Automated Negotiations Among Autonomous Agents In Negotiation Networks

by

Hrishikesh J. Goradia

Bachelor of Engineering in Computer Engineering
University of Mumbai, 1997

Master of Science in Computer Science
University of South Carolina, 2003

Submitted in Partial Fulfillment of the

Requirements for the Degree of Doctor of Philosophy in the

Department of Computer Science and Engineering

College of Engineering and Computing

University of South Carolina

2007

Major Professor	Chairman, Examining Committee
Committee Member	Committee Member
Committee Member	Dean of The Graduate School

This dissertation is dedicated to my parents, Jyotsna and Jawahar Goradia, and my wife Deepa.

Acknowledgments

I would like to acknowledge the guidance and support of my advisor, José Vidal, as well as the help I received from the other members of my dissertation committee: Michael Huhns, Marco Valtorta, Manton Matthews, and Anand Nair. I am indebted to them for their intellectual and financial support to me at various stages of my graduate career.

Abstract

Distributed software systems are a norm in today's computing environment. These systems typically comprise of many autonomous components that interact with each other and negotiate to accomplish joint tasks. Today, we can integrate potentially disparate components such that they act coherently by coordinating their actions via message exchanges. Once this integration issue is resolved, the next big challenge in computing is the automation of the negotiation process between the various system components. In this dissertation, we address this automated negotiation problem in environments where there is a conflict of interest among the system components. We present our negotiation model - a *negotiation network* - where a software system is a network of agents representing individual components in the system. We analyze the software system as a characteristic form game, one of many concepts in this dissertation borrowed from game theory. The agents in our model preserve the self-interest of the components they represent (their owners), and make decisions that maximize the expected utilities of their owners. These agents accomplish joint tasks by forming coalitions. We show that the problem of computing the optimal solution, where the utilities of all agents are maximized, is hyper-exponential in complexity. We present an *approximate algorithm* for this hard problem, and evaluate it empirically. The simulation results show that our algorithm has many desirable properties - it is distributed, efficient, stable, scalable, and simple. Our algorithm produces the optimal (social welfare maximizing) solution for 96% of cases, generates maximal global revenue for 97% of cases, converges to 90% of the best found allocation after

only 10 rounds of negotiation, and finds a core-stable solution for revenue distribution among the agents for cases with nonempty core. Finally, to ensure stability for all cases, we present a sliding-window algorithm that computes the *nucleolus-stable* solution under all situations.

Contents

Dedication ii

Acknowledgments iii

Abstract iv

List of Tables xi

List of Figures xiii

I Overview 1

1 Introduction 2

1.1 Motivating Examples 4

1.1.1 Workflow and Business Process Management 4

1.1.2 Agent-Mediated Electronic Commerce 5

1.1.3 Multirobot Coordination 6

1.2 Problem Statement 6

1.2.1 Automated Negotiation Problem 7

1.2.2 Desiderata for the Automated Negotitaion Mechanism 7

1.3 Dissertation Overview 9

II Background and Related Work **11**

2 Negotiation Research in Game Theory and Economics **13**

2.1 Bargaining Problem 14

2.1.1 Nash Bargaining Solution 15

2.1.2 Pareto-optimal Solution 17

2.1.3 Utilitarian Solution 18

2.1.4 Egalitarian Solution 18

2.1.5 Kalai-Smorodinsky Solution 19

2.2 Bargaining Game of Alternating Offers 20

2.2.1 Nash Equilibrium 22

2.2.2 Subgame Perfect Equilibrium 24

2.2.3 Extensions to the Bargaining Game 27

2.3 Coalitional Games and Cooperative Game Theory 27

2.3.1 Characteristic Form Games 28

2.3.2 The Core 30

2.3.3 The Stable Sets 34

2.3.4 The Bargaining Set 35

2.3.5 The Kernel 37

2.3.6 The Nucleolus 38

2.3.7 Equal Share Analysis 40

2.3.8 Equal Excess Theory 41

2.3.9 The Shapley Value 43

2.4 Issues with Game-theoretic work 44

3 Negotiation Research in Social Sciences **46**

3.1 Network Exchange Theory 46

4 Negotiation Research in Artificial Intelligence and Multiagent Systems 49

4.1 Mechanism for Automated Negotiations 49

4.1.1 Monotonic Concession Protocol 51

4.1.2 Zeuthen Strategy 53

4.1.3 One-Step Protocol 54

4.2 Automated Negotiations in Complex Settings 54

4.2.1 Auctions 55

4.2.2 Coalition Formation 58

4.2.3 Contracting 61

4.2.4 Market-Oriented Programming 64

III Research Contributions of this Dissertation 65

5 Negotiation Networks 66

5.1 Our Agent-Based Negotiation Model 66

5.1.1 Negotiation Network 67

5.1.2 Deal (or Coalition) in a Negotiation Network 68

5.1.3 Expectation (or Utility) of an Agent 68

5.1.4 Deal (or Coalition) Configuration 69

5.2 Modeling Automated Negotiation Problem as a Negotiation Network 70

5.3 Agent Coordination through Coalition Formation 72

6 Approximation Algorithm for the Automated Negotiation Problem 73

6.1 Automated Negotiation Problem Revisited 74

6.2 Theoretical Analysis of the Automated Negotiation Problem 74

6.2.1 Determining the Coalition Values 75

6.2.2 Generating the Optimal Coalition Structure 75

6.2.3 Computing a Stable Payoff Configuration 76
6.3 PACT - Our Negotiation Mechanism to Solve the Automated Negotiation Problem 78
6.3.1 Basic Principles 79
6.3.2 Negotiation Protocol 80
6.3.3 Negotiation Strategy 80
6.3.4 The Algorithm 80
6.3.5 Demonstration of the PACT Algorithm 86
6.4 Experimental Evaluation of PACT 88
6.4.1 Experiments comparing PACT with an Optimal algorithm . . 89
6.4.2 Experiments to test the Scalability of PACT algorithm 92
6.4.3 Convergence in PACT 96
6.4.4 Core Stability Test for PACT Solutions 98
6.5 Generalized Automated Negotiation Problem 99
6.6 PACT Negotiation Mechanism for this Problem 101
6.6.1 Determining Potential Coalitions 101
6.6.2 Determining the Coalition Configuration 102
6.7 Experimental Analysis 103
6.7.1 Experiments comparing our Bargaining Algorithm with a Utilitarian Solution 103
6.7.2 Experiments to test the Scalability of our Bargaining Algorithm 106
6.7.3 Convergence in our Bargaining Algorithm 107
6.7.4 Bargaining Set Stability Test 109
6.8 Summary 111

7 Nucleolus-stable Solution for the Automated Negotiation Problem 113

7.1 Basic Definitions and Related Work 115
7.2 Our Distributed Algorithm for Computing the Nucleolus 116

7.3 Test Results 125
7.4 Summary 128

IV Conclusions **130**

8 Conclusions and Future Work **131**

8.1 Conclusions 131
8.2 Ideas for Future Research 132
8.2.1 Automated Multilateral Multiple-Issue Negotiations 133
8.2.2 Electronic Business and Electronic Commerce 134
8.2.3 Web Services, Service-Oriented Computing, and Business Process Management 135
8.2.4 Semantic Web Services and the Semantic Web 136

Bibliography **137**

List of Tables

6.1 Growth rate of the number of coalition structures with increasing number of agents 76

6.2 Growth rate of the imputation set. We set $v(N) = 8$. The columns show the total number of imputations where the payoff of an agent, $i \in N$, is $x_i = 8, 7, \ldots, 0$ with different agent set sizes $n = 2, \ldots, 10$. The final column represents the total size of the imputation set for each row. 77

7.1 Performance against agent set sizes 126

7.2 Performance with respect to $v(N)$ 127

7.3 Performance for various agent set size / $v(N)$ combinations 128

7.4 Performance against precision levels 128

List of Figures

5.1 Negotiation Network 68

6.1 PACT Negotiation Protocol 80

6.2 Step 1 of the PACT Negotiation Protocol – Iterative, Distributed Negotiations over Payoff Divisions 81

6.3 Step 2 of the PACT Negotiation Protocol – Near-Optimal Coalition Structure Selection 81

6.4 PACT Negotiation Strategy 82

6.5 The FIND-COALITION-PACT algorithm to find the best task allocation for agent i. 84

6.6 Demonstration of the PACT Algorithm 87

6.7 Comparison of the coalition structure size for PACT and optimal algorithms 90

6.8 Comparison of the coalition structure value for PACT and optimal algorithms 91

6.9 Comparison of the computational cost of PACT and optimal algorithms 92

6.10 PACT Scalability with respect to Number of Agents 93

6.11 PACT Scalability with respect to Number of Tasks 95

6.12 Scalability with respect to the Coalition Size 96

6.13 Convergence results in PACT 97

6.14 PACT solutions that are Core Stable 98

6.15 Algorithm to find Potential Coalitions 102

6.16 Comparison of the coalition structure size 104
6.17 Comparison of the coalition structure value 105
6.18 Scalability of our Bargaining Algorithm 107
6.19 Convergence results for our Bargaining Algorithm 108
6.20 Stability results for our Bargaining Algorithm 110

7.1 Geometric representation of imputation set and nucleolus solution for a 3-agent CFG. (a) shows the imputation set, and (b) shows the nucleolus solution in the set. 117
7.2 Our distributed algorithm for computing the nucleolus 118
7.3 Graphical description of our algorithm for computing the nucleolus. (a) shows the coarse-level search stage, and (b) shows the fine-level search stage of the algorithm. 119

Part I

Overview

Chapter 1

Introduction

Distributed information systems are a norm in today's computing environment. All but the most trivial of systems contain a number of subsystems that must interact with one another in order to successfully carry out their tasks. These subsystems can either be a part of the same organization, cooperatively solving the complex problems daily faced by the organization, or be owned by separate, autonomous organizations, which collaborate competitively only to further their own interests. Nonetheless, building information systems that can collaborate with each other and make joint decisions over issues of mutual interest remains a hard problem in computer science.

We are witnessing a fundamental shift in the way enterprises conduct their businesses today. The current trend in information systems is toward increased distribution, decoupling, local intelligence, and collaboration. Traditional integrated enterprises with centralized control are giving way to loosely-coupled networks of applications (or services) owned and managed by diverse business partners that interact via standard protocols. The Web services technology adopts a data model based on XML, and uses standard Internet protocols for interacting with other services. This standards-based approach helps reduce development and maintenance costs for integrated systems, and prepares the enterprises to address the heterogeneous, autonomous, and dynamic nature of today's business environment. Eventually, Web

services could become the basis for a seamless and almost completely automated infrastructure for enterprise integration. Now, once the integration issues are resolved for an information system, we believe that the research focus will shift on higher-level goals such as automation of the decision processes of subsystems. This can lead to higher efficiency and robustness towards achieving the enterprise's business goals. In an attempt to collaborate, the subsystems will negotiate with each other to reach agreements that are mutually favorable. This dissertation focuses on automating this negotiation step in enterprise information systems.

In this dissertation, we model information systems as multiagent systems. Modeling information systems as a society of autonomous agents has its merits. They naturally represent the decentralized nature, the multiple loci of control, the multiple perspectives and/or the competing interests that prevail in most real-world problems (Jennings, Faratin, Lomuscio, Parsons, Wooldridge, & Sierra, 2001). They are also capable of interacting with other agents (or humans) in order to satisfy their design objectives (Wooldridge & Jennings, 1995). Now, for many real-world application scenarios, particularly domains where the subsystems represent interests of humans (or groups of humans), we encounter situations where there is a conflict of interest. For example, business partners in a small business can have diverging interests that result in different opinions on how to allocate the resources of their corporation. For such settings, agents (representing the various subsystems) can potentially negotiate and automatically resolve the situation between the involved parties by reaching an agreement that best meets everybody's requirements. The focus of our research is in devising negotiation mechanisms that allow agents to cooperate and coordinate their actions, and automatically arrive at mutually acceptable agreements in situations of conflict.

We present some examples of target applications for this work in the next section. These examples highlight the scope and significance of this research. We follow this

up with an explicit description of the problem statement, along with the deliverables for this work. Finally, we present an overview of the rest of the dissertation.

1.1 Motivating Examples

Three motivating examples from three different fields are presented in order to illustrate various settings where negotiations among agents can be beneficial. The first example involves software agents, the second concerns with a futuristic technology, while the third scenario involves mobile robots. Clearly, these are just a sampling of the plethora of application domains for automated negotiations among autonomous agents.

1.1.1 Workflow and Business Process Management

Workflow management systems, which aim to automate the processes of a business, have been around for decades. With the advent of the service-oriented computing paradigm and the Web services technology, we have means for addressing the heterogeneity that currently exists across enterprises. Today we have standard languages such as WS-BPEL for defining business processes that span across enterprise boundaries. We need business process management systems to handle such cross-enterprise workflows. The current incarnations of such systems, such as IBM's BPEL4J engine, leave a lot to desire. Typically, these systems are centralized and lack the adaptability to cope with unpredictable events. Systems for handling workflows involving multiple businesses must adopt a decentralized, peer-to-peer architecture to avoid privacy and trust issues. There have been many proposals for modeling business process management systems as multiagent systems (Buhler & Vidal, 2005; Vidal, Buhler, & Stahl, 2004; Goradia & Vidal, 2005) as they naturally address many issues that plague the current workflow systems. For example, adopting an agent-based approach naturally

addresses the issue of aggregating information from distributed data sources owned by different parties. The inherent dependencies and conflicts of interest among the participants in the business processes can be resolved through agent interactions. Such systems could also respond more rapidly to changing circumstances in business environments. The ADEPT system (Jennings, Faratin, Johnson, Norman, O'Brien, & Wiegand, 1996) is an example of an agent-based business process management system from the pre-Web services era. Once the integration issues between businesses are resolved, there will be a need for automatically resolving higher level issues such as contractual agreements. Automated negotiation mechanisms can potentially address these concerns. (Goradia, 2006) presents our position paper for future research in this area.

1.1.2 Agent-Mediated Electronic Commerce

As we all know, electronic commerce is rapidly gaining acceptance in every industry today, and it will only become more widespread in the future. It offers opportunities to significantly improve the way that businesses interact with both their customers and their suppliers. Current (first-generation) e-commerce systems such as Amazon.com allow a user to browse an online catalog of products, choose some, and then purchase these selected products using a credit card. However, agents can lead to second-generation e-commerce systems where many aspects of consumer buying behaviors (in B2C systems) and business-to-business transactions (in B2B systems) are automated (Guttman, Moukas, & Maes, 1998; Sierra & Dignum, 2001). Sophisticated automation on both the buyer's and the seller's side can lead to applications that are more dynamic and personalized. Both buyers and sellers gain from these changes - the buyers can expect the agents to search for and retrieve the best deals available, while the sellers can have the agents that automatically customize their offerings to customers based on various parameters, such as the customer type, current seller

competition, and current state of the seller's own business. This advanced degree of automation can be achieved by modeling the e-commerce systems as interacting agents (Jennings, 2001; He, Jennings, & Leung, 2003). Research in this dissertation can potentially lead to the kind of agent negotiation mechanisms necessary for second-generation e-commerce systems.

1.1.3 Multirobot Coordination

Consider multirobot environments where these robots are responsible for accomplishing tasks such as transporting equipments within a manufacturing plant, delivering packages in an office, rescuing victims in situations inaccessible by humans, or tracking enemy targets in battlefields. In all these scenarios, the robots have to coordinate their actions, usually through communication, in order to achieve their goals. They have to make independent decisions based on their perception of the environment, and act in a manner that optimizes the global utility. (Dias, Zlot, Kalra, & Stentz, 2006) presents an recent survey of multirobot coordination research. Again, the automated negotiation mechanisms from this dissertation address these issues.

1.2 Problem Statement

How can a group of agents in a situation where there is a conflict of interest (i.e. scenarios where the agents have to cooperate in order to improve their utilities, but an agent's gain is always at the expense of other agents) automatically come to an agreement that is mutually beneficial? This is the automated negotiation problem that we address in this dissertation. We define this problem formally in the following subsection.

1.2.1 Automated Negotiation Problem

Definition 1 (Automated Negotiation Problem) *Say, there are N utility-maximizing agents in the environment, where $N > 2$. The worth $v(S)$ of each subset $S \subseteq N$ agents is common knowledge. Now, the agents have to collaborate and commit as a group of size i, where $1 \leq i \leq N$, to make any profit (determined by the worth of the group). Also, each agent can commit to at most one group at a time. How should an agent choose its group such that it maximizes its expected utility? What should the revenue of each agent in a committed group be (such that they cannot do any better with some other group, and therefore, will not have any incentive to break their commitment)? Can the collective utility of all agents be maximized too?*

An intuitive approach for addressing the automated multilateral negotiation problem is where the agents go through a coordination process involving negotiations over all possible agreements covering issues of common interest, eventually bringing them all to a consensus. In their pioneering work, Jeffrey Rosenschein and Gilad Zlotkin (Rosenschein & Zlotkin, 1994) proposed that we must define the 'rules of encounter' between agents [1]. Defining these rules entail mechanism design, which involves devising a negotiation protocol and a negotiation strategy (or strategies) for the agents. What is a good negotiation mechanism that automates the agents' decision-making process while preserving the best interests for each of them? We present such a mechanism in this dissertation.

1.2.2 Desiderata for the Automated Negotitaion Mechanism

There are many desirable properties in a negotiation mechanism. *Efficiency* is arguably the most important property for a mechanism. Here we mean efficiency in

[1] This topic is discussed in further details in section 4.1

many forms - economic efficiency, computational efficiency, and communication efficiency. By economic efficiency we mean that the agreement that a mechanism yields should be (close to) optimal. Optimality can also be measured in many ways, and is domain-specific. For example, maximizing the global revenue might be the primary concern for certain domains (e.g. e-commerce applications), while it might be more important to perform as many tasks as possible in some other domains (e.g. multi-robot coordination in manufacturing companies). We would want a mechanism to be computationally efficient for obvious reasons - faster the agreement is reached, fewer the resources consumed. This can lead to significant savings in cost. The majority of computer systems today are established over networks that are unreliable and/or expensive. So, assuming all else is equal, a mechanism that involves less communication between the agents during the negotiation process would be preferred in a real-world setting.

Another highly desirable property in a mechanism is *stability*. A mechanism is stable if it provides all agents with an incentive to behave in a particular way. This is an essential property in multiagent systems with self-interested agents because if a self-interested agent is better off by behaving in some other manner, then it will do so. Having said that, even for many settings with cooperative agents, it would not be a far-fetched idea to assume that the agents constantly work towards improving their individual utilities within the confines of the mechanism. Therefore, stability is important in either case.

Distribution of command and decision-making is essential in certain settings where the very nature of the problem makes it infeasible to aggregate all the necessary data at some central location and perform the computation for all agents over there. For example, consider the B2B automation scenario, where individual businesses form dynamic alliances (also called virtual organizations or virtual enterprises) to perform tasks of mutual interest. In such scenarios, the companies would not be interested

in sharing their private information with others while trying to come up with an agreement. They would rather make local decisions based on existing knowledge and share only the non-sensitive information with other companies while negotiating with them. Distribution is desirable even for systems that are not inherently decentralized, as it avoids a single point of failure and minimizes performance bottleneck among other things.

Scalability is another important issue in settings where we have a large number of agents. We would want a mechanism that degrades gracefully with increasing number of agents in the system.

