figures 7.8 (a) and (b).



(a) (b)

Figure 7.8: Two networks resulting from the distributed topology control algorithm LINT: (a) $k_l = 2, k_d = 3$, and $k_h = 4$; and (2) $k_l = 3, k_d = 4$, and $k_h = 5$. The algorithms does not guarantee connectivity.

7.2.5 An AON for Topology Control

Distributed topology control is an obvious application of learning-based AONs. In this section, I develop an AON for topology control based on the general AON framework proposed in Chapter 4. Unlike the other multi-agent domains considered in this dissertation, agents for topology control in wireless networks do not have the ability to control or adapt individual connections. Rather, agents for topology control only have the ability to increase or decrease their transmission power, which subsequently affects their connectivity with other agents in the network.

Without the ability to control individual connections (and as a result the *when*, *which*, and *where* of connection adaptation), agents using AON strategies for topology control require a more complex approach. In particular, the AON for topology control is a stateful Q-learning approach, where the state of an agent is its current degree (i.e., the number of bidirectional connections with other agents). This is an abstraction of true state, but it provides the agents with additional information and discrimination power for deciding when and how to adapt their transmission power.

The action set is $A = \{increase, decrease, nothing\}$, where the $nothing$ action allows an agent to monitor and make use of the changes that result from other agents. As a result of extending the general AON framework from stateless to stateful, I now use the full Q-learning update function for assign value to state and action pairs:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R(s,a) - Q(s,a) + \beta V(s')], \tag{7.10}$$

where α is the learning rate, $R(s,a)$ is the reward for taking action a in state s, β is the discount factor, s' is the resulting state from taking action a in state s, and

$$V(s) = \max_a Q(s,a). \tag{7.11}$$

Following the general AON framework, a variable learning rate is used: $\alpha = \alpha_{max}$ when $R(s,a) < 0$; otherwise, $\alpha = \alpha_{min}$. Last is the design of the reward function, which is critical to AONs for topology control.

With the goal of establishing maximum connectivity with minimum power consumption, the reward function provides reinforcement for *unique* local connectivity and low power consumption. By unique local connectivity, I mean non-redundant connectivity within local areas of the network. That is, if i and j, j and k, and k and i all have sufficient power levels to communicate with one another, then the local connectivity is not unique. In such a situation, it is preferable for at least one of the agents to reduce its power. To determine the unique set of neighbors, agents that can communicate must share information about their neighbors with one another.

Let P be the maximum power level for each of the agents. Let k_i, u_i, and p_i be the current degree, number of unique neighbors, and transmit power, respectively, for agent i. There are many possibilities for a reward function based on local information; here, I use:

$$R(s,a) = \begin{cases} -1 \quad \text{if} \quad k_i < k_{min}, \quad \text{and} \\ \frac{u_i}{k_i} + \frac{P-p_i}{P} \quad \text{otherwise}. \end{cases} \tag{7.12}$$

In this reward function, k_{min} is a parameter that specifies the minimum degree required. By setting the reward to -1 below the minimum degree, the agent is driven to increase power until its degree increases above the minimum degree threshold. This reward function was chosen because of its simplicity and intuitive nature.

The last component of the AON strategy for topology control is the method the agents employ for determining how much to increase or decrease power. Again, I implemented a simple and intuitive power adjustment method. I choose an adaptive method to allow the agents to adjust more quickly when when large changes in power are necessary and more slowly when small changes in power are necessary. Each agent maintains a step size σ that is multiplied by a constant factor ψ if two increase actions or two decrease actions are taken in consecutive iterations. Likewise, σ is divided by the constant factor $\psi \geq 1$ when an increase action is followed by a decrease action or a decrease action is followed by an increase action. The adjustment to step size is made after the second action is decided upon, but before it is actually taken.

7.2.6 Experimental Results and Discussion

To evaluate the AON described in the previous section for topology control in wireless networks, I conducted a serious of experiments in order to compare the performance of the various topology control algorithms. The experiments compare the AON strategy with LINT and CONNECT.

description	**symbol**	**value**
maximum power	P	25
propagation loss exponent	ε	2
AON learning rate (min)	α_{min}	0.05
AON learning rate (max)	α_{max}	0.4
AON discount factor	β	0.9
AON step size constant	ψ	1.5

Table 7.2: The parameters used to configure the topology control model for the experiments.

Figure 7.9 shows the network topology if there is no topology control and each of the agents maintains a constant transmission power P. The value of the maximum transmission power and all other parameters that do not vary over the experiments are shown in Table 7.2.

In the experiments, I measured the performance of each of the topology control methods

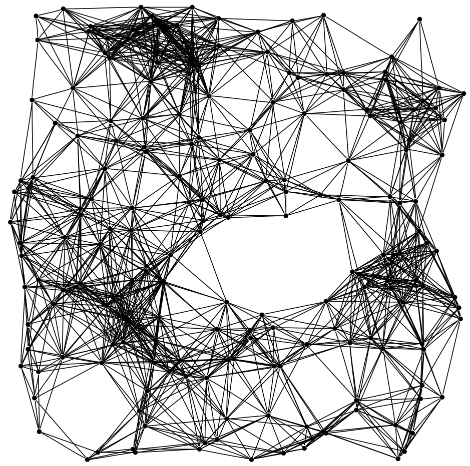
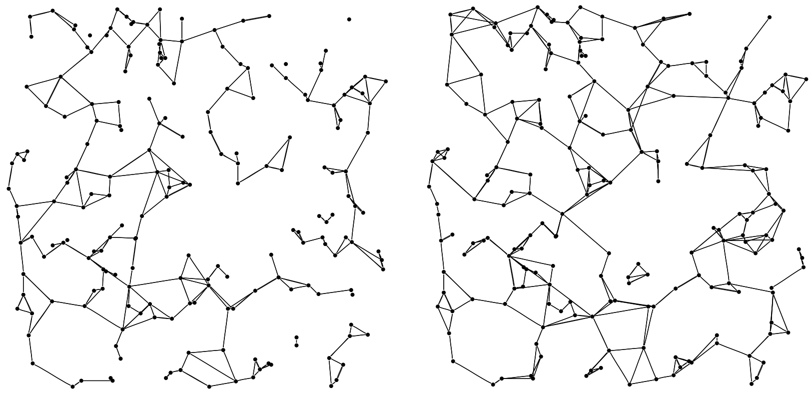


Figure 7.9: A wireless sensor network topology when the sensors are communicating at maximum power given the parameter settings provided in Table 7.2.

for networks with 200, 300, and 400 agents. All of the measurements presented in the results below are averaged over 25 simulations. For LINT, two sets of parameters were used for low, desired, and high degree: $\{2, 3, 4\}$ and $\{3, 4, 5\}$. Two versions of the AON for topology control were used: $k_{min} = 2$ and $k_{min} = 3$.

During the experiments, measurements were taken for *average transmission power*, *number of components*, *size/ratio of the largest component*, and *average degree*. All measurements were taken when the various topology control methods converged (i.e., when there was little or no change in the power distribution over the agents).

Figure 7.11 shows the results of the experiments. The results for average transmission power (Figure 7.11(a)) show that the AON strategies outperform LINT in all cases. Certainly the AON with $k_{min} = 2$ has significantly less power consumption that LINT with both of the two parameter sets tested. To provide an optimal baseline for power consump-

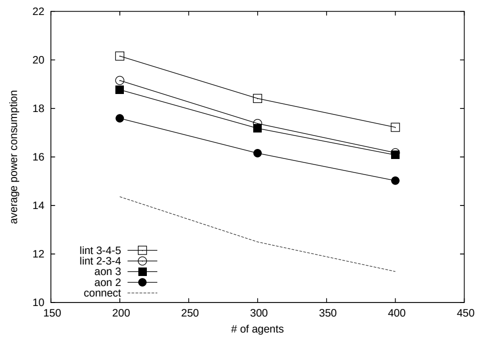


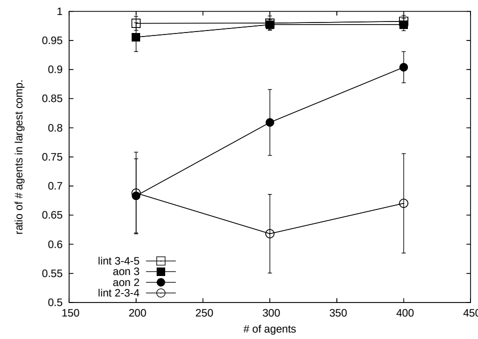
(a) (b)

Figure 7.10: Networks resulting from application of AONs for topology control: (a) $k_{min} = 2$, and (b) $k_{min} = 3$.

tion, the results for CONNECT are also shown in Figure 7.11(a). While average power consumption of the AON is lower than LINT, this result should not be considered independently of the size of the largest component.

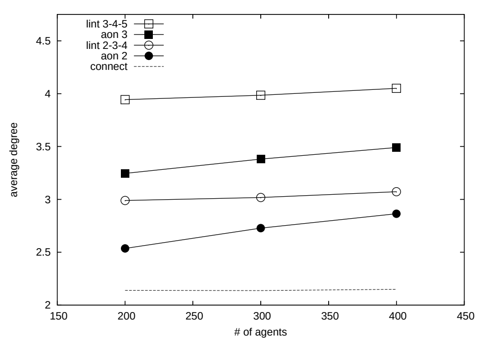
Ideally, the ratio of the number of nodes in the largest component would be 1, but neither LINT nor the AON topology control method guarantee connectivity, largely due to their distributed control with local nature. While this is true, the AON strategies, which resulted in lower average power consumption, result in as large or larger sizes of the largest component. The AON with $k_{min} = 2$ yields significantly larger largest components over LINT with low, desired, and high degree set to 2, 3, and 4 respectively. This is a promising result given that the AON with $k_{min} = 2$ has significantly lower average power consumption. Similarly, the AON with $k_{min} = 3$ sustains a largest component that is nearly the same size, on average, as LINT with the degree settings 3, 4, and 5, although the AON consumes

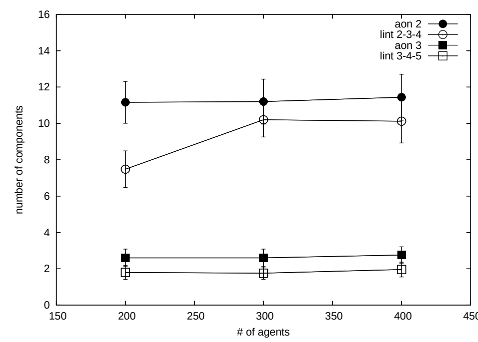




(a)

(b)





(c)

(d)

Figure 7.11: Experimental results of the AONs for topology control for three different size networks (200, 300, and 400): (a) average power consumption, (b) size of the largest component, (c) average degree, and (d) the total number of components.

less power on average. Finally, the number of components results show that there is no significant difference, although LINT tends to have few components on average.