It is also desirable that a mechanism be *simple*. A simple mechanism is one where the choice of the negotiation protocol makes it tractable for the agents to determine their optimal strategies. Negotiation processes that are simple and efficient are preferable to complex processes, as they are feasible to build into an automated agent. It is not a mere coincidence that most widely adopted protocols are simple (e.g. monotonic concession protocol (section 4.1.1)).

Clearly, the desiderata for a mechanism presented above is by no means comprehensive. However, we believe that the above-mentioned properties are desirable for all negotiation scenarios. In this work, we present a mechanism that incorporates all of the above-mentioned properties.

1.3 Dissertation Overview

In this section, we summarize the contents of this dissertation. The dissertation is divided into four major parts - *Overview, Literature Survey, Original Research Contributions,* and *Future Extensions and Conclusions.*

Chapter 1 introduced the audience to the automated multilateral negotiation problem addressed in this dissertation, and provided an overview of the target applications

for the problem. It forms the first part of the dissertation.

The second part comprises of chapters 2-4, where we present a literature survey on the research problem. Negotiation theory is of immense interest in many fields such as economics/game theory, social sciences, and multiagent systems. We discuss some of the most notable work in each of these areas in the chapters for this part.

Our original contributions to the literature are collectively presented in the third part of this dissertation. Chapter 5 describes negotiation networks - our agent-based model for information systems studied in this work. We also provide examples of how the computer systems for various target applications can be modeled as negotiation networks. In Chapter 6 we analyze the automated multilateral negotiation problem as a negotiation network and discuss the theoretical complexity of solving it optimally. We present PACT (Progressive Anytime Convergent Time-efficient) - our approximation algorithm for solving the hyper-exponentially complex automated negotiation problem. We evaluate our mechanism empirically, and present simulation results to express the properties of our algorithm. We also apply the algorithm to a slightly modified automated negotiation problem that further extends the scope of our work. Chapter 7 addresses the stability property in our mechanism, and presents our sliding-window algorithm that always produces a nucleolus-stable solution for any negotiation network. Again, we present theoretical and experimental results for this algorithm.

Finally, chapter 8 forms the final part of this dissertation, where we describe some possible future extensions of our work and conclude respectively.

Part II

Background and Related Work

Given the ubiquity and importance of negotiations, researchers from various disciplines such as Game Theory, Sociology, and Artificial Intelligence have contributed significantly towards its theory. We present a synopsis of this work along with references for further reading in this part.

Chapter 2

Negotiation Research in Game Theory and Economics

Game Theory has its roots in the work of John Von Neumann and Oskar Morgenstern (Neumann & Morgenstern, 1944) and provides general mathematical techniques for analyzing situations in which two or more players make decisions that will influence each other's welfare. We provide a short introduction to the field of game theory here, emphasizing more on the aspects that are relevant to our work. Readers are referred to the following books (Kahan & Rapoport, 1984; Moulin, 1995; Myerson, 1997; Osborne & Rubinstein, 1999; Osborne, 2004) for a detailed description of the field.

In all game theoretic models, the basic entity is a *player*. A player may represent an individual or a group of individuals making a decision. Given the set of players, there are two types of models that are studied: *noncooperative (or strategic) games* (where the sets of possible actions of *individual* players are primitives), and *cooperative (or coalitional) games* (where the sets of joint actions of *groups* of players are primitives)[1]. The defining quality of a cooperative game is that players may enter

[1]Note that the players in cooperative games are still self-interested; the term *cooperative* simply means that the players form coalitions to maximize their expected utilities in these games.

into mutually binding agreements. Such binding agreements prior to decisions by the players are not allowed in noncooperative games. A distinction is made within noncooperative literature between *normal (or standard) form games* (where all players move simultaneously, or if they do not move simultaneously, the later players are unaware of the earlier players' actions, making them effectively simultaneous) and *extensive (or sequential) form games* (where the players move sequentially, and later players have some information about earlier actions). A further distinction is made within extensive games based on whether the players have *perfect information* (where all players know the moves previously made by all other players) or *imperfect information*. Coalition formation in cooperative game theory is usually studied in the context of *characteristic function form games*, where it is assumed that the utilities of the coalition members are independent of the nonmembers' actions.

2.1 Bargaining Problem

Negotiations among self-interested players was formally introduced by John Nash as a bargaining problem (Nash, 1950) in game theory.

Definition 2 (Bargaining problem) *A **bargaining problem** is a pair $\langle U, d\rangle$, where U is a set of pairs of numbers (the set of pairs of payoffs to agreements) and d is a pair of numbers (the pair of payoffs to disagreement), satisfying the following conditions:*

- *d is a member of U. (Disagreement is a possible outcome of bargaining - the players may agree to disagree.)*
- *For some member $(v_1, v_2) \in U$ we have $v_1 > d_1$ and $v_2 > d_2$. (Some agreement is better for both players than disagreement.)*
- *If (v_1, v_2) and (w_1, w_2) are both in U, then for every α with $0 \leq \alpha \leq 1$ the pair of payoffs $(\alpha v_1 + (1-\alpha)w_1, \alpha v_2 + (1-\alpha)w_2)$ is also in U. (The set U is convex.)*

- *U is bounded (i.e. it is a subset of a sufficiently large disk) and closed (i.e. the limit of every convergent sequence $(v^1, v^2, \ldots)$ of members of U is in U).*

Note that the set U represents the von Neumann-Morgenstern utility functions for the individual players over all agreements. Let Δ be the set of agreements (which includes the conflict deal d) in U. Then the utility function for each player i is given by $u_i : \Delta \to \Re$.

Definition 3 (Bargaining solution) *A **bargaining solution** is a function that associates with every bargaining problem $\langle U, d \rangle$ a member of U.*

2.1.1 Nash Bargaining Solution

Nash presented an axiomatic model for studying the bargaining problem, where he proposed a set of reasonable conditions, or axioms, that a bargaining solution must satisfy to be fair to both players, without actually modeling the bargaining process. He suggested five axioms for a fair outcome. First, the bargaining solution must be individually rational to both players, i.e. no player should receive a payoff lower than its disagreement payoff.

Axiom 1 (Individual rationality (IR)) *Let $\langle U, d \rangle$ be a bargaining problem. The bargaining solution does not assign (v_1, v_2) to U where $v_1 < d_1$ or $v_2 < d_2$.*

Second, the outcome should be Pareto efficient, i.e. no other agreement yields both players higher payoffs.

Axiom 2 (Pareto efficiency (PAR)) *Let $\langle U, d \rangle$ be a bargaining problem, and let (v_1, v_2) and (w_1, w_2) be members of U. If $v_1 > w_1$ and $v_2 > w_2$, then the bargaining solution does not assign (w_1, w_2) to $\langle U, d \rangle$.*

Third, in the absence of any asymmetry between the players, the outcome should give all players the same payoff. Equivalently, if the two players swap their utility functions, then the bargaining solution must swap their payoffs too.

Axiom 3 (Symmetry (SYM)) *Let $\langle U, d\rangle$ be a bargaining problem for which (v_1, v_2) is in U iff and only if (v_2, v_1) is in U, and $d_1 = d_2$. Then the pair (v_1^*, v_2^*) of payoffs the bargaining solution assigns to $\langle U, d\rangle$ satisfies $v_1^* = v_2^*$.*

Fourth, a bargaining solution depends only on player preferences, not on their payoff representations.

Axiom 4 (Invariance to equivalent payoff representations (INV)) *Let $\langle U, d\rangle$ be a bargaining problem, let α_i and β_i be numbers with $\alpha_i > 0$ for $i = \{1, 2\}$, let U' be the set of all pairs $(\alpha_1 v_1 + \beta_1, \alpha_2 v_2 + \beta_2)$, where (v_1, v_2) is a member of U, and let $d' = (\alpha_1 d_1 + \beta_1, \alpha_2 d_2 + \beta_2)$. If the bargaining solution assigns (w_1, w_2) to $\langle U, d\rangle$, then it assigns $(\alpha_1 w_1 + \beta_1, \alpha_2 w_2 + \beta_2)$ to $\langle U', d'\rangle$.*

Fifth, a bargaining solution must not change when a losing agreement is removed from the original bargaining problem.

Axiom 5 (Independence of irrelevant alternatives (IIA)) *Let $\langle U, d\rangle$ and $\langle U', d'\rangle$ be bargaining problems for which $U' \subset U$ and $d = d'$. If the agreement the bargaining solution assigns to $\langle U, d\rangle$ is in U', then the bargaining solution assigns the same agreement to $\langle U', d'\rangle$.*

The five axioms IR, PAR, SYM, INV, and IIA are sufficient to determine a unique bargaining solution. It is the Nash bargaining solution.

Definition 4 (Nash bargaining solution) *A unique bargaining solution satisfies the axioms INV, SYM, IIA, IR, and PAR. This solution assigns to the bargaining*

problem $\langle U, d \rangle$ *the pair of payoffs* (v_1, v_2) *that solves the problem*

$$\arg \max_{(v_1, v_2)} (v_1 - d_1)(v_2 - d_2)$$

subject to $(v_1, v_2) \in U$.

Apart from the Nash bargaining solution, many other axiomatic solution concepts have been proposed in welfare economics and social choice theory for the bargaining problem. We present some of the important concepts here. Note that while these solution concepts have been defined in the context of a two person bargaining problem defined above, they can be easily extended to games with more players.

2.1.2 Pareto-optimal Solution

Definition 5 (Pareto optimality) *Let* $\langle U, d \rangle$ *be a bargaining problem and* $u_i : \Delta \rightarrow \Re$ *where* $i = \{1, 2\}$ *be the utility functions for the two players over the agreements* Δ *in* U. *An outcome* $\delta \in \Delta$ *is* ***Pareto optimal (or Pareto efficient)*** *if*

$$\neg \exists \delta' \in \Delta$$

such that

$$\forall_i u_i(\delta') > u_i(\delta)$$

If a bargaining solution is Pareto efficient, then it is the case that no player can be made better off without another being made worse off. If a bargaining solution is not Pareto efficient, then there is theoretical potential for a Pareto improvement - an increase in the utility of at least one participant without reducing any other participant's utilities. It is commonly accepted that outcomes that are not Pareto efficient are to be avoided, and therefore Pareto efficiency is an important criterion for

evaluating negotiation mechanisms. A bargaining problem can have multiple Pareto-optimal solutions; the set of all these solutions is referred to as the Pareto frontier.

2.1.3 Utilitarian Solution

Definition 6 (Utilitarian solution) *Let $\langle U, d\rangle$ be a bargaining problem. A bargaining solution is a **utilitarian solution** if it assigns to the bargaining problem $\langle U, d\rangle$ the pair of payoffs (v_1, v_2) that solves the problem*

$$\arg\max_{(v_1,v_2)} \sum_{i=\{1,2\}} v_i$$

subject to

$$(v_1, v_2) \in U$$

A utilitarian solution, also referred to as the **social welfare maximizing solution**, is the one where the participants are assigned the agreement that maximizes the sum of their individual payoffs. Thus, a utilitarian solution is also a Pareto-optimal solution.

2.1.4 Egalitarian Solution

Definition 7 (Egalitarian solution) *Let $\langle U, d\rangle$ be a bargaining problem. A bargaining solution is an **egalitarian solution** if it assigns to the bargaining problem $\langle U, d\rangle$ the pair of payoffs (v_1, v_2) that solves the problem*

$$\arg\max_{(v_1,v_2)} \sum_{i=\{1,2\}} v_i$$

subject to

$$(v_1, v_2) \in U$$

and

$$v_1 = v_2$$

In an egalitarian solution, the participants seek for the agreement that yields the highest payoffs, which are then split equally among them. This is a useful solution concept in systems where the participants are working for some higher order, where there is no self-interest involved. A variation of the egalitarian solution - the egalitarian social welfare solution, is useful for systems where there is no feasible agreement that provides the same payoff for all players. This solution concept choses the agreement that maximizes the utility of the weakest player in the bargaining game.

Definition 8 (Egalitarian social welfare solution) *Let $\langle U, d\rangle$ be a bargaining problem. A bargaining solution is an* ***egalitarian social welfare solution*** *if it assigns to the bargaining problem $\langle U, d\rangle$ the pair of payoffs (v_1, v_2) that solves the problem*

$$\arg\max_{(v_1,v_2)} v_i$$

subject to

$$(v_1, v_2) \in U$$

where player i is given by

$$i = \arg\min_j v_j$$

2.1.5 Kalai-Smorodinsky Solution

Definition 9 (Kalai-Smorodinsky solution) *Let $\langle U, d\rangle$ be a bargaining problem. A bargaining solution is a* ***Kalai-Smorodinsky solution*** *if it assigns to the bargaining problem $\langle U, d\rangle$ the pair of payoffs (v_1, v_2) that solves the problem*

$$\arg\max_{(v_1,v_2)} \sum_{i=\{1,2\}} v_i$$

subject to

$$(v_1, v_2) \in U$$

and

$$\frac{v_1}{v_2} = \frac{\max v_1}{\max v_2}$$

This is another solution concept of note. The Kalai-Smorodinsky solution (Kalai & Smorodinsky, 1975; Vidal, 2006) is the agreement on the Pareto frontier that distributes the payoffs between the participants in proportion to the maximum payoff that they can get from any agreement in the system.

2.2 Bargaining Game of Alternating Offers

Presenting axiomatic models with desired properties is one way of analyzing the bargaining problem. The other way is to model the bargaining process as a noncooperative game, allow each player to behave in a manner that satisfies its personal utility maximization criterion, and analyze the equilibrium solutions that it would lead to. Ariel Rubinstein presented a model for the bargaining process between two players as an extensive game with perfect information (Rubinstein, 1982).

Definition 10 (Extensive game with perfect information) *An **extensive game with perfect information** consists of:*

- *a set of **players***
- *a set of sequences (**terminal histories**) with the property that no sequence is a proper subhistory of any other sequence*
- *a function (the **player function**) that assigns a player to every sequence that is a proper subhistory of some terminal history*
- *for each player, **preferences** over the set of terminal histories.*

The set of terminal histories is the set of all sequences of actions that may occur in the game. The player assigned by the player function to any history h is the player who takes an action after h. Note that although the definition does not directly specify the set of actions available to the players at their various moves, these can be deduced from the set of terminal histories and player function. If, for some nonterminal history h, the sequence (h, a) is a history, then a is one of the actions available to the player who moves after h. Thus the set of all actions available to the player who moves after h is $A(h) = \left\{ a : \ (h, a) \text{ is a history} \right\}$. In some situations, an outcome is associated with each terminal history, and the players' preferences are defined as payoff functions over these outcomes (as in normal games) rather than directly over the terminal histories.

Given this background, a bargaining game of alternating offers is defined as follows.

Definition 11 (Bargaining game of alternating offers) *The* ***bargaining game of alternating offers*** *is the following extensive game with perfect information.*

Players: *Two negotiators, say 1 and 2.*

Terminal histories: *Every sequence of the form* $(x^1, N, x^2, N, \ldots, x^t, Y)$ *for* $t \geq 1$*, and every (infinite) sequence of the form* $(x^1, N, x^2, N, \ldots)$*, where each* x^r *is a division of the pie (a pair of numbers that sums to 1).*

Player function: $P(\Phi) = 1$ *(player 1 makes the first offer), and*

$$P(x^1, N, x^2, N, \ldots, x^t) = P(x^1, N, x^2, N, \ldots, x^t, N) = \begin{cases} 1 & \text{if } t \text{ is even} \\ 2 & \text{if } t \text{ is odd.} \end{cases}$$

Preferences: *For* $i = 1, 2$*, player* i*'s payoff to the terminal history* $(x^1, N, x^2, N, \ldots, x^t, Y)$ *is* $\delta_i^{t-1} x_i^t$*, where* $0 < \delta_i < 1$*, and its payoff to every (infinite) terminal history* $(x^1, N, x^2, N, \ldots)$ *is 0.*

The structure of this game is as follows: player 1 makes a proposal, which player 2 either accepts or rejects; then, if player 2 rejects the proposal, it makes a proposal, which player 1 either accepts or rejects; and so on. Note that each subgame [2] is identical to the whole game - not only are the players, terminal histories, and player function the same in the subgame as they are in the game, but so too are the players' preferences (for any number k, each player i is indifferent between receiving k units of payoff with t periods of delay and receiving $\delta_i^t \cdot k$ units of payoff immediately). However, the players' payoffs are discounted by a factor of δ over each time period.

2.2.1 Nash Equilibrium

How would a self-interested player behave in an extensive game? What outcome would result when all players adopt strategies that are in their best interest? The Nash equilibrium is the most commonly used solution concept in game theory to answers these questions.

Definition 12 (Strategy) *A **strategy** of player i in an extensive game with perfect information is a function that assigns to each history h after which it is player i's turn to move (i.e. $P(h) = i$, where P is the player function) and action in $A(h)$ (the set of actions available after h).*

A **strategy profile** is a collection of strategies, one for each player. For each strategy profile $s = (s_i)_{i \in N}$ (where N is the number of players) in an extensive game, we define the **outcome** $O(s)$ of s to be the terminal history that results when each player $i \in N$ follows the precepts of s_i.

---

[2] For any non-terminal history h, the subgame following h is the part of the game that remains after h has occured. This concept is defined formally in section 2.2.2.

Definition 13 (Nash equilibrium of extensive game with perfect information) *The strategy profile s^* in an extensive game with perfect information is a* ***Nash equilibrium*** *if, for every player i and every strategy r_i of player i, the outcome $O(s^*)$ generated by s^* is at least as good according to player i's preferences as the outcome $O(r_i, s^*_{-i})$ generated by the strategy profile (r_i, s^*_{-i}) in which player i chooses r_i while every other player j chooses s^*_j. Equivalently, for each player i,*

$$u_i(O(s^*)) \geq u_i(O(r_i, s^*_{-i}) \quad \textit{for every strategy } r_i \textit{ of player } i,$$

where u_i is a payoff function that represents player i's preferences and O is the outcome function of the game.

This means that if all the other players use the strategies specified for them in the strategy profile of the Nash equilibrium, then the best strategy for any player is the one given by the Nash equilibrium. The most significant aspect of Nash equilibrium as a solution concept is that every game has at least one Nash equilibrium, as long as mixed strategies are allowed. Unfortunately, there are also some significant limitations with the Nash equilibrium concept. First, many games have multiple Nash equilibria, and it is not always clear which one the players should actually adopt. Second, a stationary Nash equilibrium does not exist for many games, and since a player often has only discrete actions available in many real-world systems, a mixed strategy might be difficult to incorporate. Third, while the Nash equilibrium is stable under single player deviations, it does not guarantee stability against multiple player deviations - a group of players can collude and defect from the proposed solution to gain a higher utility for them all.

2.2.2 Subgame Perfect Equilibrium

The notion of Nash equilibrium ignores the sequential structure of an extensive game - it treats strategies as choices made once and for all before play begins (as in normal games). A player that plays later in the extensive game should have the advantage of knowing the moves made by the other players before it has to make its own move. Instead, with Nash equilibrium, the strategy profile is determined based on perceived threats that the players might not play if they are not in their best interests. Consequently, Nash equilibrium strategies may be in equilibrium only in the beginning of the extensive games; it may be unstable in later stages. The subgame perfect equilibrium solution concept is more appropriate for extensive games.