The experimental results demonstrate that there is a gain to be made in using a distributed learning technique such as AONs to organize a network of wireless sensors, or devices. Additionally, I conjecture that the AON techniques are useful when the agents (i.e., sensors) are mobile or faulty. The AON topology control method would allow agents to detect unexpected changes as a result of movement or failure and then to adapt accordingly. Experiments to test this conjecture are beyond the scope of this dissertation and are left for future work.

7.3 Concluding Remarks

In this chapter, I demonstrated the ability of AONs to improve performance in two very different networked multi-agent domains: supply chain management and wireless ad hoc sensor networks. In demonstrating the application of AONs, I highlighted the necessary design decisions required to tailor AONs to a specific environment. When using the general learning-based AON framework, the design decisions include developing a reward function for updating the values of actions, tailoring specific types of changes to connectivity, and customizing a performance measure for evaluating the specialized AONs. In both of the application domains, I demonstrated AONs that provide significant performs increases over static networks or other adaptation mechanisms.

Chapter 8

Summary and Conclusions

> *Learning is any change in a system that produces a more or less permanent change in its capacity for adapting to its environment.*
>
> Herb Simon

8.1 On The Design and Dynamics of Multi-Agent Organizations

In many studies and applications of multi-agent systems, it is assumed that all agents can interact with one another all of the time. While this is a realistic assumption in most small-scale multi-agent systems, this assumption becomes less valid as agent systems grow in size and complexity. The problem is further complicated in open multi-agent systems where the control of the agents does not fall under a single authority.

In large, open multi-agent systems, the agent-to-agent interactions can be limited by cognitive, communication, computational, or locality constraints. When these constraints are present, the agents are limited to interacting with a subset of all of the agents in the system, leading to the emergence of an agent social network. In this dissertation, I have demonstrated that endowing agents with the ability to manage and manipulate their connections in an agent social network can lead to substantial increases in collective performance.

The findings presented in this dissertation have important implications for the design of multi-agent systems. First, the findings emphasize that the structure of agent-to-agent interaction networks can have a dramatic effect on the dynamics of multi-agent systems. Therefore, designers of multi-agent systems, or designers of agents to be embedded within open multi-agent systems, should consider the structure of the interactions among the agents in the system and the ability of the agents to modify this structure.

In Chapter 4, I provided evidence that the design of optimal, or near-optimal, network structures for multi-agent organizations is computationally complex. This translates to intractability for many large multi-agent organizations. Although it is difficult to design optimal organizational networks for multi-agent systems, I have shown that agent-organized networks can discover, or continue to adapt, efficient organizational structures.

8.2 Navigating Social Structures in Networked Multi-Agent Systems

One observation from my experimental results presented in Chapters 5, 6, and 7 is that adaptive organizations tend to outperform static organizations. This observation holds true even when the adaptation strategies are completely random. I conjecture that this is true because random, agent-driven, network adaptations are able to spread resources and computational load throughout the system.

While it is true that random network adaptations provided increased performance over static networks in both the team formation and market environments, adding intelligence to the agents' network adaptation strategies systematically increased performance over the organizations of agents using random adaptation policies. In both environments, increasing the sophistication, or intelligence, of the agents' network adaptation strategies was directly correlated with increased organizational performance. These levels of sophistication were proposed as components of a general learning-based agent-organized network framework presented in Chapter 4. The experimental results support the hypothesis that endowing the agents with the ability to learn *when* and *how* to adapt their connectivity leads to substantial gains in organizational performance.

Another common theme throughout this dissertation is the need to tailor agent-organized networks for specific multi-agent environments. The general AON framework provides a methodology for developing and implementing domain-specific AONs, but there are several components of any AON that must be tailored for a specific environment. These include the design of local performance estimation methods and methods for determining the value of specific connections. The ability to perceive local performance provides the agents with a reinforcement signal for determining the value of adaptations. Methods for determining the value of specific connections allow the agents to wisely determine which connections should be removed or maintained.

Finally, by demonstrating that agent-organized networks increased performance in two general multi-agent environments and two specific multi-agent applications, I have provided evidence for the wide applicability of agent-organized networks in networked multi-agent systems. Furthermore, my results suggest that agent-organized networks are useful in broad ranges of systems, as shown by the diversity of the application domains studied in Chapter 7: topology control in wireless sensor networks and supply network formation. These two application domains are very different in their structure and dynamics, and I showed that extensions of the general AON framework from Chapter 4 improved organizational performance in both applications.

8.3 Future Directions and Final Thoughts

In this dissertation, I have shown that agent-organized networks are a general method for organizational learning in networked multi-agent systems. While I have developed a foundation for agent-organized networks, there are many opportunities for the development of AON theory and the application of AONs.

The Economic Theory of Network Formation Network formation is a relatively new area of study in economics. The early work on the economic theory of network formation has assumed rational, selfish, and myopic individual network formation strategies (Jackson & Wolinsky 1996; Jackson 2003). There are several promising future directions related to the economic theory of network formation including: the application of advanced planning techniques to non-myopic, or farsighted, network formation; the study of cooperative (i.e., non-selfish) network formation processes; and more detailed examinations of the non-equilibrium dynamics of network formation games.

Stability of Agent-Organized Networks The concept of stability is important to both the theory of distributed network adaptation and the design of AON strategies. It may be possible to develop theory that predicts which features of agent-organized networks lead to stable network structures. There are also interesting theoretical questions related to the correlation between stable and efficient agent-organized networks. Answers to these theoretical questions could provide useful guidelines for the design of distributed network adaptation strategies.

A Theory of Network Structure The study of agent-organized networks may help in developing a theory of efficient network structures. Such a theory may be able to develop correlations between network structure and general organizational principles or organizational performance. If a theory of network structures were developed, if would have to deal with both global network structures and locally perceived network structures. This type of theory would help both system designers and agent designers. System designers could use the theory to design organizational network structures. The designers of agents for agent-organized networks could use the theory for developing local network adaptation policies.

Faulty Networks and Autonomic Computing A promising application domain of agent-organized networks is to the self-management and self-healing of networked multi-agent systems. These type of properties have recently been associated with autonomic computing (Ganek & Corbi 2003). While not studied in this dissertation, the application of agent-organized networks to such problems is obvious.

Other Applications of Agent-Organized Networks There are many potential application domains for agent-organized networks. These domains include distributed information retrieval, consumer profiling, recommender systems, automated on-line social networking, automated trading systems, intelligent routing in computer networks, and grid computing. In addition, the applications of agent-organized networks to wireless sensor networks and supply chain management can be further refined and developed.

As agent technology continues to develop and expand, the need for well designed distributed network adaptation strategies will become increasingly important. The adoption of the concepts of ubiquitous computing, autonomic computing, and grid computing will also promote the development of large, open multi-agent systems. In such systems, agents that can manage, maintain, and adapt their own social networks will thrive both individually and collectively.

Finally, the interdisciplinary foundations of agent-organized networks makes them broadly applicable. In computer science, agent-organized networks will likely be used to increase the collective performance of socially-intelligent, distributed software systems. Beyond computer science, the theory and concepts of agent-organized networks may be used to understand economic behavior and the behavior of dynamic social networks.

References

Abdallah, S., and Lesser, V. 2004. Organization-based cooperative coalition formation. In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Techonology (IAT)*.

Abramson, G., and Kuperman, M. 2001. Social games in a social network. *Phys. Rev. E* 63(030901).

Akkermans, H. 2001. Emergent supply networks: System dynamics simulation of adaptive supply agents and the semantic web. In *Proceedings of the 34th Hawaii International Conference on System Sciences*, 3006–3016. IEEE.

Albert, R., and Barabási, A. 2002. Statistical mechanics of complex networks. *Review of Modern Physics* 99(3):7314–7316.

Albert, R.; Jeong, H.; and Barabási, A. 1999. The diameter of the world-wide web. *Nature* 401:130–131.

Alonso, E.; d'Inverno, M.; Kudenko, D.; Luck, M.; and Noble, J. 2001. Learning in multi-agents systems. *Knowledge Engineering Review* 16(3):277–284.

Anghel, M.; Toroczkai, Z.; Bassler, K.; and Korniss, G. 2004. Competition-driven network dynamics: Emergence of a scale-free leadership structure and collective efficiency. *Physical Review Letters* 92(5).

Arunachalam, R.; Sadeh, N.; Eriksson, J.; Finne, N.; and Janson, S. 2004. The supply chain management game for the trading agent competition competition. Technical Report CMU-CS-04-107, Carnegie Mellon University School of Computer Science.

Axtell, R. 2000. Effects of interaction topology and activation regime in several multi-agent systems. In *MABS*, 33–48.

Barrat, A., and Weigt, M. 2000. On the properties of small-world network models. *Europ. Phys. J. B* 13(547).

Bellman, R. 1957. *Dynamic Programming*. Dover Publications.

Bowling, M., and Veloso, M. 2002. Multiagent learning using a variable learning rate. *Artificial Intelligence* 136(2):215–250.

Brooks, C. H., and Durfee, E. H. 2002. Congregating and market fomration. In *Proceedings of the First International Joint Conference on Autonomous Systems and Multiagent Systems*, 96–103. ACM Press.

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

Brown, J. S., and Duguid, P. 1996. Organizational learning and communities of practice. In *Organizational Learning*. Sage Publications.

Bulka, B.; Gaston, M.; and desJardins, M. 2005. Local learning to improve organizational performance in networked multi-agent team formation. In *Proceedings of the AAAI 05 Workshop on Multi-Agent Learning*.

Caillou, P.; Aknine, S.; and Pinson, S. 2002. A multi-agent method for forming and dynamic restructuring of Pareto optimal coalitions. In *Proceedings of the 1st International Conference on Autonomous Agents and Multiagent systems*.

Carley, K., and Gasser, L. 1999. Computational organization theory. In Weiss, G., ed., *Introduction to Multiagent Systems*. Morgan Kaufmann. Chapter 7.

Carley, K. 1998. Organizational adaptation. *Annals of Operations Research* 85:25–47.

Carley, K. 2002. Computational organization science: A new frontier. *Proceedings of the National Academy of Science* 99(3):7314–7316.

Cerpa, A., and Estrin, D. 2004. Ascent: Adaptive self-configuring sensor networks topologies. *IEEE Transactions on Mobile Computing* 3(3):272–285.

Claus, C., and Boutilier, C. 1998. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 746–752.

Cohen, P.; Levesque, H.; and Smith, I. 1997. On team formation. In Hintikka, J., and Tuomela, R., eds., *Contemporary Action Theory*. Synthese.

Conte, R., and Paolucci, M. 2001. Intelligent social learning. *Journal of Artificial Societies and Social Simulation* 4(1).

Corkill, D., and Lesser, V. 1983. The use of meta-level control for coordination in a distributed problem solving network. In *Proceedings of the Eigth International Joint Conference on Artificial Intelligence*, 748–756.

Cramton, P.; Shoham, Y.; and Steinberg, R., eds. 2006. *Combinatorial Auctions*. MIT Press.

Culler, D.; Estrin, D.; and Srivastava, M. 2004. Overview of sensor networks. *IEEE Computer* 37(8):41–19.

Dall, J., and Christensen, M. 2002. Random geometric graphs. *Phys. Rev. E.* 66(016121).