Definition 14 (Subgame of extensive game with perfect information) *Let* Γ *be an extensive game with perfect information, with player function* P*. For any non-terminal history* h *of* Γ*, the* ***subgame*** $\Gamma(h)$ *following the history* h *is the following extensive game:*

Players: *The players in* Γ*.*

Terminal histories: *The set of all sequences* h' *of actions such that* (h, h') *is a terminal history of* Γ*.*

Player function: *The player* $P(h, h')$ *is assigned to each proper subhistory* h' *of a terminal history.*

Preferences: *Each player prefers* h' *to* h" *if and only if it prefers* (h, h') *to* $(h', h$"$)$ *in* Γ*.*

We now define the subgame perfect equilibrium solution concept.

Definition 15 (Subgame perfect equilibrium of extensive game with perfect informatio

The strategy profile s^* *in an extensive game with perfect information is a* ***subgame***

perfect equilibrium *if, for every player i, every history $h$after which it is player i's turn to move (i.e. $P(h) = i$), and every strategy r_i of player i, the terminal history $O_h(s^*)$ generated by s^* after the history h is at least as good according t player i's preferences as the terminal history $O_h(r_i, s^*_{-i})$ generated by the strategy profile (r_i, s^*_{-i}) in which player i chooses r_i while every other player j chooses s^*_j. Equivalently, for every player i and every history h after which it is player i's turn to move,*

$$u_i(O_h(s^*)) \geq u_i(O_h(r_i, s^*_{-i})) \quad \textit{for every strategy } r_i \textit{ of player } i,$$

where u_i is a payoff function that represents player i's preferences and $O_h(s)$ is the terminal history consisting of h followed by the sequence of actions generated by s after h.

The important point in this definition is that each player's strategy is required to be optimal for *every* history after which it is the player's turn to move, not only at the start of the game as in the definition of a Nash equilibrium. Thus, a subgame perfect equilibrium is a strategy profile that induces a Nash equilibrium in every subgame.

From the above discussion it must be clear that the subgame perfect equilibrium solution concept is appropriate for analyzing the bargaining game of alternating offers introduced in section 2.2. The big question is - does the bargaining game have a subgame perfect equilibrium in which each player's strategy is stationary (i.e. each player always makes the same proposal and always accepts the same set of proposals)? The answer is in the affirmative.

Definition 16 (Subgame perfect equilibrium of bargaining game of alternating offers) *The bargaining game of alternating offers has a unique subgame perfect equilibrium, in which*

- *player 1 always proposes x^* and accepts a proposal y if and only if $y_1 \geq y^*_1$*

- *player 2 always proposes y^* and accepts a proposal x if and only if $x_2 \geq x_2^*$*

where

$$x^* = \left(\frac{1-\delta_2}{1-\delta_1\delta_2}, \frac{\delta_2(1-\delta_1)}{1-\delta_1\delta_2} \right)$$

$$y^* = \left(\frac{\delta_1(1-\delta_2)}{1-\delta_1\delta_2}, \frac{1-\delta_1}{1-\delta_1\delta_2} \right)$$

The outcome of the equilibrium strategy pair is that player 1 proposes x^* at the start of the game, and player 2 immediately accepts this proposal. An informal discussion that provides insights into how this outcome is derived follows.

Note that since the players' payoffs are discounted over time, it is in their best interest to reach an agreement in the first step. Therefore, for equilibrium, each player must propose an offer that yields a sufficiently high payoff to the opponent without unnecessarily cutting into its own utility. Now, say players 1 and 2 offer each other proposals x^* and y^* respectively. Clearly, the proposals must satisfy the following conditions for equilibrium: $x_2^* = \delta_2 y_2^*$ and $y_1^* = \delta_1 x_1^*$. Also note that the players' payoffs in the bargaining game are complimentary, therefore $x_1^* + x_2^* = 1$ and $y_1^* + y_2^* = 1$. Solving these simultaneous equations gives the values for x^* and y^* in the above definition.

The subgame perfect equilibrium has some noteworthy properties:

Efficiency: The equilibrium agreement for the bargaining game of alternating offers is reached immediately; no resources are wasted in delay. Also, the agreement is pareto efficient.

Stationarity of strategies: As mentioned before, the subgame perfect equilibrium strategies are stationary for both players. Also, the players have no motivation to deviate from the proposed strategies.

Effect of changes in patience: For a given value of δ_2, the value of x_1^*, the equilibrium payoff of player 1, increases as δ_1 increases to 1. That is, assuming

that player 2's patience is fixed, player 1's share increases as it becomes more patient. Further, as player 1 becomes extremely patient (δ_1 close to 1), its share approaches 1. Symmetrically, fixing the patience of player 1, player 2's share increases to 1 as it becomes more patient.

First-mover advantage: If $\delta_1 = \delta_2 = \delta$, then the only asymmetry in the game is that player 1 moves first. Player 1's equilibrium payoff is $(1-\delta)/(1-\delta^2) = 1/(1+\delta) > 1/2$. The fact that player 1 obtains more than half of the pie indicates that there is an advantage to being the first to make a proposal.

2.2.3 Extensions to the Bargaining Game

Many extensions of the bargaining game of alternating offers have been studied in the literature. The most relevant of those with respect to our research are the variants of models that allow three or more players. However, these extensions have not been very successful in one major aspect. They have many nonstationary subgame perfect equilibria, with a wide range of equilibrium payoffs. To the best of our knowledge, there is not much work done on circumventing this problem.

2.3 Coalitional Games and Cooperative Game Theory

Our research interest is in bargaining games with multiple players, where players coordinate their activities and form coalitions with other players to further their own self interest. As mentioned earlier, the Nash equilibrium is not stable against deviation in a coordinated manner by a group of players. The subgame perfect equilibrium concept also suffers from existence and uniqueness problems. Instead of the strategic approach that uses equilibrium analysis, game theorists typically study

coalition formation as a characteristic form game (also called characteristic function games, and coalitional games), which matches more closely to the actual negotiation practices in real-world scenarios.

2.3.1 Characteristic Form Games

In a characteristic form game, each group of players is associated with a single number, interpreted as the payoff that is available to the group. A special case of the characteristic form game is the transferable payoff game, where there are no restrictions on how the payoff may be divided among the members of the group. This is the most commonly studied model in coalition games, and is the only form introduced in this report [3].

Definition 17 (Characteristic form game with transferable payoff) *A **characteristic form game with transferable payoff** consists of:*

- *a finite set N (the set of **players**)*
- *a function v that associates with every nonempty subset S of N (a **coalition**) a real number $v(S)$ (the **worth** of S).*

There are many points worth mentioning about characteristic form games. First, for each coalition S, the number $v(S)$ is the total payoff that is available for division among the members of S. That is, the set of joint actions that the coalition S can take consists of all divisions of $v(S)$ among the members of S. The coalition S forms by making a unanimous, consensual, and binding agreement on the way its value $v(S)$ is to be distributed among its members. Second, a characteristic form game models a situation in which the actions of the players who are not part of S do not influence

[3] From hereon in the document, a characteristic form game (or coalitional game) represents a transferable payoff characteristic form game, unless stated otherwise.

$v(S)$. None of the amount $v(S)$ can be given to a member of $N - S$, nor can any member of S receive payment from $N - S$. Third, the characteristic function v is assumed to be common knowledge - all players know the valuations for all coalitions. Any agreement concerning the formation of a coalition and the disbursement of its value is known to all N players as soon as it is made. The termination of negotiations with respect to a proposed agreement is also publicly known. Finally, it is commonly assumed that the characteristic form game has the property that the worth of the coalition N of all players is at least as large as the sum of the worths of the members of any partition of N, i.e. it is cohesive. Note this assumption is true for almost all real-world settings as, in the worst-case scenario, a player can cooperate with another player by rejecting every joint deal. It also ensures that it is optimal that the coalition N of all players form.

Definition 18 (Cohesive characteristic form game) *A characteristic form game* $\langle N, v \rangle$ *with transferable utility is* ***cohesive*** *if*

$$v(N) \geq \sum_{k=1}^{K} v(S_k) \quad \textit{for every partition } \{S_1, \ldots, S_K\} \textit{ of } N$$

Just as in bilateral game theory discussed above, we have to address the following question in coalitional games - which coalitions will form in situations where all players are trying to maximize their expected utilities? John von Neumann and Morgenstern adopted the approach of defining some sense of stability that all players subscribe to, and make that concept inherent in the solutions for a game. Many others have followed this approach for coalitional games and, therefore, we have many solution concepts that tell us which solutions will be agreed upon by the players in such games. We present some definitions next, and following that we describe some of the important solution concepts in n-person cooperative game theory.

Definition 19 (Coalition structure) *Let* $\langle N, v \rangle$ *be a characteristic form game. A*

coalition structure *is a division of N players in the game into mutually exclusive and exhaustive coalitions, given by*

$$CS = \{S_1, S_2, \ldots, S_m\}, \quad 1 \leq m \leq N$$

where

$$S_j \neq \Phi, \quad j = 1, \ldots, m$$

and

$$\forall_{i \neq j} S_i \cap S_j = \Phi$$

and

$$\bigcup_{S_j \in CS} S_j = N$$

Definition 20 (S-feasible payoff vector) *Let $\langle N, v \rangle$ be a characteristic form game with transferable payoff. For any profile $(x_i)_{i \in N}$ of real numbers and any coalition S we let $x(S) = \sum_{i \in S} x_i$. A vector $(x_i)_{i \in S}$ of real numbers is an S-**feasible payoff vector** if $x(S) = v(S)$.*

Definition 21 (Feasible payoff profile) *Let $\langle N, v \rangle$ be a characteristic form game with transferable payoff. A **feasible payoff profile** is an N-feasible payoff vector.*

Definition 22 (Imputation) *Let $\langle N, v \rangle$ be a characteristic form game with transferable payoff. An **imputation** of $\langle N, v \rangle$ is a feasible payoff profile x for which $x_i \geq v(\{i\})$ for all $i \in N$. Thus, an imputation is a payoff profile that is both individually rational and feasible.*

2.3.2 The Core

The core solution concept was introduced in (Gillies, 1953). The idea behind the core is analogous to that behind a Nash equilibrium of a noncooperative game: an

outcome is stable if no deviation is profitable. In the case of the core, an outcome is stable of no coalition can deviate and obtain an outcome better for all its members. For a coalitional game, the stability condition is that no coalition can obtain a payoff that exceeds the sum of its members' current payoffs. Assuming that the game is cohesive, we confine ourselves to outcomes in which the coalition N of all players forms.

Definition 23 (Core) *The **core** of the characteristic form game $\langle N, v\rangle$ with transferable utility is the set of feasible payoff profiles $(x_i)_{i\in N}$ for which there is no coalition S and S-feasible payoff vector $(y_i)_{i\in S}$ for which $y_i > x_i$ for all $i \in S$. Equivalently, the core is the set of feasible payoff profiles $(x_i)_{i\in N}$ for which $v(S) \leq x(S)$ for every coalition S.*

The search for a solution to a game can be viewed as a search for a feasible payoff profile that satisfies certain restrictions (based on the desired solution concept). Some of these restrictions can be stated in terms of rationality requirements - individual rationality, group rationality, and coalitional rationality. *Individual rationality* states that a player in a coalition will not accept any payoff less than what it could obtain by forming its own 1-person coalition (i.e. $x_i \geq v(i)$, for all $i \in N$). *Group rationality* extends the notion of individual rationality for the whole group of players. It states that the players will only accept a coalition structure if it maximizes social welfare (i.e. $\sum_{S\in CS} v(S) = v(N)$[4]. *Coalitional rationality* further extends the principle of group rationality to every subset of players in a coalition structure. It states that no combination of players will settle for less than what they can obtain by forming a

[4]For nonsuperadditive games, $v(N)$ is replaced by its superadditive cover, given by

$$\widehat{v}(N) = \max \sum_{j=1}^{P} v(S_j) \quad \text{such that } P = (S_1, \ldots, S_p) \text{ is a partition of N}$$

coalition (i.e. $x(T) \geq v(T)$ for every $T \subseteq N$). The biggest strength of the core as a solution concept for characteristic function games is that it satisfies all three forms of rationality mentioned above.

As seen above, many constraints must be satisfied for a feasible allocation of a characteristic function game to be in the core. These constraints are often too strong and result in coreless games - games where the set of core solutions is empty. Consider a cohesive characteristic form game $\langle N, v\rangle$, and a feasible payoff profile $(x_i)_{i\in N}$ for the game that lies in the core. For a coalition S in the game, we have

$$x_i \leq v(N) - v(N-i) \quad \text{for all } i \in S \quad \text{and} \quad v(S) \leq \textstyle\sum_{i\in S} x_i$$

Combining these inequalities gives us the other necessary condition for nonemptiness of the core in a cohesive coalitional game:

$$v(S) + \sum_{i\in S} v(N-i) \leq |S| \cdot v(N)$$

As can be seen from the above formula, the number of inequalities that must be fulfilled for a game to have a nonempty core increase rapidly with the number of players in the game. Therefore, most real-world situations with large number of participants are coreless.

Definition 24 (Vector of balanced weights) *Let $\langle N, v\rangle$ be a characteristic form game with transferable payoff. A balanced family of coalitions is a subset Ψ of $2^N - N$ such that there exists, for each S in Ψ, a weight δ_s, $0 \leq \delta_s \leq 1$, satisfying*

$$\textit{for all } i \in N\textit{:} \quad \textstyle\sum_{S\in\Psi_i} \delta_S = 1, \quad \textit{where } \Psi_i = \{S \in \Psi | i \in S\}$$

A mapping δ satisfying the above equations is called a ***vector of balanced weights****.*

Definition 25 (Balanced game) *Let $\langle N, v\rangle$ be a characteristic form game with*

transferable payoff. It is a ***balanced game*** *if, for every vector of balanced weights* δ*, we have*

$$\sum_{S \subset N} \delta_S \cdot v(S) \leq v(N)$$

A characteristic form game has a nonempty core if and only if it is balanced. Thus, the core is empty for many coalitional games, and this property limits its usefulness. On the other hand, there are many situations where a game has multiple core solutions. This is problematic in an axiomatic approach like the core and reduces its prescriptive powers as the players would not know which core solution to converge on. The other drawback of the core solution is that for certain classes of games (e.g. veto games), the core leads to solutions that might seem *unreasonable*. For example, consider the following 3-person game: $v(12) = 50; v(13) = 50; v(23) = 0; v(123) = 50$. The core of this game is the payoff profile $(x_1 = 50; x_2 = 0; x_3 = 0)$ for any of the possible coalition structures $\{12, 3\}, \{13, 2\}, \{123\}$. So, the necessary member of all nonzero coalitions, player 1, gets all the reward. However, one might argue that even though player 1 is necessary to any coalition, players 2 and 3 are not powerless as they can form a blocking coalition and keep player 1 from obtaining any reward since $v(1) = 0$. Therefore, they should be (either singly or jointly) entitled to some of the reward.

As suggested by the above discussion, the core solution concept is inadequate. It imposes an overly strong constraint that no deviations are allowed, ignoring the fact that a deviation may trigger a reaction that leads to a different final outcome, one where the deviating players are worse off than in the original solution! Many solution concepts discussed in the next few sections allow deviations by players under certain restrictions such as the deviation itself must be stable, or not be balanced by a counterdeviation.

2.3.3 The Stable Sets

The idea behind the stable sets (Neumann & Morgenstern, 1944) solution concept is that a coalition S that is unsatisfied with the current division of $v(N)$ can credibly object by suggesting a stable division x of $v(N)$ that is better for all the members of S and is backed up by a threat to implement $(x_i)_{i \in S}$ on its own (by dividing the worth $v(S)$ among its members). The logic behind the requirement that an objection itself be stable is that otherwise the objection may unleash a process involving further objections by other coalitions, at the end of which some members of the deviating coalition may be worse off.

An imputation x is considered an *objection* of the coalition S to the imputation y in a coalitional game if $x_i > y_i$ for all $i \in S$ and $x(S) \leq v(S)$. This is also referred to in the literature as *x dominates y via S*, denoted as $x \succ_S y$.

Definition 26 (Stable set) *A subset Y of the set X of imputations of a characteristic form game with transferable payoffs $\langle N, v \rangle$ is a **stable set** if it satisfies the following two conditions:*

Internal stability: *If $y \in Y$ then for no $z \in Y$ does there exist a coalition S for which $z \succ_S y$.*

External stability: *If $z \in X - Y$ then there exists $y \in Y$ such that $y \succ_S z$ for some coalition S.*

The stable set solution concept produces feasible payoff profiles that are individually and group rational but relaxes the coalitional rationality requirement. The core is the set of imputations to which there are no objections. Therefore, whenever the core is nonempty, it lies in the stable set. Unfortunately, the stable set is also plagued with the same limitations of emptiness and nonuniqueness as the core. The number of stable sets and the number of constraints characterizing those stable sets increases rapidly with the number of players.

2.3.4 The Bargaining Set

As mentioned earlier, the basic assumption in game theory while analyzing characteristic form games is that the characteristic function is common knowledge. Therefore, the players in a game can be interpreted to negotiate together with perfect information while attempting to reach an agreement. The outcome of the game is a settlement on an equilibrium point based on the threats and counterthreats that each player possesses. These threats and counterthreats reflect the bargaining power of the players as defined by the characteristic function. The central theme of the bargaining set theory (Aumann & Mashler, 1964) is a negotiation process taking place within a given coalition structure in which the stability of a proposed payoff allocation within each coalition of the structure is established as the result of sequences of threats and counterthreats among members of that coalition, as each player tries to maximize its own payoff. Any unsatisfied player in the current allocation for any coalition is allowed to raise a credible objection, i.e. an objection for which there is no balancing counterobjection. Unlike the stable set, the objection or the counterobjection is not required to be stable in any sense.

Let x be an imputation in a characteristic form game with transferable payoff $\langle N, v\rangle$. We define objections and counterobjections in the bargaining set theory as follows:

- A pair (y, S), where S is a coalition and y is an S-feasible payoff vector, is an **objection** of player i against player j to x if S includes i but not j and $y_k > x_k$ for all $k \in S$.

- A pair (z, T), where T is a coalition and z is a T-feasible payoff vector, is a **counterobjection** to the objection (y, S) of player i against player j if T includes j but not i, $z_k \geq x_k$ for all $k \in T - S$, and $z_k \geq y_k$ for all $k \in T \cap S$.

An objection of i against j to x specifies a coalition S that includes i but not j and

a division y of $v(S)$ that is preferred by all members of S to x. A counterobjection to (y, S) by j specifies an alternative coalition T that contains j but not i and a division of $v(T)$ that is at least as good as y for all the members of T who are also in S and is at least as good as x for the other members of T. Given this, the bargaining set is defined as follows.

Definition 27 (Bargaining set) *The **bargaining set** of a characteristic form game with transferable payoff $\langle N, v\rangle$ is the set of all imputations x with the property that for every objection (y, S) of any player i against any player j to x there is a counter-objection to (y, S) by j.*

The bargaining set solution concept has many desirable properties. First, it remedies the emptiness problem of other classical solution concepts such as the core and the stable set. A bargaining set stable solution exists for all coalition structures of all characteristic form games. Second, when the core is nonempty, it is included in the bargaining set. Note that an imputation is in the core if and only if no player has an objection against any other player. Third, the basic notions of objection and counterobjection are amenable to psychological interpretation, which strongly enhances the potential usefulness of bargaining set theory. Unlike the core and the stable set theories which require stability in terms of coalitions not being able to disrupt group rational payoff allocations, the bargaining set relaxes this requirement of group rationality (along with coalitional rationality) and instead, defines stability in terms of objections and counterobjections that may be lodged against individually rational payoff profiles.