Dash, R. K.; Jennings, N. R.; and Parkes, D. C. 2003. Computational-mechanism design: A call to arms. *IEEE Intelligent Systems* 40–47.

DeCanio, S.; Watkins, W.; Mitchell, G.; Ami-Atefi, K.; and Dibble, C. 2000. *Advances in Economics of Environmental Resources*, volume 3. JAI Press. chapter Complexity in Organizations.

Delgado, J. 2002. Emergence of social conventions in complex networks. *Artificial Intelligence* 141:171–185.

Dias, M. B.; Browning, B.; and Veloso, M. 2005. Dynamics heterogeneous robot teams engaged in adversarial tasks. Technical Report CMU-RI-05-14, Carnegie Mellon University.

Dignum, V.; Dignum, F.; and Sonenberg, L. 2004. Toward dynamic reorganization of agent societies. In *Proceedings of the Workshop on Coordination in Emergent Agent Societies at ECAI 2004*.

Durfee, E. H., and Lesser, V. R. 1991. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics* 21(5):1167–1183.

Durfee, E. H. 1993. Organizations, plans, and schedules: An interdisciplinary perspective on coordinating ai systems. *Journal of Intelligent Systems, Special Issue on the Social Context of Intelligent Systems* 3(2-4):157–187.

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

Emerson, D., and Piramuthu, S. 2004. Agent-based framework for dynamic supply chain configuration. In *Proceedings of the 37th Hawaii International Conference on Systems Sciences*. IEEE.

Emery-Montemerlo, R.; Gordon, G.; Schneider, J.; and Thrun, S. 2004. Approximate solutions for partially observable stochastic games with common payoffs. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 136–143.

Epstein, J., and Axtell, R. 1996. *Growing Artificial Societies: Social Science from the Bottom Up*. The Brookings Institute Press and MIT Press.

Erdos, P., and Renyi, A. 1959. On random graphs. *Publicationes Mathematicae* (6):290–297.

Erramilli, V.; Matta, I.; and Bestavros, A. 2004. On the interaction between data aggregation and topology control in wireless sensor networks. In *Proceedings of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*.

Excelente-Toledo, C. B., and Jennings, N. 2003. Learning when and how to coordinate. *International Journal of Web Intelligence and Agent Systems* 1(3-4):203–218.

Excelente-Toledo, C. B. 2004. The dynamic selection of coordination mechanisms. *Autonomous Agents and Multi-Agent Systems* 9(1-2):55–85.

Feigenbaum, J., and Shenker, S. 2002. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the Sixth International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1–13. ACM Press.

Foster, I.; Jennings, N. R.; and Kesselman, C. 2004. Brain meets brawn: Why grid and agents need each other. In *Proceedings of the Third International Conference on Autonomous Agents and Multi-Agent Systems*, 8–15.

Foster, I.; Kesselman, C.; and Tuecke, S. 2001. The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications* 15(3):200–222.

Fox, M.; Barbuceanu, M.; and Teigen, R. 2000. Agent-oriented supply-chain management. *International Journal of Flexible Manufacturing Systems* 12(2/3):165–188.

Freidman, E., and Parkes, D. C. 2003. Pricing wifi at starbucks – issues in online mechanism design. In *Fourth ACM Conf. on Electronic Commerce (EC'03)*, 240–241.

Gan, L.; Liu, J.; and Jin, X. 2004. Agent-based, energy efficient routing in sensor networks. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems*.

Ganek, A. G., and Corbi, T. A. 2003. The dawning of the autonomic computing era. *IBM Systems Journal* 42(1).

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

Gasser, L., and Ishida, T. 1991. A dynamic organizational architecture for adaptive problem solving. In *Proceedings of the National Conference on Artificial Intelligence*, 185–190.

Gaston, M., and desJardins, M. 2003. Team formation in complex networks. In *Proceedings of the 1st NAACSOS Conference*.

Gaston, M., and desJardins, M. 2005. Agent-organized networks for dynamic team formation. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*.

Gaston, M.; Simmons, J.; and desJardins, M. 2004. Adapting network structures for efficient team formation. In *Proceedings of AAMAS-04 Workshop on Learning and Evolution in Agent-based Systems*.

Ghosh, S.; Lillis, K.; Pandit, S.; and Pemmaraju, S. 2004. Robust topology control protocols. In *Proceedings of the 8th International Conference on the Principles of Distributed Systems*, 94–109.

Goyal, S. 2003. Learning in networks: A survey. In Demange, G., and Wooders, M., eds., *Group Formation in Economics: Networks, Clubs, and Coalitions*. Cambridge University Press.

Grosz, B. 1996. Collaborating systems. *AI Magazine* 17(2).

Hales, D., and Edmonds, B. 2005. Applying a socially-inspired technique (tags) to improve cooperation in p2p networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part A* 35(3):385–395.

Hansen, E.; Bernstein, D.; and Zilberstein, S. 2004. Dynamic programming for partially observable stochastic games. In *Proceedings of the 19th National Conference on Artificial Intelligence*.

Hardin, G. 1968. The tragedy of the commons. *Science* 162:1243–1248.

He, M.; Jennings, N.; and Leung, H.-F. 2003. On agent-mediated electronic commerce. *IEEE Transactions on Knowledge and Data Engineering* 15(4):985–1003.

Hendler, J. 2001. Agents and the semantic web. *IEEE Intelligent Systems* 16(2):30–37.

Holland, J. 1975. *Adaptation in Natural and Artificial Systems*. The MIT Press.