Unfortunately, even the bargaining set has its drawbacks. While the computational complexity for deriving bargaining set solutions is much less than that for the stable set, it is still exponential in the number of players. The size of the bargaining set for a game is typically large, which limits its usefulness as a prescriptive solution concept for coalitional games. Finally, just like the core, the bargaining set also

proposes outcomes that seem unreasonable for certain classes of games like the veto games.

2.3.5 The Kernel

The kernel (Davis & Mashler, 1965) solution concept is similar to the bargaining set in that it is defined by the condition that to every objection there is a counterobjection, but differs from the bargaining set in the nature of objections and counterobjections that are considered effective. Instead of defining stability in terms of potential threats, the kernel balances the various alternative prospects that are available to the players or coalitions by fairly distributing the excesses.

Definition 28 (Excess of a coalition) *Let x be an imputation in a characteristic form game with transferable payoff $\langle N, v\rangle$. For any coalition S, the **excess** of S is given by*

$$e(S, x) = v(S) - x(S)$$

The excess of S therefore represents the total amount that the prospective members of coalition S collectively gain or lose (depending on the sign of the excess) if they withdraw from the coalition structure implied by the imputation x to form coalition S.

A player i objects to an imputation x by forming a coalition S that excludes some player j for whom $x_j > v(\{j\})$ and pointing out that it is dissatisfied with the sacrifice or gain of this coalition. Player j counterobjects by pointing to the existence of a coalition that contains j but not i and sacrifices more (if $e(S, x) > 0$) or gains less (if $e(S, x) < 0$). We define objections and counterobjections in the kernel as follows:

- A coalition S is an **objection** of i against j to x if S includes i but not j and $x_j > v(\{j\})$.

- A coalition T is a **counterobjection** to the objection S of i against j if T includes j but not i and $e(T, x) \geq e(S, x)$.

Definition 29 (Kernel) *The **kernel** of a characteristic form game with transferable payoff $\langle N, v \rangle$ is the set of all imputations x with the property that for every objection S of any player i against any other player j to x there is a counterobjection of j to S.*

For any two players i and j and any imputation x define $s_{ij}(x)$ to be the maximum excess of any coalition that contains i but not j:

$$s_{ij}(x) = \max_{S|i \in S, j \notin S} e(S, x)$$

Then we can alternatively define the kernel as the set of imputations $x \in X$ such that for every pair (i, j) of players either $s_{ji}(x) \geq s_{ij}(x)$ or $x_j = v(\{j\})$.

Thus, all players in the kernel are in an equilibrium where symmetric players receive equal payoffs, and the more desirable players get a higher payoff than the less desirable ones. The kernel of a characteristic function game is always nonempty, and is always a subset of the bargaining set. The calculations required to find the kernel, while still formidable in the general case, are considerably less than those for the bargaining set. The size of the kernel is generally at least an order of magnitude smaller than the bargaining set. The major sticking point with the kernel is that, unlike the core and the bargaining set, it requires interpersonal comparison of utilities. This requires that the 'intensity of feeling' of each player towards its utility units be the same.

2.3.6 The Nucleolus

Just like the bargaining set and the kernel, the nucleolus (Schmeidler, 1969) solution concept can also be defined in terms of objections and counterobjections. The nucle-

olus is closely related to the kernel in that it is also based on the notion of sharing the excesses fairly. However, instead of players individually employing excesses in pairwise comparisons against each other like in the kernel, the excesses of each coalition are compared without regard to the coalition's membership.

Let x be an imputation in a characteristic form game with transferable payoff $\langle N, v\rangle$. The objections and counterobjections for the nucleolus are defined as follows:

- A pair (S, y) consisting of a coalition S and an imputation y is an **objection** to x if $e(S, x) > e(S, y)$ (i.e. $y(S) > x(S)$).
- A coalition T is a **counterobjection** to the objection (S, y) if $e(T, y) > e(T, x)$ (i.e. $x(T) > y(T)$) and $e(T, y) \geq e(S, x)$.

Definition 30 (Nucleolus) *The **nucleolus** of a characteristic form game with transferable payoff $\langle N, v\rangle$ is the set of all imputations x with the property that for every objection (S, y) to x there is a counterobjection to (S, y).*

Thus, an imputation is stable according to the nucleolus if there is no coalition S whose excess can be reduced without increasing the excess of some other coalition to a level at least as large as that of the original excess of S. This definition of the nucleolus is not standard, but it facilitates a comparison with the kernel and the bargaining set. We now present an equivalent, standard definition for the nucleolus.

For any imputation x let $S_1, \ldots, S_{2^{|N|}-1}$ be an ordering of the coalitions for which $e(S_l, x) \geq e(S_{l+1}, x)$ for $l = 1, \ldots, 2^{|N|} - 2$ and let $E(x)$ be the vector of excesses defined by $E_l(x) = e(S_l, x)$ for all $l = 1, \ldots, 2^{|N|} - 1$. Let $B_1(x), \ldots, B_K(x)$ be the partition of the set of all coalitions in which S and S' are in the same cell if and only if $e(S, x) = e(S', x)$. For any $S \in B_k(x)$ let $e(S, x) = e_k(x)$, so that $e_1(x) > e_2(x) > \ldots > e_K(x)$.

Given two imputations x and y, we say that $E(x)$ is lexicographically less that $E(y)$ if $E_l(x) < E_l(y)$ for the smallest l for which $E_l(x) \neq E_l(y)$, or equivalently if

there exists k^* such that for all $k < k^*$ we have $|B_k(x)| = |B_k(y)|$ and $e_k(x) = e_k(y)$, and either $(i) e_{k^*}(x) < e_{k^*}(y)$ or $(ii) e_{k^*}(x) = e_{k^*}(y), |B_{k^*}(x)| < |B_{k^*}(y)|$.

Then, the nucleolus of the coalitional game is the set of imputations x for which the vector $E(x)$ is lexicographically minimal.

The nucleolus is nonempty and unique for each coalition structure of every characteristic form game. It is the subset of the kernel as well as the core (whenever nonempty). Therefore, it inherits the desirable properties of symmetry and desirability from the kernel, and satisfies individual, group, and coalitional rationality whenever the core is nonempty. The computation of the nucleolus is also considerably simpler than that of the kernel or the bargaining sets.

2.3.7 Equal Share Analysis

The theory of equal share analysis (Selten, 1972) is founded on reasoning similar to the core, but incorporates additional assumptions regarding norms of equality and perception of power. It can be expressed in the form of three hypotheses about the outcomes of characteristic form games. The first hypothesis states that any coalition structure in the solution space must be exhaustive (i.e. no two coalitions in the coalition structure would be better off by merging together). The second hypothesis introduces the concept of equal division core.

Definition 31 (Equal division core) *An imputation x for a particular coalition structure in a characteristic form game with transferable payoff $\langle N, v \rangle$ is in the* ***equal division core*** *if there is no coalition $S \subseteq N$ such that*

$$x_i < \frac{v(S)}{|S|} \quad \textit{for all } i \in S$$

Thus, an imputation is in the equal division core if no coalition can divide its value equally among its members and in this way give more to each of the members

than they receive in the imputation. The second hypothesis states that the outcome will lie in the equal division core. Finally, the third hypothesis is that, within a formed coalition, stronger members will receive payoff at least equal to those of weaker members.

The equal share analysis solution resembles the core, and the latter is indeed a subset of the former when nonempty. The equal share analysis has the added advantage of always being nonempty for every characteristic form game. However, its solutions are limited to certain coalition structures, and some Pareto optimal coalition structures have empty solution spaces. The size of the equal share analysis solution space is typically large relative to the space of feasible outcomes. The theory is also not conducive to many classes of coalitional games.

2.3.8 Equal Excess Theory

Unlike the solution concepts presented earlier, in which an imputation under consideration is tested to see if it satisfies certain requirements of stability, the equal excess theory (Komorita, 1979) posits assumptions of reasonableness and fairness and derives an allocation of a coalition's value accordingly.

Consider a characteristic form game with transferable payoff $\langle N, v \rangle$. According to the equal excess theory, the bargaining process among the players in the game executes over multiple rounds, and each player i has an *expectation* $E^r(i, S)$ of what it will obtain from each coalition S of which it is a member in each round r. The player expectations for various coalitions begin with an equal division, and are determined in later rounds from their expectations in the previous round. For a player i and coalition S, let $A^r(i, S)$ denote the highest expectation from a set of valid coalitional alternatives in round r. Then,

$$A^r(i, S) = \max_{T \neq S} [E^r(i, T)]$$

It is implicitly assumed that

$$\sum_{i \in S} E^r(i, S) = v(S) \quad \text{for all } r \text{ and all } S \subseteq N$$

Therefore, to create player expectations for round $r+1$, each member of coalition S is allowed to claim their best alternative from round r and the remainder from the sum of these claims is divided into equal parts and added to each player's claim. That is,

$$E^{r+1}(i, S) = A^r(i, S) + \frac{v(S) - \sum_{j \in S} A^r(j, S)}{|S|}$$

The players rank their coalitions based on their expectations, and negotiate with the members of the coalitions at the top of their preference order. While the equal excess theory does not explicitly define when a coalition forms, it is implicitly assumed to form when all its members are satisfied by their current expectations.

The strongest property of equal excess theory is that, unlike the solution concepts described above, it does not fix the coalition structure while determining a suitable payoff profile for the characteristic form game. Instead, it presents a bargaining process to compute both a coalition structure and a feasible payoff profile for a game. The theory is also psychologically appealing in the sense that it is intuitive and rewards the players in proportion to their strengths as determined by the characteristic function for the game. On the flip side, simulation results show that the bargaining process does not converge for every game. Instead, it either falls into a cyclic equilibrium where few expectations keep toggling between some values, or the expectation of the most powerful player in a coalition keeps increasing infinitely at the expense of other members. Also, many concepts in the theory are not defined crisply. For example, the theory does not specify when the bargaining process terminates, or which coalitions are chosen by the players (or when are they chosen).

2.3.9 The Shapley Value

All solution concepts considered so far rely on an *a posteriori* analysis of the imputations proposed as solutions of a coalitional game. The Shapley value (Shapley, 1953) gives us a feasible payoff profile for the given game and the particular coalition structure under consideration, determined by the *a priori* worth to each player of playing the game. Each player's payoff in the Shapley value corresponds to its marginal contribution to the chosen coalition structure of the game.

Definition 32 (Marginal contribution) *The* ***marginal contribution*** *of player i to any coalition S with $i \notin S$ in a characteristic form game with transferable payoff $\langle N, v\rangle$ is given by*

$$\Delta_i(S) = v(S \cup \{i\}) - v(S)$$

Definition 33 (Shapley value) *Let $\langle N, v\rangle$ be a characteristic form game with transferable payoff. The* ***Shapley value*** *Φ defining the payoffs for individual players in the game is given by*

$$\phi_i(N, v) = \tfrac{1}{|N|!} \textstyle\sum_{\pi \in \Pi} \Delta_i(S_i(\pi)) \quad \textit{for each } i \in N,$$

where Π is the set of all $|N|!$ orderings of N and $S_i(\pi)$ is the set of players preceeding i in the ordering π.

The Shapley value can alternatively be defined as

$$\phi_i(N, v) = \textstyle\sum_{S \subseteq N} \frac{(|N|-|S|)!\cdot(|S|-1)!}{|N|!} [v(S) - v(S - \{i\})] \quad \textit{for each } i \in N$$

We can interpret the Shapley value as follows. Suppose that all the players are arranged in some order, all orders being equally likely. Then $\phi_i(N, v)$ is the expected marginal contribution over all orders of player i to the set of players who precede

it. Note that the sum of the marginal contributions of all players in any ordering is $v(N)$, so that the Shapley value is always feasible.

The Shapley value payoff profile is unique and exists for each coalition structure for any characteristic form game. It is guaranteed to satisfy individual rationality, but does not necessarily satisfy group and coalitional rationality. It need not be a member of the core even if the latter is not empty. The solution concept satisfies the desirable property of symmetry among players, and gives more desirable players a higher payoff than less desirable ones. The biggest drawback of Shapley value is the computational complexity.

2.4 Issues with Game-theoretic work

Game theory provides us with many useful techniques for mathematically analyzing situations of conflicting interests among the players. However, there are also several limitations of the game theoretic solutions which prevent us from applying it directly to automated negotiations:

- Game theory does not provide algorithms that can be embedded into the agents for forming coalitions. Game theoretic solution concepts are axiomatic, not descriptive.
- Computing game theoretic solutions is usually exponentially complex, as game theory is not concerned with the computational aspects of the solution concepts.
- Some stability solution concepts provide multiple equilibria points with no mechanisms for selecting among them. This limits the applicability of such solution concepts while engineering multiagent systems.
- Game theoretic solution concepts are inherently centralized as they require the computing entity to have complete, global knowledge.

- The solutions are static, i.e. stable payoff divisions are proposed for fixed coalition structures; agents are not allowed to change their existing coalitions.

Negotiation research in artificial intelligence and multiagent systems adopts a heuristics based approach to overcome these limitations (Jennings et al., 2001).

Chapter 3

Negotiation Research in Social Sciences

Sociologists study how humans act in situations where there is a conflict of interest. Instead of assuming humans as purely rational, utility-maximizing entities as in game theory, sociologists adopt a more descriptive approach and build their models based on how humans actually behave in conflicting situations. We present one of the most widely accepted theories for modeling such problems in sociology - the Network Exchange Theory.

3.1 Network Exchange Theory

The Network Exchange Theory (Willer, 1999; Vidal, 2006) studies the effects of power on the outcomes of exchanges between people in power relation networks. In a network, the nodes are the human participants and any two nodes can negotiate (for dividing a resource or exchanging goods) if they have an edge between them. The edges represent the amount the agents are trying to divide. Based on extensive studies with human subjects, sociologists have been able to identify equations that can predict the outcome of human negotiations in particular networks.

According to the theory, each agent has a **resistance** to each particular payment p given by the agent's resistance equation. Agent i's resistance to receiving payment p is given by,

$$r_i = \frac{p_i^{max} - p_i}{p_i - p_i^{con}}$$

where,

$$p_i^{max} = \text{Maximum revenue that } i \text{ can get in the network}$$

and

$$p_i^{con} = \text{Conflict deal, worth revenue } 0$$

So, if two agents i and j want to divide some revenue p^{total} between them, they will have a resistance of r_i and r_j towards getting p_i and p_j respectively.

The Network Exchange Theory states that under these circumstances, an exchange occurs between the two agents at the **equi-resistance point**.

$$r_i = \frac{p_i^{max} - p_i}{p_i - p_i^{con}} = \frac{p_j^{max} - p_j}{p_j - p_j^{con}} = r_j$$

such that

$$p_i + p_j = p^{total}$$

We can solve complex networks using the **iterated equi-resistance algorithm**. The algorithm simply uses the equi-resistance equation repeatedly on each edge of the graph in order to calculate the payments that these agents can expect to receiver. This is repeated for all edges until payments stop changing.

The iterated equi-resistance algorithm is not guaranteed to converge and it might converge to different solutions when the edges are sorted in differently. Even when it does converge the deal it reaches is not guaranteed to be the one that humans would reach, that would mean we can predict human behavior. However, many experiments have been performed with humans in small networks (less than 12 nodes) which have

shown that the iterated equi-resistance algorithm correctly predicts the resulting deal.

Chapter 4

Negotiation Research in Artificial Intelligence and Multiagent Systems

As mentioned in section 2.4, there are many limitations with research results from game theory that make them difficult to adopt in practical settings. Research in artificial intelligence and multiagent systems focuses on devising tractable solutions for the negotiation problem. Many approaches make simplifying assumptions about the negotiation problem, making it more specific to particular domains, and provide optimal solutions for them. Others focus on devising approximate solutions without any guarantees of optimality. We now present an overview of some of this research. Interested readers can refer to the following books for further details: (Raiffa, 1982; Rosenschein & Zlotkin, 1994; Kraus, 2001; Wooldridge, 2002; Vidal, 2006).

4.1 Mechanism for Automated Negotiations

Some of the earliest work in devising mechanisms for automating negotiations among autonomous agents was conducted by Rosenschein and Zlotkin (Rosenschein & Zlotkin,

1994). They addressed the problem of facilitating rational agents with conflicting interests to arrive at an agreement that is mutually beneficial. The tools from game theory were used for designing and evaluating their mechanisms. They adopted an intuitive approach, where the agents go through a coordination process involving negotiations over all possible agreements covering issues of common interest, eventually bringing them all to a consensus. There are three main issues in defining such a coordination process: defining the space of possible deals, the negotiation process, and the negotiation strategy.

Space of Possible Deals: This represents a finite set of candidate deals for the agents to consider (e.g. agents may only consider payoff divisions where the total value does not exceed $100). Possible proposals that the agents can make are restricted by this set.

Negotiation Process: This is a negotiation protocol, which, given the set of possible deals, defines how the agents will converge to an agreement on a single deal. It specifies the set of rules that govern the agent interactions while they attempt to reach a consensus. The process explicitly defines the various negotiation states (e.g. accepting bids, negotiation closed), the events that cause negotiation states to change (e.g. no more bidders, bid accepted), and the valid actions for the agents in particular states (e.g. which messages can be sent by whom, to whom, at what stage). The negotiation process also clearly defines the rule(s) that determines when a deal is struck, and what this agreement deal is.

Negotiation Strategy: Given a set of possible deals and a negotiation process, a negotiation strategy represents a model that individual agents employ to make decisions and achieve their objectives. The negotiation protocol as well as certain agent characteristics (e.g. whether the agent has complete knowledge of its

environment, whether it is truthful) determine the sophistication of the agent decision model.

The above parameters lead to a rich and complex environment for analyzing negotiation problems. Negotiations typically involve a series of rounds, with each agent making a proposal at every round, before eventually converging to an agreement. However, there are many attributes that can complicate this model in real-world negotiation settings. For example, the complexity of the model increases rapidly with the number of agents in the negotiation process. Not only does the negotiation space explode, but also the negotiation strategies of the individual agents become more complex. Also, the agents in real-world settings typically have multiple issues over which agreement must be reached. Say there are n issues that the agents have to negotiate over where each issue can have m possible values. This leads to a set of m^n possible deals. To make matters worse, in many situations the number of issues in the bargaining process may itself vary based on other parameters.

Devising a negotiation mechanism entails defining a negotiation protocol and a negotiation strategy (or strategies) for the agents in the system. The choice of the negotiation protocol in conjunction with the negotiation strategies adopted by the agents determines the type of outcome that is produced by the mechanism. As mentioned earlier in the document, it is desirable that the mechanism be simple. The most widely adopted protocol for bilateral negotiations is the monotonic concession protocol, which we describe next. Then we describe a suitable negotiation strategy for the monotonic concession protocol - the Zeuthen strategy. We also present the one-step protocol that allows the agents to reach an agreement in just one step.

4.1.1 Monotonic Concession Protocol

The monotonic concession protocol (Rosenschein & Zlotkin, 1994) is a simple, bilateral negotiation protocol and a direct formalization of a person's natural negotiation

behavior. It also adheres to the Rubinstein's bargaining model of alternating offers 2.2.