Holme, P.; Trusina, A.; Kim, B. J.; and Minnhagen, P. 2003. Prisoners' dilemma in real-world acquaintance networks: Spikes and quasi-equilibbria induced by the interplay between structure and dynamics. *Phys. Rev. E* 68(030901).

Horling, B.; Benyo, B.; and Lesser, V. 2001. Using self diagnosis to adapt organizational structure. In *5th International Conference on Autonomous Agents*.

Horling, B.; Mailler, R.; Shen, J.; Vincent, R.; and Lesser, V. 2003. Using autonomy, organizational design and negotion in a distributed sensor network. In Lesser, V.; Ortiz, C.; and Tambe, M., eds., *Distributed Sensor Networks*. Kluwer Academic Publishers. 139–183.

Horling, B.; Mailler, R.; and Lesser, V. 2004. A case study of organizational effects in a distributed sensor network. In *Proceedings of the International Conference on Intelligent Agent Technology*, 51–57.

Hu, J., and Wellman, P. 1998. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the Fifth International Conference on Machine Learning*, 242–250.

Hu, J., and Wellman, P. 2002. Multiagent Q-learning. *Journal of Machine Learning*.

Huberman, B. A., and Hogg, T. 1995. Communities of practice: Performance and evolution. *Computational and Mathematical Organizational Theory* 1(1):73–92.

Huberman, B. A. 2001. The dynamics of organizational learning. *Computational & Mathematical Organization Theory* 7(2):145–153.

Hunsberger, L., and Grosz, B. 2000. A combinatorial auction for collaborative planning. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS-2000)*.

Ishida, T.; Gasser, L.; and Yokoo, M. 1992. Organization self-design of distributed production systems. *IEEE Transactions on Knowledge and Data Engineering* 4(2):123–134.

Jackson, M. O., and Watts, A. 2002. The evolution of social and economic networks. *Journal of Economic Theory* 106:265–295.

Jackson, M. O., and Wolinsky, A. 1996. A strategic model of social and economic networks. *Journal of Economic Theory* 71:44–74.

Jackson, M. O. 2003. A survey of models of network formation: Stability and efficiency. In Demange, G., and Wooders, M., eds., *Group Formation in Economics: Networks, Clubs, and Coalitions*. Cambridge University Press.

Jensen, D., and Lesser, V. 2002. Social pathologies of adaptive agents. In Barley, M., and

Guesgen, H., eds., *Safe Learning Agents: Papers from the 2002 AAAI Spring Symposium*. AAAI Press. 13–19.

Jones, C. E.; Sivalingam, K. M.; Agrawal, P.; and Chen, J. C. 2001. A survey of energy efficient network protocols for wireless networks. *Wireless Networks* 7(4):343–358.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4:237–285.

Kakade, S.; Kearns, M.; Ortiz, L.; Pemantle, R.; and Suri, S. 2004. The economic properties of social networks. In *Proceedings of Neural Information Processing Systems (NIPS)*.

Kakade, S.; Kearns, M.; and Ortiz, L. 2004. Graphical economics. In *Proceedings of Computational Learning Theory (COLT)*.

Kearns, M.; Littman, M.; and Singh, S. 2001. Graphical models for game theory. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 253–260.

Kim, B.; Trusina, A.; Holme, P.; Minnhagen, P.; Chung, J.; and Choi, M. 2002. Dynamics instabilities induced by asymmetric influence: Prisoners' dilemma game in small-world networks. *Phys. Rev. E* 66(021907).

Kimbrough, S.; Wu, D. J.; and Zhong, F. 2001. Computers play the beer game: Can artificial agents manage supply chains? In *Proceedings of the 34th Hawaii International Conference on System Sciences*, 7016–7025. IEEE.

Koza, J. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press.

Kraus, S.; Shehory, O.; and Taase, G. 2003. Coalition formation with uncertain heterogeneous information. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '03)*.

Lee, H.; Padmanabhan, V.; and Whang, S. 1997. Information distortion in a supply chain: The bullwhip effects of interaction topology and activation regime in several multi-agent systems. *Management Science* 43(4):546–558.

Lerman, K., and Shehory, O. 2000. Coalition formation for large-scale electronic markets. In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, 167–174.

Levinthal, D. A., and March, J. G. 1993. The myopia of learning. *Strategic Management Journal* 14:95–112.

Li, N., and Hou, J. 2004. Topology control in heterogeneous wireless networks: Problems and solutions. In *INFOCOM*.

Li, L.; Halpern, J.; Bahl, P.; Wang, Y.-M.; and Wattenhofer, R. 2001. Analysis of a cone-based distributed topology control algorithm for wireless multi-hop networks. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, 264–273.

Liao, E.; Scerri, P.; and Sycara, K. 2004. A framework for very large teams. In *Proceedings of the Autonomous Agents and Multiagent Systems (AAMAS '04) Workshop on Coalitions and Teams*.

Littman, M. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, 157–163.

Littman, M. 2001a. Friend-or-foe Q-learning for general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learning*, 322–328. Morgan Kaufmann.

Littman, M. 2001b. Value-function reinforcement learning in markov games. *Cognitive Systems Research* 2(1):55–66.

Liu, H., and Iba, H. 2003. Multi-agent learning by evolutionary subsumption. In *Proceedings of the Congress on Evolutionary Computation*, 1115–1122.

Milgram, S. 1967. The small world problem. *Psychology Today* 22:61–67.

Miller, J. 2001. Evolving information processing organizations. In Lomi, A., and Larsen, E., eds., *Dynamics of Organizations*. MIT Press. chapter 10.

Mitchell, T. 1997. *Machine Learning*. McGraw Hill.