Let $A = \{1, 2\}$ be a set of two agents and let NS be a finite set of *potential deals* Each of the agents $i \in A$ is equipped with a utility function $u_i : NS \rightarrow \Re^+$ mapping deals to non-negative reals. NS also includes a *conflict deal* yielding utility 0 to both agents, which is chosen in case the negotiations break down. The monotonic concession protocol proceeds in *rounds*, with both agents simultaneously proposing deals from the negotiation set NS in each round. In the first round, each agent is free to make any proposal. In any subsequent round, each agent $i \in A$ has two options (let $\delta_i \in NS$ be the most recent deal proposed by agent i):

- Make a concession and propose a new deal δ_i' that is preferable to the other agent j: $u_j(\delta_i') > u_j(\delta_i)$.
- Refuse to make a concession and stick to proposal δ_i.

An *agreement* is reached when one agent makes a proposal that its opponent rates at least as high as its own current proposal, i.e. when $u_1(\delta_1) \leq u_1(\delta_2)$ or $u_2(\delta_2) \leq u_2(\delta_1)$ or both. On the other hand, a *conflict* arises when both agents refuse to make a concession during a given round. The protocols runs until either an agreement is found or a conflict occurs. In case of an agreement, the proposal satisfying both agents is chosen (in case both proposals are satisfiable, one is selected at random). In case of a conflict, the conflict deal is the outcome of the negotiation.

The monotonic concession protocol is verifiable, and guaranteed to terminate. However, it is inefficient and take a long time to converge. It also has this potentially undesirable final step where both agents make a concession that is acceptable to the other agent, but only one concession can be accepted, to the chagrin of its proposer. Also, an agent has to determine its utilities for all deals in the negotiation set, which might be computationally expensive as the negotiation set is typically huge.

4.1.2 Zeuthen Strategy

What would be a suitable strategy to adopt when following the monotonic concession protocol? There are three key questions to be answered for that: What should an agent's first proposal be? On any given round, who should concede? Finally, if an agent concedes, then how much should it concede?

Obviously, it would be in the best interest of an agent if its first proposal is its most preferred deal. The Zeuthen strategy (Zeuthen, 1930) can be used to answer the other two questions. Zeuthen proposed that agents should evaluate their respective *willingness to risk conflict*, and that the agent with the lower value for this measure should make the next concession. An agent's willingness to risk conflict at round t is the quotient of the loss in utility incurred by conceding and accepting the other agent's proposal and the loss incurred by not conceding and causing conflict. Therefore, the willingness of agent i to risk conflict (with j being the other agent) is given by

$$risk_i^t = \begin{cases} 1 & \text{if } u_i(\delta_i^t) = 0, \\ \frac{u_i(\delta_i^t) - u_i(\delta_j^t)}{u_i(\delta_i^t)} & \text{otherwise.} \end{cases}$$

The Zeuthen strategy also postulates that the conceding agent's concession should be *just enough* to ensure that the other agent will have to concede in the following round. If the agent does not concede enough, then in the next round, the balance of risk will indicate that it still has most to lose from conflict, and so should concede again. On the other hand, if the agent concedes too much, then it unnecessarily cuts into its own utility.

The Zeuthen strategy is guaranteed to terminate, and the agreement that it reaches upon termination is guaranteed to be Pareto optimal and individually rational. Furthermore, it has been shown that two agents using the Zeuthen strategy will converge to the Nash bargaining solution (Harsanyi, 1956, 1977; Rosenschein &

Zlotkin, 1994). While the strategy is not stable in situations where the risk values for both agents are equal, an Extended Zeuthen strategy (Rosenschein & Zlotkin, 1994), where an agent is randomly selected to concede in equal risk situations, is in Nash equilibrium.

4.1.3 One-Step Protocol

As mentioned above, the monotonic concession protocol is inefficient in that it is computationally expensive and it also takes multiple steps to converge. The alternative, one-step protocol (Rosenschein & Zlotkin, 1994) simplifies this matter. Here, each agent is allowed to make a single offer, and the proposal that yields the higher product of utilities is adopted. Situations where the products are equal are resolved by flipping a coin. The best strategy that agents can follow in this protocol is to propose the agreement that is best for themselves amongst those with maximal product of utilities. This is clearly both stable and efficient.

4.2 Automated Negotiations in Complex Settings

The mechanisms considered in the previous section are applicable only to simple settings with two agents that negotiate over a single issue. As mentioned earlier, negotiation settings in real-world settings are much more complex due to at least two factors - there are usually many issues to be addressed, and there are typically more than two participants in the negotiation. Artificial intelligence techniques have been combined with methods and techniques from many other multi-entity fields such as game theory, operations research, physics, and philosophy to yield useful mechanisms for such complex domains. (Kraus, 1997; Sandholm, 1999) present an excellent overview of negotiation research in multiagent systems. We present a brief literature survey of the key models used for reaching agreements in multiagent environments.

4.2.1 Auctions

Auctions present a very simple mechanism for agents to reach agreements on the distribution of a set of items. Agents (the *bidders*) can define how much they value each item (or a group of items) by placing a bid for it. All these bids are analyzed by a central agent (the *auctioneer*) who then decides on the allocation of items to the bidders. There are two patterns of interactions in auctions - one-to-many auction protocols, and many-to-many auction protocols (Wurman, Walsh, & Wellman, 1998). In *one-to-many* interactions we have one agent that initiates the auction and a number of other agents can bid in the auction. These are the most common type of interactions observed in auction houses such as eBay (eBay, 2001). The *many-to-many* auction protocols have several agents initiate an auction and several other agents can bid in the auction. Given the pattern of interaction, the first issue to determine is the type of protocols to use in the auction. Given the protocol, the agents need to decide on their bidding strategy.

There are several types of one-to-many auctions that are used, including the English auction, the Dutch auction, the first-price sealed-bid auction, and the second-price sealed-bid auction (the Vickrey auction). These auction protocols differ across various dimensions, including how the winner is determined, whether or not the bids are common knowledge, and how the bidding proceeds in the auction. The *winner determination problem* is the problem of deciding who gets the item that all agents are bidding for, and how much does the winner pay for it. Obviously, the agent that bids the most gets the item. However, two possibilities are observed for the second question in the actual auction settings. If the agent has to pay its quoted amount for the item, it is known as *first-price* auctions. In some settings, the winning agent has to pay only the amount quoted by the second highest bidder; these are known as *second-price* auctions. The second dimension along which auction protocols can vary is whether or not the bids are *common knowledge* (i.e. whether or not the bids

made by the agents are known to each other). If every agent can observe what every other agent is bidding, it is an *open-cry* auction, otherwise it is a *sealed-bid* auction. A third dimension is the mechanism by which the bidding proceeds. The simplest possibility is to have a single round of bidding, after which the auctioneer allocates the item to the winner. Such auctions are known as *one shot*. The second possibility is that the price starts low (often at a *reservation price*) and successive bids are for increasingly large amounts. These are known as *ascending* auctions. The alternative - *descending* - is for the auctioneer to start off with a high value, and to decrease the price in successive rounds.

The *English* auction is an ascending, first-price, open cry auction. In one variant of the English auction the auctioneer calls successively higher prices until only one willing bidder remains, and the number of active bidders is publicly known at all times. In other variants the bidders call out prices themselves, or have the bids submitted electronically and the best current bid is posted. The dominant strategy for an agent in English auctions is to bid a small amount more than the current highest bid until the bid price reaches its current valuation, and then to withdraw. The *Dutch* auction is an open-cry, descending auction where the auctioneer begins by naming a very high price and then lowers it continuously until some bidder stops the auction and claims the object for that price. There is no dominant strategy for Dutch auctions in general. Both English and Dutch auctions suffer from what is known as the *winner's curse* - a situation where the winning agent worries if it has overvalued the item on offer. The first-price sealed-bid auction is an example of a one-shot auction, and is arguably the simplest of all auction types. The best strategy for an agent in this auction is to bid *less* than it's true valuation. This is the case because the winner in this auction knows that it has to bid only slightly higher than the second highest bidder for the item, and anything more is, in effect, wasted money. How much less should an agent bid is of course dependent on what the other agents bid - there is no

general solution. Finally, the Vickrey auction (Vickrey, 1961) is a second-price sealed bid auction. It is also a one-shot auction where the highest bidder wins the item on offer. Only, here the winner pays the price quoted by the second highest bid. The significance of this apparently weird auction protocol is that, under some assumptions, it is *incentive compatible*, i.e. it makes truth telling the dominant strategy for the bidders. Consider this to see why. Suppose the agent bids more than its true valuation for an item. In this case, it runs the risk of winning the item at a price higher than its private valuation, thereby incurring a loss. Suppose the agent bids less than its true valuation. Here it risks losing the item unnecessarily. Also, even if it does win, the amount that it has to pay is not affected by bidding less. Therefore, it is in the best interest of the agent to bid its true valuation. Unfortunately, as described in (Sandholm, 1996, 1999), the Vickrey auction is susceptible to bidder collusion and lying auctioneers. (Klemperer, 1999; Monderer & Tennenholtz, 2000) present further details on the advantages and disadvantages of various auctions in real-world settings.

There are situations in which the value of some item to a bidder depends on other items that it wins. In such cases, bidders may want to submit bids for combinations of items. Such auctions are called *combinatorial auctions* (Cramton, Shoham, & Steinberg, 2006). The main problem in combinatorial auctions is to determine the revenue-maximizing set of nonconflicting bids. The general problem is NP-complete and inapproximable (Rothkopf, Pekec, & Harstad, 1998; Lehmann, Muller, & Sandholm, 2006). However, in practice, modern search algorithms can optimally solve winner determination in the large (Sandholm, 2002; Fujishima, Leyton-Brown, & Shoham, 1999; Sandholm, 2006).

Double auction is the most well-known auction protocol for many-to-many auctions. In a double auction, buyers and sellers are treated symmetrically, with buyers submitting bids and sellers submitting minimal prices (Wilson, 1985). There are several algorithms used for matching buyers and sellers and for determining the transac-

tion price. Preferably, the protocol will be incentive compatible, individual rational, and Pareto optimal (Wurman et al., 1998).

4.2.2 Coalition Formation

Coalition formation is an important form of coordination and cooperation in multiagent systems. It enables the agents to satisfy tasks that they would otherwise be unable to perform, or would perform with a lower efficiency. It is a useful form of interaction in multiagent settings with both cooperative and competitive agents. As mentioned in section 2.3, coalition formation is typically studied as a characteristic form game where the value of each coalition is given by a characteristic function for the game. When applied to multiagent systems, the coalition formation problem comprises of three important steps:

Determination of coalition values: This involves the computation of utilities of all $2^{|A|}$ possible coalitions in a multiagent system with $|A|$ agents.

Optimal coalition structure generation: The agents must be partitioned into disjoint and exhaustive coalitions such that the partition is optimal in the desired sense. While the optimality criterion is domain-specific, most studies in multiagent systems aim for a social welfare maximizing solution. However, the total number of coalition structures is $O(|A|^{|A|})$ and $\omega(|A|^{|A|/2})$, and finding the coalition structure that maximizes the overall utility of the system is NP-complete (Sandholm, Larson, Anderson, Shehory, & Tohmé, 1999).

Stable payoff distribution: This involves the distribution of utilities within coalition members such that they are motivated to stay in the coalition assigned to them. The solution concepts from game theory are employed in establishing stability in coalitions. As mentioned earlier2.4, this step is also computationally expensive. This step is not required for cooperative multiagent systems.

Algorithmic approaches towards splitting the coalition value among the members such that they yield stable solutions was addressed for certain cases by game theorists. *Transfer schemes* in game theory represent a dynamic approach to the payoff division activity of coalition formation in characteristic form games (Kahan & Rapoport, 1984). The coalition structure is fixed in these schemes, and the agents iteratively exchange payments in a prespecified manner. (Stearns, 1968) provides two transfer schemes, one that converges to the kernel, and another that converges to the bargaining set of a coalitional game. (Wu, 1977) presents a transfer scheme that leads to a core solution in a coalition game when it is nonempty.

Some of the earliest work in coalition formation for multiagent systems was presented in (Zlotkin & Rosenschein, 1994). They studied the problem of utility division in subadditive task-oriented domains, which are a strict subset of characteristic form games [1]. They consider only the grand coalition structure, where all the agents belong to the same coalition, and provide a linear algorithm that guarantees each agent an expected utility that is equal to its Shapley value 2.3.9. In their algorithm, each agent gets paid its marginal contribution to the coalition. Since this depends on the order in which the agents join the coalition, the joining order is randomized in a nonmanipulable way by a trusted third party.

(Ketchpel, 1994) presents a coalition formation method which simultaneously addresses both coalition structure generation and payoff distribution in situations where there is uncertainty in the utilities of the coalitions. The adopted approach involves a pairwise auction protocol where, for each coalition, the agent that values the collaboration more highly is selected to manage the coalition and it offers an agreement to other members based on their valuations. This algorithm uses cubic time in the number of agents, but each individual step may be arbitrarily complex (depending

[1] In a subadditive task-oriented domain, for any two coalitions $S_1, S_2 \in N$, we have $v(S_1 \cup S_2) \leq v(S_1) + v(S_2)$.

on how an agent decides on its valuation of its coalition).

(Shehory & Kraus, 1998) discusses coalition formation to perform tasks in cooperative multiagent systems, where only the optimal coalition structure generation problem needs to be addressed. They provide an efficient, distributed algorithm that greedily selects feasible, highly valued coalitions. The complexity of the problem is reduced by limiting the number of agents per coalition. Their greedy algorithm guarantees solutions with low ratio bounds from the optimal solution (given the limit on the number of agents) and low computational complexities. They consider both types of scenarios, where each agent must be a member of only one coalition, and where overlapping coalitions are allowed.

(Shehory & Kraus, 1999) consider coalition formation among self-interested agents with perfect information in characteristic form games. Both the coalition structure and the payoff division problems are addressed. An exponentially complex anytime algorithm for forming coalitions that satisfy the kernel 2.3.5 stability criterion is developed. They also provide another protocol with reduced complexity that requires only a weaker form of kernel stability. They use Stearn's transfer scheme introduced above in their algorithms.

(Sandholm et al., 1999) focuses on establishing the worst case bound on the coalition structure quality while searching only a fraction of the coalition structures. As mentioned earlier, they prove that the problem of generating the optimal coalition structure has a complexity of $O(|A|^{|A|})$ and $\omega(|A|^{|A|/2})$. They show that there is a minimal number of structures that should be searched in order to establish a bound - at least $2^{|A|-1}$ structures must be searched to gather the valuations of all $2^{|A|}$ possible coalitions in a multiagent system with $|A|$ agents. They present an anytime algorithm that establishes a tight bound within this minimal amount of search. If we have the valuations for all possible combinations of agents in a system, the dynamic programming algorithm for the set partitioning problem presented by (Yeh, 1986) (and later

used for the combinatorial auction problem in (Rothkopf et al., 1998)), which runs in $O(3^{|A|})$ and $\Omega(2^{|A|})$ time, can also be used. Also, (Dang & Jennings, 2004) presents a slightly tighter bound on the anytime algorithm for computing the optimal coalition structure than the one discussed in Sandholm's paper.

(Kraus, Shehory, & Taase, 2003) proposes a coalition formation mechanism for a business-to-business domain with assumptions appropriate for real-world settings, such as incomplete information (i.e. the agents only have a partial view of their environment), multiple valuations of specific tasks (i.e. the agents differ in their valuation of various coalitions), and time-bounded processes. They adopt an auctions-based approach for allocating tasks and suggest some heuristics that the self-interested agents can adopt to form coalitions. They address only the coalition structure generation problem in this paper. (Kraus, Shehory, & Taase, 2004) extends their previous work and presents a variety of strategies for payoff distribution, including the strategy (using Stearn's transfer scheme) where the resultant distribution lies in the kernel.

Finally, (Rahwan & Jennings, 2005, 2007) address the coalition value determination problem in a distributed fashion for cooperative multiagent systems. Specifically, they present an algorithm that divides the $2^{|A|}$ equally among all $|A|$ agents, thereby significantly reducing the computational load for the first step in coalition formation.

4.2.3 Contracting

An agent may try to contract out some of the tasks that it cannot perform by itself, or that may be performed more efficiently by other agents. One self-interested agent may convince another self-interested agent to help it with its task by promises of rewards. The main question in such a setting is how one agent can convince another agent to do something for it when the agents do not share a global task and the agents are self-interested (Kraus, 1996). Furthermore, if the contractor-agent can choose different levels of effort when carrying out the task, how can the manager-

agent convince the contractor-agent to carry out the task with the level of effort that the manager prefers without requiring the manager's supervision?

The issue of incentive contracting has been investigated in economics and game theory for decades (Kreps, 1990; Mas-Colell, Whinston, & Green, 1995). There are usually two parties in all the different types of contracts considered for various applications. The first party (called *the agent* in economics literature) must choose an action or a level of effort from a number of possibilities, thereby affecting the outcome of both parties. The second party (called *the principal*) has the additional function of prescribing payoff rules. Before the agent chooses the action, the principal determines a rule of the principal's observations. Many models of the principal-agent paradigm differing in several aspects, such as the amount of information that is available to the parties, the observations that are made by the principal, and the number of agents involved, are studied in the literature.

A well-known framework for automating contracting is the contract net protocol (Smith, 1980; Smith & Davis, 1981; Davis & Smith, 1983). In the Contract Net protocol a contract is an explicit agreement between an agent that generates a task (the manager) and an agent that is willing to execute the task (the contractor). The manager is responsible for monitoring the execution of a task and processing the results of its execution, whereas the contractor is responsible for the actual execution of the task. The protocol works as follows. The manager of the task announces the task's existence to other agents. Available agents (potential contractors) then evaluate the task announcements made by several managers and submit bids for the tasks they are suited to perform. The manager then evaluates all the bids, selects the best bid and awards the contract to the bidder, and rejects the other bids. Thus, the task distribution is performed by a decentralized negotiation process involving a mutual selection by both managers and contractors. The original contract net protocol was developed for cooperative environments where all agents worked towards a common

goal, so there was no need to motivate the agents to bid for tasks or to do their best in executing them if their bids are chosen. The main problems addressed by the contract net protocol are task decomposition, subtask distribution, and synthesis of the overall solution (Smith, 1980; Smith & Davis, 1981; Davis & Smith, 1983).

A modified version of the contract net protocol for competitive agents is presented in (Sandholm, 1993). It provides a formalization of the bidding and the decision awarding processes, based on the marginal cost calculations according to local agent criteria. A contractee agent will submit a bid for a set of tasks only if the maximum price mentioned in the tasks' announcement is greater than the agent's total cost of handling both, its original tasks and these new tasks. The contractor agent, on the other hand, would only be interested in bids where it does not lose more than it saves by not handling the contracted tasks itself. A simple technique is suggested to ensure individual rationality for all agents - the actual price that the contractor pays to the contractee is half way between their marginal costs. (Sandholm, 1997, 1998; Andersson & Sandholm, 1998) provide asynchronous distributed implementations and theoretical results for the contract net protocol based on marginal costs.

(Sandholm & Lesser, 2002; Sandholm & Zhou, 2002) describes a method called *leveled commitment contract* where each party can unilaterally decommit to a contract by paying a predetermined penalty. They show that such contracts improve expected social welfare even when the agents decommit strategically in Nash equilibrium.

(Aknine, Pinson, & Shakun, 2004) also addresses agent decommitment in contract net protocol by introducing tentative contracts. It extends the contract net protocol by introducing two more steps - pre-bid and pre-assignment. This allows the agents to make tentative commitments that they can make binding at a later time, when they are more certain that no new offers better than the one currently committed to will arrive from other agents.

4.2.4 Market-Oriented Programming

Market-oriented programming is an approach to distributed computation based on market price mechanisms (Wellman, 1993; Wellman & Wurman, 1998). The idea of market-oriented programming is to exploit the institution of markets and models of them, and to build computational economies to solve particular problems of distributed resource allocation. In market-oriented programming, the distributed computational environment in a multiagent system is implemented directly as a market price system where the agents behave as computational modules in markets and interact in a very restricted manner - by offering to buy or sell quantities of commodities at fixed unit prices. When this system reaches equilibrium, the computational market has indeed computed the allocation of resources throughout the system, and dictates the activities and consumptions of the various modules. Note that this approach for multiagent coordination is applicable only when there are several units of each kind of item and the number of agents is large. Otherwise, it is not rational for the agents to ignore the effects of their behavior on the prices, when it actually has an influence. Also, there are situations in which reaching an equilibrium may be time consuming, and the system may not even converge (Wellman & Wurman, 1998). It also requires some scalable, trustworthy mechanism to manage the auctions for each type of item being traded. A survey and general discussion on the market-oriented programming approach can be found in (Wellman & Wurman, 1998; Wellman, 1996).

Part III

Research Contributions of this Dissertation

Chapter 5

Negotiation Networks

5.1 Our Agent-Based Negotiation Model

Target application domains for the work in this dissertation, such as the three examples presented in section 1.1, can be represented as multiagent systems and share the following properties:

1. The individual agents in the system are autonomous, rational, and self-interested. These agents might be owned by different organizations, and therefore, they make independent decisions and attempt to maximize the expected utilities for their owners *within* the confines of the rules laid down for admissible agent behavior by the governing bodies (also called the **mechanism**).

2. The agents encounter situations where they cannot satisfy their goal(s) by themselves - they have to cooperate (via negotiations) with other agents and arrive at mutually beneficial agreements in order to further their own selfish interests.

3. The agents have to cooperate in situations where there is a conflict of interest - multiagent interactions in the system can be modeled as constant-sum games, where an agent's gain is always at the expense of others in the system.

4. There can be an arbitrarily large number of agents in the system. However,

each agent can/must interact with only a small subset of other agents in the system.

5. The agent environment is inherently distributed - the information, capabilities, and resources are dispersed across all agents in the system, and it is not feasible to aggregate them at some central location where the coordination problem can be solved optimally.

6. The agents have an incomplete information about their environment. Their decisions are based purely on their local knowledge, and the information gathered by communicating with their neighbors.

Such problems are modeled as *negotiation networks* in this work.

5.1.1 Negotiation Network

Definition 34 (Negotiation Network) *A **negotiation network** is an undirected graph $G = \langle V, E \rangle$, with vertices $V = A \cup \Delta$, and edges $E = N \cup S$. Here, A is the set of agents in the environment, while set Δ represents the possible deals or coalitions that can exist between these agents. The negotiation edges N connect the potential deals to the agents that are involved in those deals, thereby representing the links over which negotiations must happen in the system. The social edges S connect each agent to all other agents that they can/must communicate (to make deals) with in the system, thus representing their social network.*

Figure 5.1 presents a graphical representation of a negotiation network. We formally define some of the important concepts and assumptions related to negotiation networks in the following subsections.

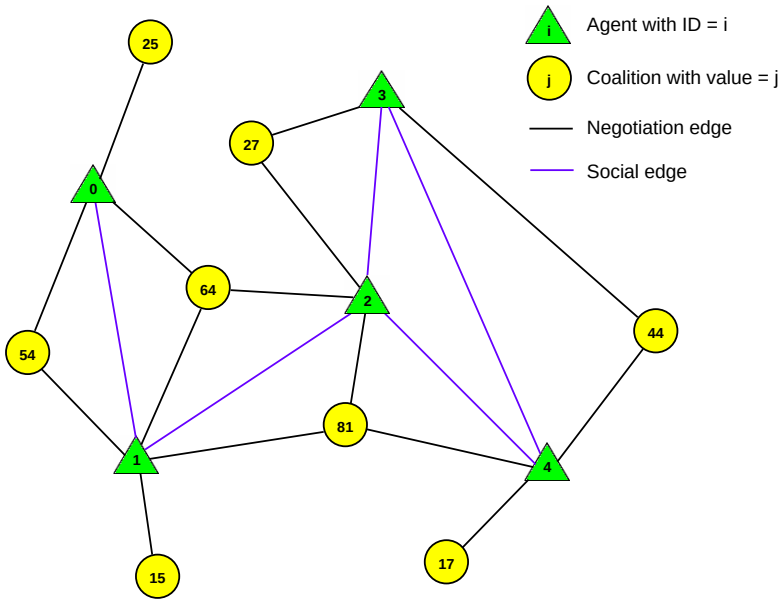

Figure 5.1: Negotiation Network

5.1.2 Deal (or Coalition) in a Negotiation Network

Definition 35 (Deal or Coalition) *A **deal** or **coalition** $\delta \in \Delta$ is a tuple $\langle \delta^a, \delta^r \rangle$, where $\delta^a \subseteq A$ is the set of agents participating in the coalition, and δ^r is the total revenue associated with the coalition (also called the coalition value) to be distributed among the agent members δ^a should the deal be a part of the negotiation outcome (a feasible coalition configuration).*

For a multiagent system comprising of $|A|$ agents, there are $2^{|A|} - 1$ possible ways in which the agents can form coalitions and agree on a deal in order to accomplish some task in the system. However, the total number of possible coalitions in our model are limited by the social networks of the individual agents.

5.1.3 Expectation (or Utility) of an Agent

Definition 36 (Expectation or Utility) *An agent $a \in A$ has an **expectation** or **utility** from each deal that it can potentially form in the system, given by the utility*

function.

$$u_a : \delta \rightarrow \Re$$

The agents in our system concurrently negotiate over all deals that they can potentially form in order to determine the one that maximizes their own utilities. Individual agent decisions lead to the selection of one of many feasible coalition configurations.

5.1.4 Deal (or Coalition) Configuration

Definition 37 (Deal or Coalition Configuration) *A **deal configuration** or **coalition configuration** is a tuple given by* $\vec{\Delta} = \left\langle \vec{\delta}, \vec{u} \right\rangle$*, where:*

- *The **coalition structure*** $\vec{\delta}$ *is a set of disjoint and exhaustive coalitions such that every agent in A participates in exactly one coalition in the set.*[1] *Formally,*

$$\vec{\delta} = \Gamma \subseteq \Delta$$

such that

$$\forall_{\gamma_1, \gamma_2 \in \Gamma} \quad \gamma_1^a \cap \gamma_2^a = \Phi$$

and

$$\forall_{i \in A} \exists \gamma \in \Gamma | i \in \gamma^a$$

- *The **payoff distribution vector*** $\vec{u}$ *represents the utility earned by each agent in the system according to the current configuration.*

$$\vec{u} \rightarrow \Re^{|A|}$$

---

[1]The coalition structure may include singleton coalitions involving solitary agents, where the utility of the agent is greater than or equal to zero.

Feasible Coalition Configuration

Definition 38 (Feasible Coalition Configuration) *A coalition configuration is* ***feasible*** *if, for all coalitions in the configuration, the total revenue associated with each coalition meets the expected utilities of all agents in the coalition. Formally, a coalition configuration is feasible iff*

$$\forall_{\delta \in \vec{\delta}} \sum_{a \in \delta^a; u_a(\delta) \in \vec{u}} u_a(\delta) \leq \delta^r$$

.

5.2 Modeling Automated Negotiation Problem as a Negotiation Network

Let us revisit the automated negotiation problem, using the following B2B scenario as a running example. There are many businesses in an electronic marketplace, each specializing in offering certain services for a price. Obviously, their sole concern is maximizing their profits by quoting the highest possible price for the services they offer for the tasks they commit to. These businesses may not be able to complete a task alone (because they do not possess all the service capabilities needed to handle the task fully). In such cases, they have to collaborate with each other to fulfill customer requests in the marketplace. A customer request is represented as a workflow specifying a list of services that need to be performed, and a reward that the customer is willing to offer to the business(es) that handles its request satisfactorily. So, for example, a customer request for online book purchase would include a workflow specifying that a seller must ship the book to the customer, and the customer's quote must be transferred to the seller by a payment service provider upon successful completion of the transaction. Thus, businesses offering the selling, shipping, and payment

services must work together to perform the customer request, and agree to share the customer's quoted reward among themselves such that they are all satisfied with the transaction. Now, there are multiple customer requests, and multiple businesses that can offer these services in the marketplace. Therefore, while the businesses are interested in gaining maximum revenue, they are also competing with other businesses. They have to outbid other similar businesses to win the contract, but lowering the bid too much might unnecessarily reduce their share of the reward from fulfilling the contract. Assuming that the businesses can only handle one customer request at a time, which request should they choose to serve? How do they determine their quote for the preferred request? These are hard questions, and are frequently encountered by businesses in the real world. The only way to solve this problem is to allow all agents to negotiate with each other over all customer requests that they are interested in.

A negotiation network captures all the above mentioned characteristics of the automated negotiation problem. Agents in a negotiation network are connected with social edges, which allow them to interact with each other and submit joint proposals for handling various customer requests. These proposals are deals or coalitions in the negotiation network. Each agent can thus establish contact with every other agent in its social network. The social network can be fully connected, as would be the case in a small, close-knit community, or sparsely connected, as in a marketplace. The model can be further extended to allow dynamic changes to the social network of agents, thereby representing changes in a dynamic environment such as a marketplace. Also, once a group of agents agree to submit a joint proposal, they would want to negotiate over how to distribute the proceeds of the deal between themselves, should it materialize. The negotiation edges in the negotiation network facilitate such interactions between the coalition members.

5.3 Agent Coordination through Coalition Formation

Agents in negotiation networks have to work together to further their own cause of making profit. The utility of an agent depends on the coalition that it commits to and on its coalition members, but not on any other coalition in the network. Also, a negotiation network can be studied as a characteristic form game. All these factors make coalition formation as the most appropriate coordination model for the automated negotiation problem. As discussed in section 4.2.2, coalition formation entails determining the coalition values, generating the optimal coalition structure, and computing a stable payoff configuration. The worth of various agent sets in automated negotiation problems represent the values of those coalitions. A feasible coalition configuration in a negotiation network constitutes a coalition structure, and a payoff configuration. The challenge is to determine a feasible coalition configuration with the optimal coalition structure, and a stable payoff configuration (according to any game-theoretic solution concept for stability). The research focus in this dissertation is on devising a negotiation mechanism that not only converges to such a solution, but also has the desiderata described in section 1.2.2.

Chapter 6

Approximation Algorithm for the Automated Negotiation Problem

As discussed in the previous chapters, the automated negotiation problem introduced in section 1.2.1 is an important problem not only in computer science, but also in many other fields such as social sciences and economics. We briefly discussed our approach for solving the problem in the previous chapter. In this chapter, we present a detailed theoretical analysis of the problem and determine the computational complexity of solving the problem optimally. Then, we present our automated negotiation mechanism, and discuss the PACT (Progressive Anytime Convergent Time-efficient) algorithm for quickly computing an approximate solution to this hard problem. We evaluate our algorithm empirically and present the simulation results. These results help us understand the various properties of our mechanism. Later, we apply our mechanism to a variant of the automated negotiation problem, where the worth of all agent coalitions is not known *a priori*. Again, we measure the performance of our approach empirically and present the simulation results for these experiments.

6.1 Automated Negotiation Problem Revisited

The automated negotiation problem from section 1.2.1 can be restated as a negotiation network problem in the following manner:

Definition 39 (Automated Negotiation Problem) *Consider a negotiation network, $G = \langle V, E \rangle$, with vertices $V = A \cup \Delta$, and edges $E = N \cup S$. The set of selfish, rational agents is represented as $A = \{a_1, \ldots, a_n\}$, and the set of coalitions in the network is $\Delta = \{\delta_1, \ldots, \delta_m\}$. Each coalition $\delta \in \Delta$ is a tuple $\langle \delta^a, \delta^r \rangle$, where $\delta^a \subseteq A$ is the set of agents participating in the coalition, and δ^r is the highest worth of that set of agents working together for some task. The negotiation edges N connect each δ to δ^a, while the social edges S connect each agent to every other agent. Given the above network, devise a negotiation mechanism that computes a feasible coalition configuration with optimal coalition structure and stable payoff distribution.*

6.2 Theoretical Analysis of the Automated Negotiation Problem

We model the automated negotiation problem as a negotiation network. Therefore, solving the automated negotiation problem entails computing a feasible coalition configuration with desired properties for its negotiation network. As mentioned in section 5.3, we use coalition formation as the coordination model for our agents in the negotiation network. This requires us to solve three subproblems: determining the coalition values, generating the optimal coalition structure, and computing a stable payoff configuration for the negotiation network. We address each of these subproblems separately in the following subsections.

6.2.1 Determining the Coalition Values

The first step towards agents forming favorable coalitions is determining the worth of each candidate coalition for them. For N agents, there are 2^n possible coalitions, where $n = |N|$. Therefore, the complexity of determining the coalition values (building the characteristic function) is $O(2^n)$. However, the worth of all coalitions is common knowledge in the automated negotiation problem, as defined in this work. Therefore, this step can be completed in constant time.

6.2.2 Generating the Optimal Coalition Structure

A coalition structure is a partition of all agents into groups such that each group is a coalition in the negotiation network. The problem of computing the total number of such coalition structures is studied in mathematics (combinatorics), and is given by

$$\sum_{k=1}^{n} S(n,k),$$

where,

$$S(n,k) = \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n$$

is the Stirling number of the second kind (Goldberg, Newman, & Haynsworth, 1972), which gives the total number of ways of partitioning a set of n elements into k nonempty sets. As can be seen from the above formulae, the number of coalition structures grows rapidly (considerably faster than the number of coalitions grows) as the number of agents increases. It is of the order of $O(n^n)$ and $\omega(n^{n/2})$. Table shows the growth rate in numbers. Therefore, exhaustive enumeration is not a viable solution for finding the optimal coalition structure, unless the number of agents is very small (under 15 or so in practice).

Number of Agents	Total number of Coalition Structures
3	5
4	15
5	52
6	203
7	877
8	4140
9	21147
10	115975

Table 6.1: Growth rate of the number of coalition structures with increasing number of agents

6.2.3 Computing a Stable Payoff Configuration

Once we have derived the optimal coalition structure for a negotiation network, we need to find a payoff distribution that is stable (as defined by the chosen game-theoretic concept). An imputation defines a way to divide the total revenue of the selected coalition structure among all agents in the network. The size of this imputation set can give us a reasonable estimate of the complexity of finding a stable solution, as stable payoff distributions are only a tiny subset of the imputation set. This is another combinatorial problem well-studied in mathematics.

The size of the imputation set depends on many factors:

Number of agents: Clearly, with increasing number of agents, there is an exponential growth in the number of ways that the revenue can be split between the agents. For example, consider a system where $n = 2$, and $v(N) = 2$. Here, the imputation set is $\{(2,0),(1,1),(0,2)\}$. Now, if the number of agents in the system changes to 3, the total number of ways to distribute $v(N)$ among the agents increase to $\{(2,0,0),(1,1,0),(1,0,1),(0,1,1),(0,2,0),(0,0,2)\}$. See table 6.2 for more examples about this relationship.

Grand coalition value: The value of the grand coalition, $v(N)$, in the characteristic function of the negotiation network influences the size of the imputation set.

x_i / n	8	7	6	5	4	3	2	1	0	Imputation set size (I)
2	1	1	1	1	1	1	1	1	1	9
3	1	2	3	4	5	6	7	8	9	45
4	1	3	6	10	15	21	28	36	45	165
5	1	4	10	20	35	56	84	120	165	495
6	1	5	15	35	70	126	210	330	495	1287
7	1	6	21	56	126	252	462	792	1287	3003
8	1	7	28	84	210	462	924	1716	3003	6435
9	1	8	36	120	330	792	1716	3432	6435	12870
10	1	9	45	165	495	1287	3003	6435	12870	24310

Table 6.2: Growth rate of the imputation set. We set $v(N) = 8$. The columns show the total number of imputations where the payoff of an agent, $i \in N$, is $x_i = 8, 7, \ldots, 0$ with different agent set sizes $n = 2, \ldots, 10$. The final column represents the total size of the imputation set for each row.

Consider the above example again - we have $n = 2, v(N) = 2$, and the imputation set is $\{(2,0),(1,1),(0,2)\}$. Upon increasing $v(N)$ from 2 to 3, the imputation set increases to $\{(3,0),(2,1),(1,2),(0,3)\}$. Table 6.2 provides the imputation values for $v(N) = 8$. To see the imputation values for $v(N) = \gamma, 0 < \gamma < 8$, move $8 - \gamma$ headings of imputation columns (columns 2 - 10 in table 6.2) to the left, and delete $8 - \gamma$ imputation columns from the right.

Desired precision level: The desired precision level in the outcome of the algorithm has a significant impact on the size of the imputation set. A precision level requirement of upto z decimal points is equivalent to an increase of $v(N)$ by a factor of 10^z. For example, a game with $n = 2, v(N) = 2$ and a desired precision level of 2 decimal points will comprise of the following imputation set: $\{(2.00, 0.00), (1.99, 0.01), (1.98, 0.02), \ldots, (0.00, 2.00)\}$.

Note that table 6.2 is the symmetric Pascal matrix. Each element in this matrix, I_{nm}, represents the total number of imputations in the imputation set of a negotiation network with n agents, where an agent $i \in N$ gets value $m = x_i$ out of the total available revenue, $\gamma = 10^z \cdot v(N)$. This value is given by,

$$I_{nm} = \binom{a}{b}$$

where,

$$a = n - 2 + \gamma - m \quad \text{and} \quad b = n - 2$$

Therefore, the total size I of the imputation set in a negotiation network is given by

$$I = \sum_{m=0}^{\gamma} I_{nm} = \sum_{m=0}^{\gamma} \frac{(n-2+\gamma-m)!}{(n-2)!(\gamma-m)!}$$

Thus, the sample space of candidate imputations while searching for the stable imputation is huge, and a naive algorithm that scans through the entire search space to derive a stable imputation will have $O(\frac{(n+\gamma)!}{n!\ \gamma!})$ complexity.

6.3 PACT - Our Negotiation Mechanism to Solve the Automated Negotiation Problem

As described in the previous section, the automated negotiation problem is hyper-exponentially complex. Therefore, the best that we can do is present some heuristics that yield results close to the optimal solution. In this section, we present our negotiation mechanism for addressing this problem. First, we describe the basic principles that were adopted to devise the mechanism. The following sections describe the negotiation protocol and the negotiation strategy of the mechanism. Then we present the algorithm that puts all these ideas together, along with a running example to demonstrate it.

6.3.1 Basic Principles

We adopt the following principles to circumvent the enormous complexity of the problem at hand:

1. The best way to ensure that the agents are satisfied with the results of any mechanism is to involve them in the decision process. We do that by *distributing* the computational load among all agents in the network. In our mechanism, each agent determines the preferable coalitions for themselves, and negotiates with other members of these coalitions to reach a consensus over the payoff distribution. This way, the computational load is shared by all agents in proportion to their relative bargaining strengths.

2. We allow agents to negotiate with other agents over all their candidate coalitions *concurrently*. This allows the agents to assess their relative bargaining strengths, and thereby, make a sound decision at each stage.

3. The agents follow an *iterative* algorithm for negotiations. Having multiple rounds of negotiation allows the agents to rectify previous bad decisions, if any, and ensures that the solution that they end up with when the negotiations converge will be acceptable to them all.