Modi, P. J.; Jung, H.; Tambe, M.; Shen, W.-M.; and Kulkarni, S. 2001. A dynamic distributed constraint satisfaction approach to resource allocation. In *Proceedings of the Seventh International Conference on Principles and Practice of Constraint Programming*.

Nair, R.; Tambe, M.; and Marsella, S. 2002. Team formation for reformation. In *Proceedings of the AAAI Spring Symposium on Intelligent Distributed and Embedded Systems*.

Nair, R.; Tambe, M.; and Marsella, S. 2003. Role allocation and reallocation in multiagent teams: Towards a practical analysis. In *Proceedings of the Second International Joint Conference on Agents and Multiagent Systems (AAMAS)*.

Newman, M. E. J. 2002. Assortative mixing in networks. *Phys. Rev. Lett.* 89(208701).

Newman, M. 2003. The structure and function of complex networks. *SIAM Review* (45):167–256.

Norman, T.; Preece, A.; Chalmers, S.; Jennings, N.; Luck, M.; Dang, V.; Nguyen, T.; Deora, V.; Shao, J.; Gray, W. A.; and Fiddian, N. 2004. Agent-based formation of virtual organizations. *Knowledge-Based Systems* 17:103–111.

Parkes, D. C., and Shneidman, J. 2004. Distributed implementations of vickerey-clarke-groves mechanisms. In *Proceedings of the Third International Conference on Autonomous Agents and Multi-agent Systems*, 261–268.

Parkes, D. C. 2004. On learnable mechanism design. In Tumer, K., and Wolpert, D., eds., *Collectives and the Design of Complex Systems*. Springer-Verlag. 107–131.

Peshkin, L.; Kim, K. E.; Meuleau, N.; and Kaelbling, L. P. 2000. Learning to cooperate via policy search. In *Proceedings of the 16th International Conference on Uncertainty in Artificial Intelligence*, 489–496.

Pishro-Nik, H.; Chan, K.; and Fekri, F. 2004. On connectivity properties of large-scale sensor networks. In *Proceedings of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*.

Potter, M., and Jong, K. D. 2000. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation* 8(1):1–29.

Pynadath, D., and Tambe, M. 2002. Multiagent teamwork: Analyzing the optimality and complexity of key theories and models. In *Proceedings of the Internation Conference on Autonomous Agents and Multiagent Systems (AAMAS '02)*.

Rajaraman, R. 2002. Topology control and routing in ad hoc networks: A survey. *SIGACT News* 33:60–73.

Ramanathan, R., and Rosales-Hain, R. 2000. Topology control of multihop wireless networks using transmit power adjustment. In *INFOCOM 2*, 404–413.

Ramanathan, M. K.; Kalogeraki, V.; and Pruyne, J. 2002. Finging good peers in peer-to-peer networks. In *International Parallel and Distributed Computing Symposium (IPDPS)*.

Rappaport, T. S. 1996. *Wirelss Communications: Principles and Practice*. Prentice Hall.

Rodoplu, V., and Meng, T. 1999. Minimum energy mobile wirelesss networks. *IEEE Journal on Selected Areas in Communications* 17(8):1333–1344.

Saha, S.; Sen, S.; and Dutta, P. S. 2003. Helping based on future expectations. In

Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems.

Sandholm, T., and Lesser, V. R. 1997. Coalitions among computationally bounded agents. *Artificial Intelligence* 94(1-2):99–137.

Sandholm, T.; Larson, K.; Andersson, M.; Shehory, O.; and Tohme, F. 1998. Anytime coalition structure generation with worst case guarantees. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 46–53.

Sandholm, T. 1999. Algorithm for optimal winner determination in combinatorial auctions. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99)*, 542–547.

Scerri, P.; Hu, Y.; Liao, E.; Lai, J.; and Sycarra, K. 2004. Scaling teamwork to very large teams. In *Proceedings of the International Conference on Autonomous Agents and Mutliagent Systems (AAMAS '04)*.

Schieritz, N., and Grobler, A. 2003. Emergent structures in supply chains – a study integrating agent-based and system dynamics modeling. In *Proceedings of the 36th Hawaii International Conference on System Sciences*, 94–102. IEEE.

Sen, S., and Dutta, P. S. 2000. Searching for optimal coalition structures. In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, 287.

Sen, S., and Weiss, G. 1999. Multiagent learning. In Weiss, G., ed., *Multiagent Systems*. MIT Press.

Sen, S.; Sekaran, M.; and Hale, J. 1994. Learning to coordinate without sharing information. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 426–431.

Silvey, P., and Hurwitz, L. 2004. Adapting peer-to-peer topologies to improve system performance. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS '04)*, volume 7, 70199a. IEEE.

Sims, M.; Goldman, C. V.; and Lesser, V. R. 2003. Self-organizing through bottom-up coalition formation. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*. ACM Press.

Skyrms, B., and Pemantle, R. 2000. A dynamic model of social network formation. *Proceedings of the National Academy of Sciences USA* 97:9340–9346.

So, Y., and Durfee, E. 1993. An organizational self-design model for organizational change. In *Working Notes of the AAAI-93 Workshop on AI and Theories of Groups and Organizations*.

So, Y., and Durfee, E. H. 1996. Designing tree-structured organizations for computational agents agents. *Computational and Mathematical Organization Theory* 2(3):219–246.

Song, W.-Z.; Wang, Y.; Li, Z.-Y.; and Frieder, O. 2004. Localized algorithms for energy efficient topology in wireless ad hoc networks. In *Proceedings of the 5th ACM International Symposium on Mobile Ad Hc Networking and Computing*, 98–108.