4. Our mechanism simultaneously handles the constraints for optimal coalition structure generation and stable payoff configuration determination problems, thereby drastically *limiting* the sample space considered while searching for the solution. Searching for stable solutions facilitates the agents to identify the coalitions they prefer, and we consider only those coalition structures that include preferred coalitions.

6.3.2 Negotiation Protocol

The PACT negotiation protocol is a two-step process - the first step involves iterative, distributed negotiations over payoff divisions among the agents in the negotiation network, and the second step selects the near-optimal coalition structure based on the payoff allocations agreed upon in the previous step.

Figures 6.2 and 6.3 describe the negotiation protocol in detail.

Input: A negotiation network, with coalition values

Output: A feasible coalition configuration, with optimal coalition structure and stable payoff configuration

Protocol: The PACT negotiation protocol has two steps –

Step 1 – Iterative, distributed negotiations over payoff divisions

Step 2 – Selection of the (near-)optimal coalition structure based on the payoffs decided in step 1

Figure 6.1: PACT Negotiation Protocol

6.3.3 Negotiation Strategy

Given the above negotiation protocol, we now present a negotiation strategy for the agents in figure 6.4.

6.3.4 The Algorithm

We now present our approximation algorithm for the automated negotiation problem based on the ideas discussed above (Goradia & Vidal, 2007b, 2007c).

The Progressive Anytime Convergent Time-efficient (PACT) algorithm is a distributed, hill-climbing algorithm that facilitates selfish, rational agents to form coalitions in order to handle tasks (customer requests in our B2B example from section 5.2). It is an iterative algorithm that progressively produces a better solution over

- For the first round, each agent claims equal share of the total value for each of its coalitions
- For each subsequent round, each agent has three options for each of its coalitions
  - Claim a new share of the coalition value
  - Vote to fix the negotiations over the coalition
  - Opt out of the negotiations for the coalition
- If any agent in a coalition claims a new share, then
  - Compute the surplus (or deficit) for the coalition
  - Equally share surplus (or deficit) among the agent members to derive new expectations from the coalition for the current round
- If all agents vote to fix a coalition, then fix it. No negotiations happen over the coalition in future rounds
- If any agent opts out, then remove the coalition from the system
- Repeat the negotiations for another round until all coalitions are fixed or opted out of

Figure 6.2: Step 1 of the PACT Negotiation Protocol – Iterative, Distributed Negotiations over Payoff Divisions

- Repeat until there are coalitions in the system
  - Each agent broadcasts its preferred coalition
  - The best of the preferred coalitions is cleared
    - Coalitions with higher values, fewer members, and earlier arrival in the system have a higher preference over others
    - All agents participating in the best coalition are cleared
  - All other coalitions with members from the cleared coalition are removed from the system
- The set of cleared coalitions form the coalition structure chosen by the PACT mechanism for the given CFG. The payoffs within the coalitions is in accordance with the negotiations in step 1.

Figure 6.3: Step 2 of the PACT Negotiation Protocol – Near-Optimal Coalition Structure Selection

Step 1 - Agent decision function for each negotiation round

- For each of its coalitions, an agent decides on its action for the current round as follows:
  - It **opts out** of the negotiations for the coalition if its derived expectations from the previous round is below its reservation price
  - It computes its maximum expectation from alternative coalitions that it can participate in
  - If the maximum expected value is identical to that from the previous round, then it **votes to fix** the coalition, else it **claims the new value** for the coalition in the current negotiation round

Step 2 – Agents choice of preferred coalition

- An agent always chooses the coalition that yields it the highest payoff

Figure 6.4: PACT Negotiation Strategy

time. It is also an any-time algorithm, so it can be halted at any time before its normal termination to provide the best solution until that time. We will show in our experimental evaluation of the algorithm that it always converges and is very efficient too. It is based on the Equal Excess Theory (EET) (see section 2.3.8 for details) for coalition formation, which is a bargaining solution concept in game theory to realize stable solutions for any characteristic function game.

We point the readers to one significant issue that we discovered about EET. We quote from (Kahan & Rapoport, 1984), page 153:

> For all games examined to-date, the sequence [of payoff allocations for every coalition] converges to an asymptotic value, which often corresponds to the solution of other theories, but a general proof has not yet appeared.

Our simulations show that for many games, the player expectations do not converge. A change in a player's expectation for a coalition changes the negotiating power of its neighbors in the coalition, which in turn leads to a potential change (if it is the

highest expectation) in their expectations for all coalitions in which they participate. In some games, this results in a change in the expectations of one or more neighbors of the original player for some other coalition(s), which forces the player to switch back to its original expectation. This results in a cyclic equilibrium where the expectations of all the involved players toggle between two or more values. For example, consider this characteristic function game:

$$v(A) = 8;\ v(AB) = 15;\ v(AC) = 23;\ v(BC) = 13;\ v(N) = 0$$

All player expectations toggle between two values after round 10 of player negotiations.

PACT is a distributed implementation of the EET. PACT goes beyond the EET in that it enforces convergence, and it has a mechanism for the agents to select the winning coalitions in a distributed fashion. 6.5 describes our PACT algorithm.

The PACT algorithm 6.5 passes through the following phases:

Pre-negotiation phase (lines 1-6): All agents are initialized here. For the given coalition model, each agent determines its candidate coalitions (line 2) and neighbors (line 3). For each candidate coalition, the agent initially expects an equal division of its total value (line 5) and sets its negotiation status to *active* (line 6).

Negotiation phase (lines 7-17): Each agent concurrently participates in all its candidate coalitions. For each candidate coalition, the agent determines all the valid alternative coalitions that it can participate in, and sets its logical claim (expectation) from this coalition for the current round of negotiations as the maximum revenue it received from any of these alternative coalitions (line 9). The agent exchanges its demands with all its neighbors and then computes its actual payoffs for the various coalitions. Demands by all agents in a coalition may lead to some surplus (or deficit) in revenue. This excess is split

FIND-COALITION-PACT(i)

$t \leftarrow 0$ ▷ Pre-negotiation phase
$L_i \leftarrow \{c \in T : i \in c^{agents}\}$
$N^{agents} \leftarrow \bigcup_{c \in L_i} c^{agents}$
for $c \in L_i$
  do $E^t(i,c) \leftarrow \frac{c^{value}}{|c^{agents}|}$
    $fixed(c) \leftarrow FALSE$
$t \leftarrow t+1$ ▷ Negotiation phase
for $c \in L_i$
  do $A^t(i,c) \leftarrow \max_{c' \in L_i - c} E^t(i,c')$
    for $j \in c^{agents}$
      do send $A^t(i,c)$; receive $A^t(j,c)$
    if $fixed(c) = FALSE$
      then $E^t(i,c) \leftarrow A^t(i,c) + \frac{c^{value} - \sum_{j \in c^{agents}} A^t(j,c)}{|c^{agents}|}$
    if $(E^t(i,c)$`has converged`$) \vee (E^t(i,c) \leq 0)$
      then $fixed(c) \leftarrow TRUE$
if $\exists c \in L_i : fixed(c) = FALSE$
  then goto 7
$c_i^* \leftarrow$ `argmax`$_{c \in L_i} E^t(i,c)$ ▷ Post-negotiation phase
for $j \in A$
  do send c_i^*; receive c_j^*; put all into $C^*_{preferred}$ set
$c^* \leftarrow \left\{c_x \in C^*_{preferred} : \forall x \neq y, c_x^{value} > c_y^{value}\right\}$
if $i \in c^{*agents}$
  then return c^*
$C^*_{preferred} \leftarrow C^*_{preferred} - c^*$
for $j \in c^{*agents}$
  do $L_i \leftarrow L_i - \{c \in L_i : j \in c\}$
if `not empty?`L_i
  then goto 18
return

Figure 6.5: The FIND-COALITION-PACT algorithm to find the best task allocation for agent i.

equally between all the members and added to the agent's claim to get its actual payoff for the concerned coalition in the current negotiation round (line 13). If the agents payoff for a coalition remains steady (i.e. the coalition converges), then the agent sets the coalitions negotiation status to *terminated* and no more negotiations are conducted for this coalition (lines 14-15). An agent might also terminate negotiations for a coalition if its payoff goes negative (i.e. the agent has to pay to the coalition in order to participate in it). We have noticed in our simulations that sometimes the agent payoff does not converge. Instead, it keeps toggling between two values. However, even under such unstable conditions the total excess generated within the coalition converges. Our agents test this convergence to determine the termination of the negotiation process.

Post-negotiation phase (lines 18-29): This stage involves the selection of the most preferred coalitions by the agents. Each agent initially ranks all their coalitions based on their derived payoff from each of them and broadcasts its most preferred coalition (lines18-20). Thus, each agent is informed about every other agent's preferences over the choice of coalitions. Each agent sorts this list of preferred coalitions based on their coalition values. Ties are broken by ranking the coalitions that have fewer agents ahead of the larger ones, as this can potentially maximize social welfare. For further ties, the coalition that was formed earlier in the environment is preferred. Thus, every agent knows the winning coalition in the current round (line 21). The agent joins the winning coalition if it is a member, or removes all coalitions involving agents in the winning coalition and repeats the winner determination process if it is not (lines 22-26). An agent remains unallocated if it runs out of candidate coalitions to participate in.

6.3.5 Demonstration of the PACT Algorithm

Figure 6.6 shows a pictorial description of the PACT algorithm. We consider the following negotiation network here:

$$A = \{0, 1, 2\}, \quad \Delta = \{70, 80, 90, 105\}$$

$$\delta_1^a = \{0, 1\}, \delta_1^r = 90; \qquad \delta_2^a = \{0, 2\}, \delta_2^r = 80;$$
$$\delta_3^a = \{1, 2\}, \delta_3^r = 70; \quad \delta_4^a = \{0, 1, 2\}, \delta_4^r = 105.$$

Timestep $t = 0$ shows the status of the network at the end of the pre-negotiation stage. All coalitions are equally split among all their agent members. For the first round of negotiations, each agent computes, for each of its coalitions, its maximum expectation from an alternate coalition. For example, consider agent 0 for coalition 105. The alternative coalitions to coalition 105 for agent 0 are coalitions 90 and 80, and its current expectations from them are 45 and 40 respectively, with the maximum expectation being 45. Now, each agent claims its highest expectation from an alternate coalition for all their coalitions. So, for coalition 105, agents 0, 1, and 2 claim 45, 45, and 40 respectively. The resultant surplus (or deficit) is then shared equally by all coalition members. This yields expectations 36.67, 36.67, and 31.67 for agents 0, 1, and 2 from coalition 105 at the end of the first negotiation round, as shown in the figure with timestep $t = 1$. The procedure continues until the negotiations converge after 13 rounds, as shown in the figure with timestep $t = 13$. This is the end of the first stage of PACT negotiations

After the completion of the first stage, all agents have agreed upon the distribution of the payoffs from all their coalitions. Therefore, the agents are now in a position to select the best (most rewarding) coalition to participate in. Each agent ranks all its coalitions and votes for their selection according to their preference order. The coalitions chosen by all agents are selected for the coalition structure proposed by

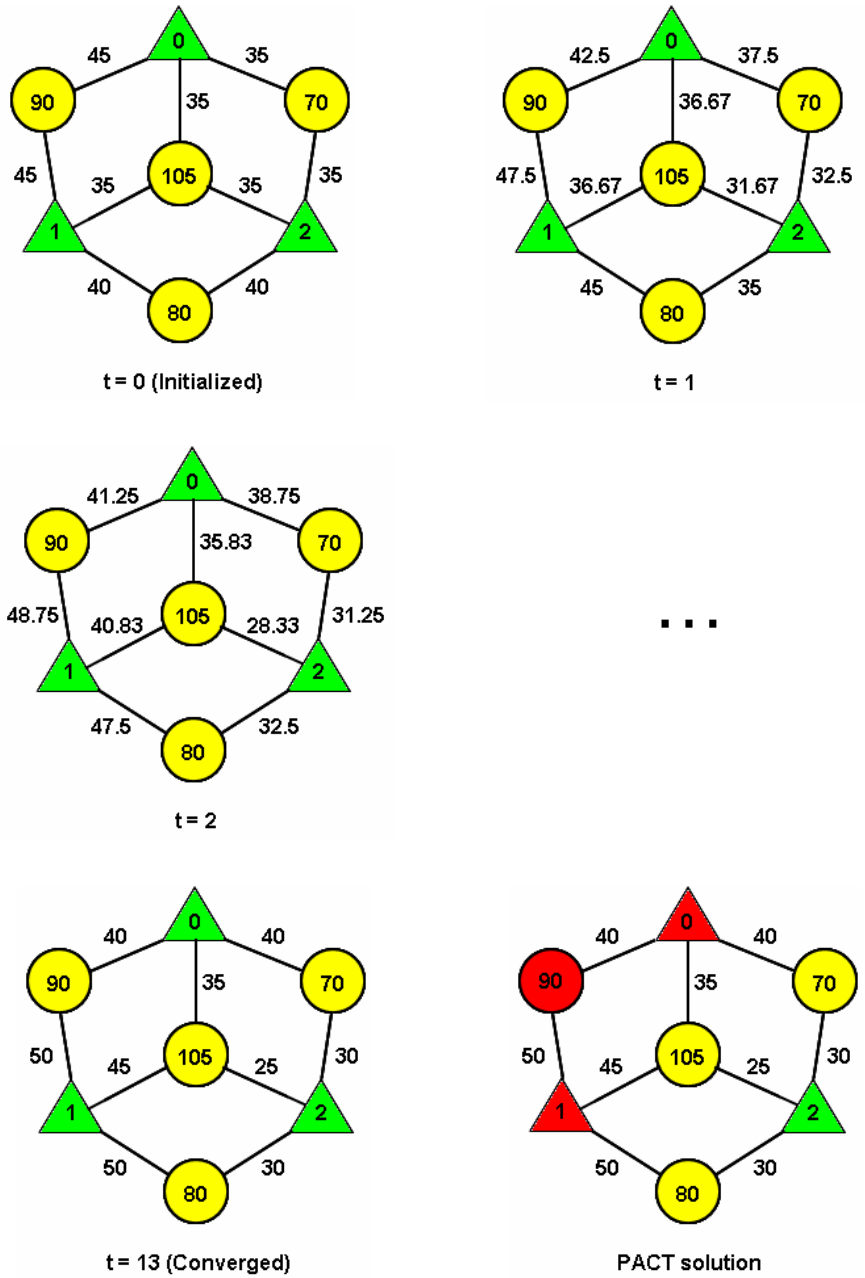

Figure 6.6: Demonstration of the PACT Algorithm

our mechanism. Once a coalition is selected, all other coalitions that its members participate in are removed from the system before the remaining agents make a new selection. This procedure eventually yields the coalition structure that our PACT mechanism proposes for the negotiation network. The payoff divisions for the selected coalitions are in accordance with the decisions made in the first step. The network marked PACT solution is the chosen solution for the above example.

6.4 Experimental Evaluation of PACT

We have developed a simulation system implementing the PACT algorithm and conducted a series of experiments to test the quality of the solutions that it leads to. First, we compare our algorithm's performance with that of a centralized algorithm that finds an optimal solution by performing an exhaustive search of the entire solution space. Specifically, we compare the coalition structures that the two algorithms propose in terms of the number of tasks that get allocated, and the total revenue generated as a result of that. We also present the results that show the computation time that our PACT algorithm requires to produce it's results. Our next set of experiments concern with testing the scalability of our algorithm. Scalability is tested with respect to the number of agents, the number of tasks, and the coalition size for handling the tasks. Finally, we also demonstrate the anytime properties of the PACT algorithm and show the rate of improvement in the quality of the solution produced with extended negotiations.

6.4.1 Experiments comparing PACT with an Optimal algorithm

Comparison of the generated Coalition Structure

The goal of the first experiment is to determine how far the results generated by PACT are from an optimal solution in terms of the number of tasks that are allocated from the given set. As mentioned before, PACT facilitates agent negotiations such that they can rank the tasks based on their expectations from each of them. The agents then greedily seek out the best tasks for themselves. With such an approach, there is a tendency for the agents to get stuck on some local maxima and this has an overall adverse effect on the system-wide performance. To determine this sub-optimality, we tested our system with a widely ranging distribution of agents and tasks, with scenarios where the agent/task ratio is 0.25, 0.5, 0.75, 1, 1.33, 2, and 4. Figure 6.7 shows our algorithm's performance as a ratio of that of the optimal solution.

Surprisingly, the simulation results show that the PACT algorithm produces solutions that are almost as good as those generated by the optimal solution. Apparently, the negotiation phase in PACT nudges the various agents towards coalitions that are mutually beneficial to all the concerned agents, and the payoff exchanges converge at an equilibrium point where it is in the best interest of all the participants to select those coalitions. Therefore, although all agents adopt a myopic view in selecting the allocations most favorable to them in the post-negotiation phase, all agents end up selecting the favorable coalitions.

Comparison of the Global Value of the generated Coalition Structure

Allocating almost the same number of tasks as an optimal solution is good, but the more important question is *which* tasks are allocated. If the agents end up selecting tasks that are relatively less worthy, then their contribution towards social welfare

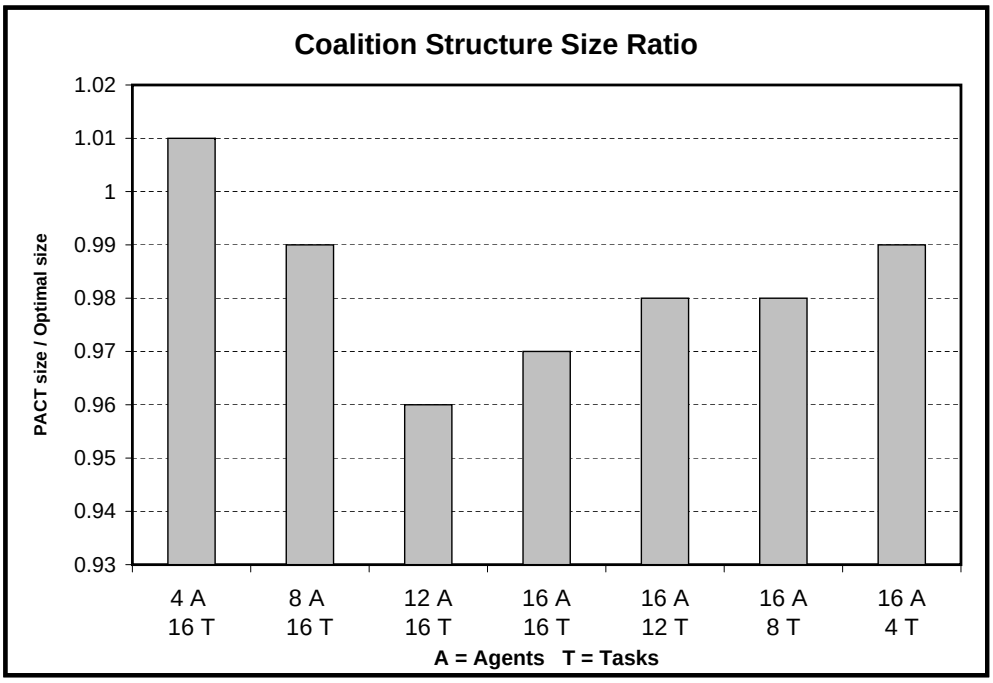

Figure 6.7: Comparison of the coalition structure size for PACT and optimal algorithms

maximization will be low. In this experiment we compare the total wealth generated by the PACT algorithm in comparison to the highest possible value. Figure 6.8 shows the results for the same set of agent/task distributions as in the previous experiment.