Strogatz, S. 2002. Exploring complex networks. *Nature* (410):268–276.

Sun, R. 2001. Individual action and collective function: From sociology to multi-agent learning. *Journal of Cognitive Systems Research* 2:1–3.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning*. MIT Press.

Szolnoki, A., and Szabo, G. 2004. Phase transitions for rock-scissors-paper game on different networks. *Physical Reveiw E* 70(037120).

Tambe, M. 1997. Towards flexible teamwork. *Journal of Artificial Intelligence Research* 7:83–124.

Tesfatsion, L. 1997. A trade network game with endogenous partner selection. In Amman, H. M.; Rustem, B.; and Whinston, A. B., eds., *Computational Approaches to Economic Problems*. Kluwer Academic Publishers.

Thadakamalla, H. P.; Raghaven, U. N.; Kumera, S.; and Albert, R. 2004. Survivability of multiagent-based supply networks: A topological perspective. *IEEE Intelligent Systems* 19(5):24–31.

Tran, T., and Cohen, R. 2003. Learning algorithms for software agents in uncertain and untrusted market environments. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*.

Tumer, K., and Wolpert, D. 2000. Collective intelligence and braess' paradox. In *Proceedings of teh Sixteenth National Conference on Artificial Intelligence*, 104–109.

Walsh, W. E., and Wellman, M. P. 1998. A market protocol for decentralized task allocation. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS-98)*.

Walsh, W. E., and Wellman, M. 2000. Modeling supply chain formation in multiagent systems. In *Lecture Notes in Artificial Intelligence: Agent-Mediated Electronic Commerce II*, volume 1788. Springer-Verlage.

Walsh, W. E., and Wellman, M. P. 2003. Decentralized supply chain formation: A market protocol and competitive equilibrium analysis. *Journal of Artificial Intelligence Research* 19:513–567.

Watkins, C. J. C. H., and Dayan, P. 1992. Technical note: Q-learning. *Machine Learning* 8(3/4):279–292.

Wattenhofer, R., and Zollinger, A. 2004. Xtc: A practical topology control algorithm for ad-hoc networks. In *Proceedings of the 4th International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks*.

Wattenhofer, R.; Li, L.; Bahl, P.; and Wang, Y.-M. 2001. Distributed topology control for wireless multihop ad-hoc networks. In *INFOCOM*.

Watts, D., and Strogatz, S. 1998. Collective dynamics of ‘small-world’ networks. *Nature* 393:440–442.

Watts, D. 1999. *Small Worlds*. Princeton University Press.

Watts, A. 2001. A dynamic model of network formation. *Games and Economic Behavior* 34(331-341).

Wellman, M.; Cheng, S.-F.; Reeves, D.; and Lochner, K. 2003a. Trading agent competition: Performance, progress, and market effectiveness. *IEEE Intelligent Systems* 18(6):48–53.

Wellman, M.; Greenwald, A.; Stone, P.; and Wurman, P. 2003b. The 2001 trading agent competition. *Electronic Markets* 12(1).

Wellman, M. 2004. Online marketplaces. In Singh, M., ed., *Practical Handbook of Internet Computing*. CRC Press.

Wilhite, A. 2001. Bilateral trade and ’small-world’ networks. *Computational Economics* 18(1):49–64.

Wilhite, A. 2003. Self-organizing production and exchange. *Computational Economics* 21(1-2):107–123.

Wolpert, D., and Tumer, K. 1999. An introduction to collective intelligence. In Bradshaw, J. M., ed., *Handbook of Agent Technology*. AAAI Press/MIT Press.

Wooldridge, M., and Jennings, W. 1999. The cooperative problem solving process. *Journal of Logic and Computation* 9(4):563–592.

Xu, Y.; Heidemann, J.; and Estrin, D. 2001. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the ACM Intrenational Conference on Mobile Computing and Networking*, 74–84.

Yolum, P., and Singh, M. P. 2003. Emergent personalized communities in referral networks. In *Proceedings of the IJCAI Workshop on Intelligent Techniques for Web Personalization (ITWP)*.

Yong, C., and Miikkulainen, R. 2001. Cooperative coevolution of mult-agent systems. Technical Report AI01-287, University of Texax at Austin.

Yu, B., and Singh, M. P. 2003. Searching social networks. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. ACM Press.

Yu, B.; Venkatraman, M.; and Singh, M. P. 2003. An adaptive social network for information access: Architecture and experimental results. *Applied Artificial Intelligence* 17(1):21–38.

Zeng, D., and Sycara, K. 1999. Dynamic supply chain structuring for electronic commerce among agents. In Klusch, M., ed., *Intelligent Information Agents: Cooperative, Rational and Adaptive Information Gathering on the Internet*. Springer Verlag.

Zhang, H.; Croft, W. B.; Levine, B.; and Lesser, V. 2004a. A multi-agent approach for peer-to-peer-based information retrieval systems. In *Proceedings of Autonomous Agents and Multie-agent Systems (AAMAS04)*.

Zhang, S. C.; Koprulu, F. I.; Koetter, R.; and Jones, D. 2004b. Feasibility analysis of stochastic sensor networks. In *Proceedings of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*.

Zimmermann, M. G.; Equiluz, V. M.; and Miguel, M. S. 2001. Cooperation, adaptation and the emergence of leadership. In Kirman, A., and Zimmermann, J.-B., eds., *Economics with Heterogeneous Interacting Agents*, volume 503. Springer. 73–86.

Zimmermann, M.; Equiluz, V.; and Miguel, M. S. 2004. Coevolution of dynamical states and interactions in dynamic networks. *Physical Review E.* 69(6):065102(4).