Simulation results show that the total value of the coalition structures generated by PACT is very close to those generated by the optimal solution. This suggests that the tasks that PACT allocates are mostly identical to those selected by the optimal solution. Again, the agent negotiations drive them towards coalitions that have relatively higher values, and the negotiation process keeps them interested in those coalitions while also seeking a stable payoff configuration.

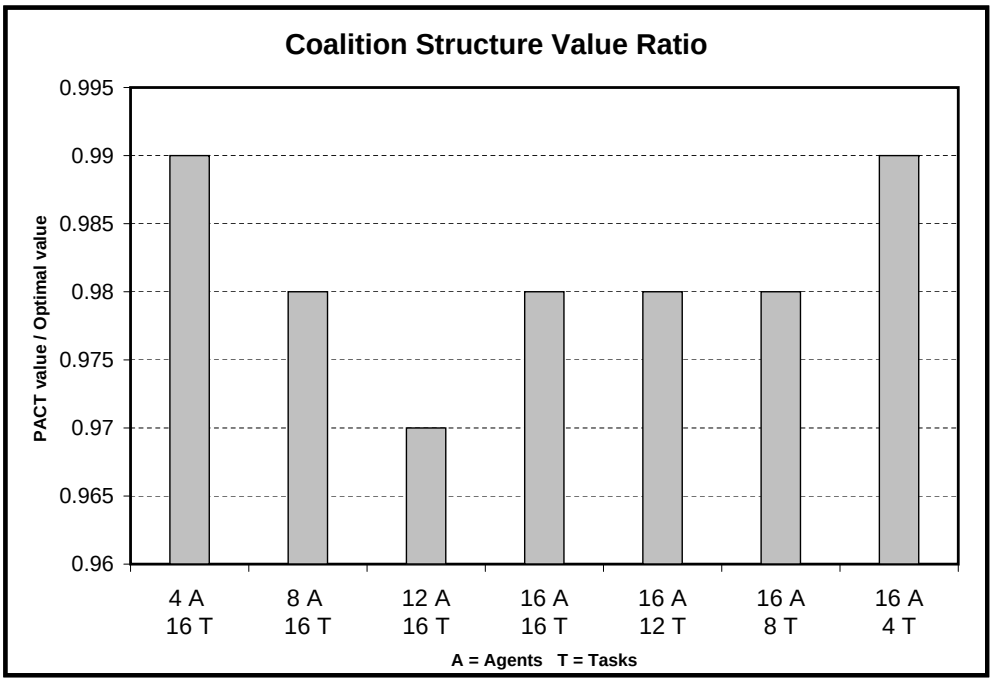

Figure 6.8: Comparison of the coalition structure value for PACT and optimal algorithms

Computation Time for PACT

The above results show that the PACT algorithm performs very creditably in comparison to the optimal algorithm. But it would all mean little if it also takes a long time like the optimal algorithm to compute those results. This set of experiments addresses that question - it determines the total computation time required by the PACT algorithm for the various agent/task distribution scenarios. Figure 6.9 presents the results.

Being a hill-climbing algorithm, PACT is very efficient. Also, being a distributed algorithm, PACT segregates the computational effort of deriving the equilibrium solution between all the participating agents. Therefore, PACT is incredibly efficient

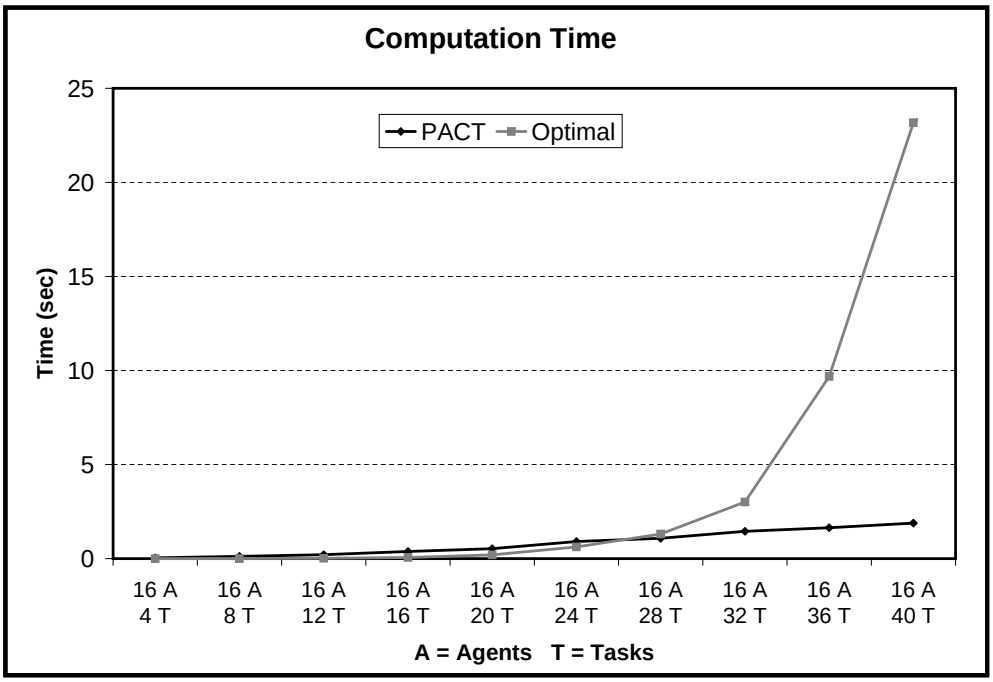

Figure 6.9: Comparison of the computational cost of PACT and optimal algorithms

for all the distribution scenarios.

Given that the PACT algorithm is very efficient, the next logical step is to determine how scalable it is. Our next set of experiments address that.

6.4.2 Experiments to test the Scalability of PACT algorithm

We test the scalability of the PACT algorithm in terms of the following: (1)the number of agents, (2)the number of tasks, and (3)the coalition size.

Scalability of PACT in terms of the Number of Agents

In this set of experiments, we test the average number of negotiation rounds that the PACT algorithm requires for various agent sizes, when the number of tasks and the

coalition size is fixed. For these experiments, we set the number of tasks to 50, and the coalition sizes for the tasks are chosen from a uniform distribution of random numbers ranging from 1 to 5. We vary the total number of agents in our system from 5 to 500, as this presents us with a rich variety of agent/task combinations to bring out any vagaries in the algorithm's performance. The results from our experiments are presented in figure .

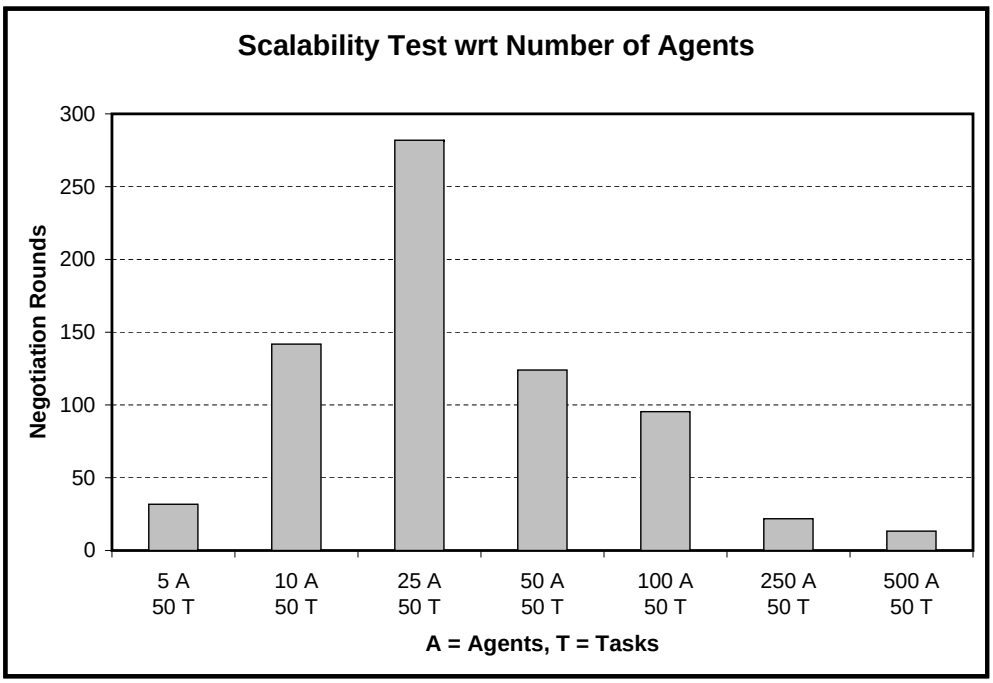

Figure 6.10: PACT Scalability with respect to Number of Agents

The results show that the negotiations extend for larger number of rounds when the agents are more competitive. Since we have 50 tasks with an average coalition size of 2.5, there will be approximately 125 edges in the network graph (see **??**) for our experimental setup. When there are very few agents in the system, each agent will be highly connected to all the tasks in the environment, and therefore will have

similar negotiating power in the bargaining process. Due to the lack of competition, these agents quickly converge to the task(s) that offer them maximum benefits. On the other hand, when we have too many agents in the system, then their connectivity to the tasks in the environment, and therefore their negotiating power, will be very limited. Again, such agents greedily accept their best offers and converge quickly. When there are just enough agents in the system such that they are all heavily connected, and they also compete with each other for the best tasks as not all of them can be supported in the most coveted ones, the negotiations prolong for a long period of time. This is what we see when there are about 25 agents in the system - each will approximately be connected to 5 tasks, and there will be multiple agents with strong negotiating powers competing for each task. PACT therefore takes a lot longer to converge in this scenario than in other cases.

Scalability of PACT in terms of the Number of Tasks

Here we test the convergence of PACT for various values of tasks while the number of agents and the coalition size is fixed. Here, we have 50 agents in our setup, and the coalition size is determined randomly from a uniform distribution of values ranging from 1 to 5. For these experiments, the number of tasks are varied from 5 to 500. Figure 6.4.2 presents the results from our tests in these experiments.

Just as the results for the previous set of experiments, over here also we see that when there are sufficient tasks in the environment for the agents to compete for, the PACT algorithm executes for large number of rounds. However, it is still within reasonable limits for being useful for practical purposes. On the other hand, when there are fewer tasks than agents to perform them, the algorithm converges quickly.

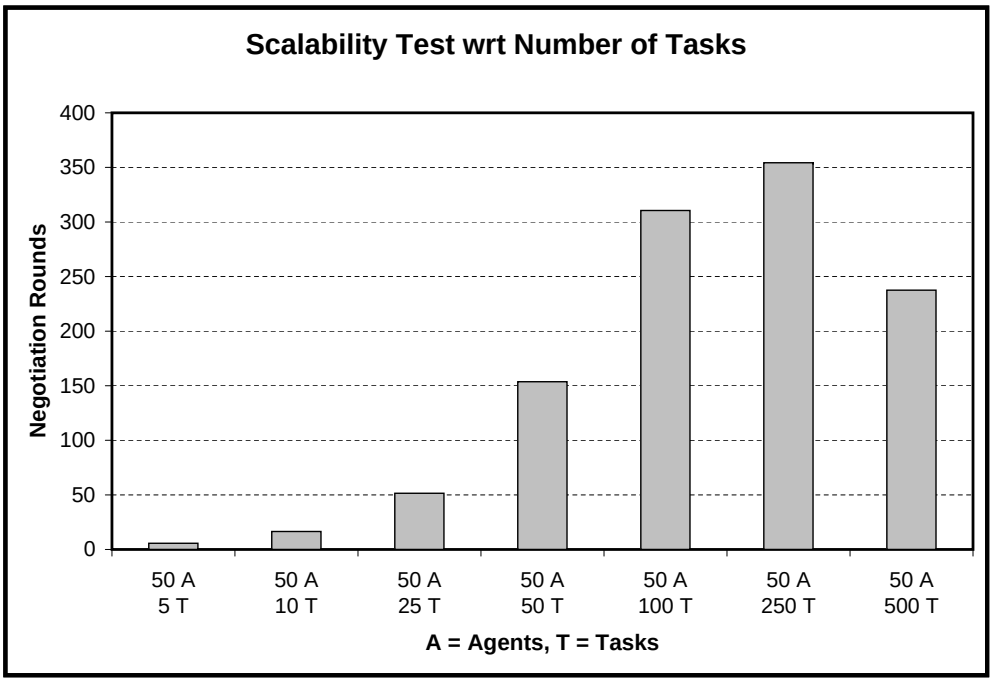

Figure 6.11: PACT Scalability with respect to Number of Tasks

Scalability of PACT in terms of Coalition Size

We had randomly generated all the coalitions of various sizes ranging from 1 to 5 for all our tests so far. The goal in this set of experiments is to test the algorithm performance in an environment where all the tasks have the same number of agents forming coalitions to perform them. Just as in previous experiments, we test for various agent/task distributions in this one too. Figure 6.12 presents our findings for these experiments.

There are no surprises in these results. As the coalition size grows larger, more agents have to endorse a particular coalition for it to be selected for allocation. It gets progressively harder to choose tasks for allocation with increasing coalition size. For each agent/task distribution, we see that the smallest coalitions converge fast,

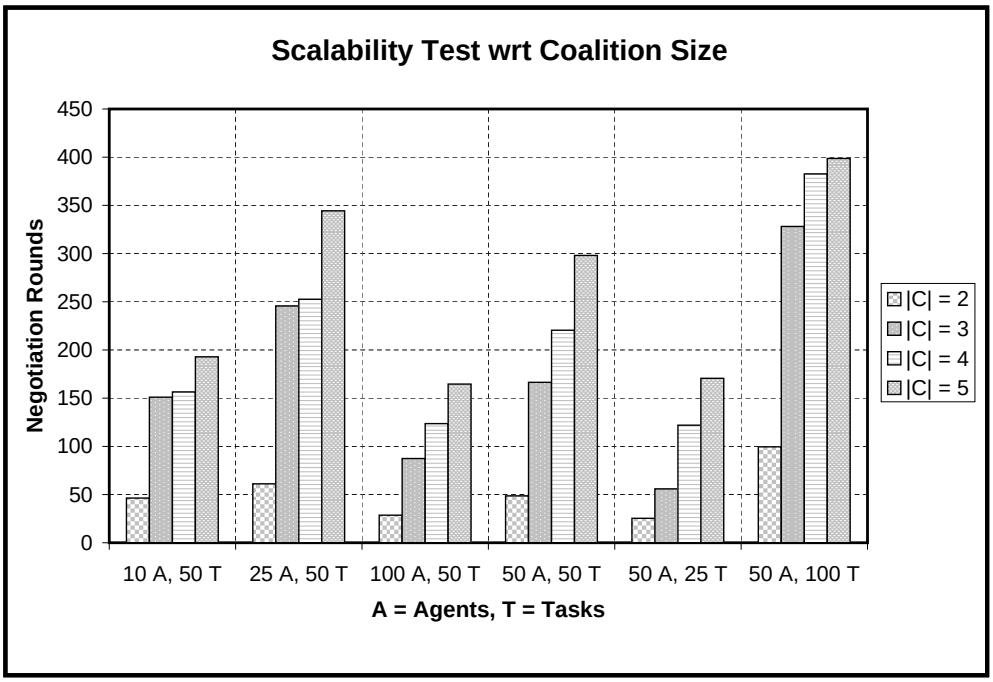

Figure 6.12: Scalability with respect to the Coalition Size

while the larger ones either don't converge, or do so after extensive negotiations only.

6.4.3 Convergence in PACT

As we saw in the previous results, PACT sometimes takes a very long time to converge, particularly in competitive environments. That begs the question - Is PACT really appropriate for real-life applications? The following set of experiments are geared towards addressing this question.

For an agent participating in PACT, each round of negotiations involves setting the expectation for each of it's coalitions, and then computing the actual payoff for that round by considering the expectations of all the other members of that coalition. When the solution converges, the difference between the expectations and the actual

payoffs diminishes. Therefore, the total change (sum of the absolute changes) in the expectations of all the agents for all its coalitions in the system is a good metric for testing convergence in PACT. In this set of experiments, we determine how long PACT takes to converge up to certain percentages of its best solution by measuring the total expectation changes for the negotiation rounds. Figure 6.13 shows the results of our tests.

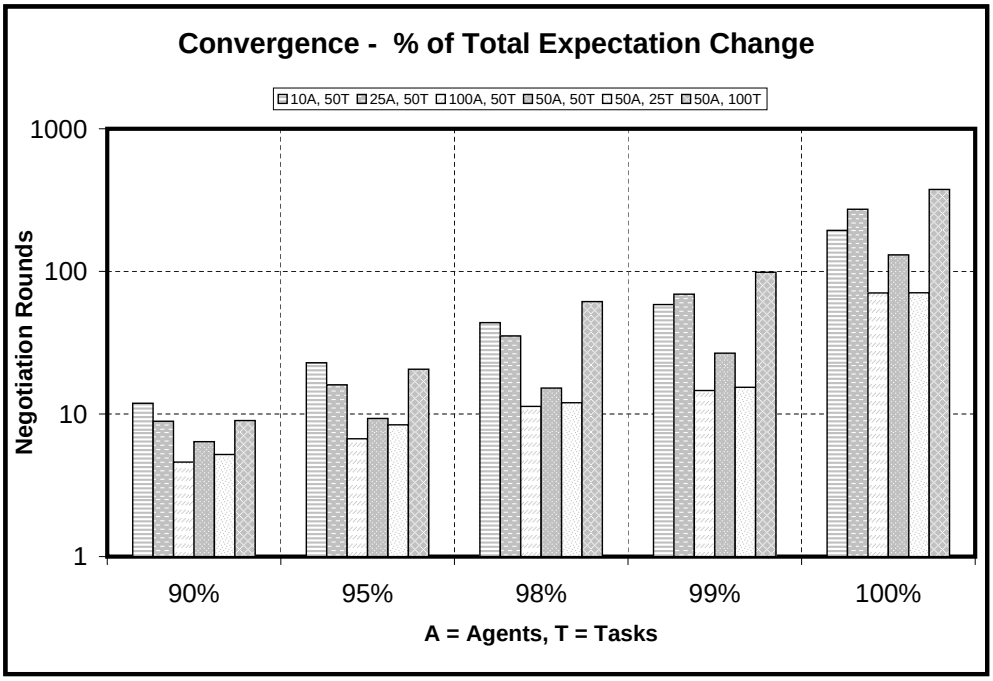

Figure 6.13: Convergence results in PACT

The most significant observation in these results is that PACT converges up to 90% of its best solution within the first 10 rounds of negotiations. This is true even for the most competitive environments. The quality of the solution keeps improving as we proceed with the negotiations, with a solution 95%, 98%, and 99% close to its best arrived by approximately the $20^{th}, 30^{th}, and 50^{th}$ rounds respectively. It is

only the final 1% that holds back the vast majority of the PACT runs from finishing earlier. This feature in PACT whereby it converges to a close approximation of its best solution very early in the negotiations is critical for its application in various real-world domains.

6.4.4 Core Stability Test for PACT Solutions

Our final set of experiments test the PACT solutions in terms of our relaxed-core stability concept. We test various agent/task distributions and coalitions with maximum size of 3 and 5 to determine the percentage of PACT solutions that lie in the relaxed-core. Figure 6.14 shows the results for these experiments.

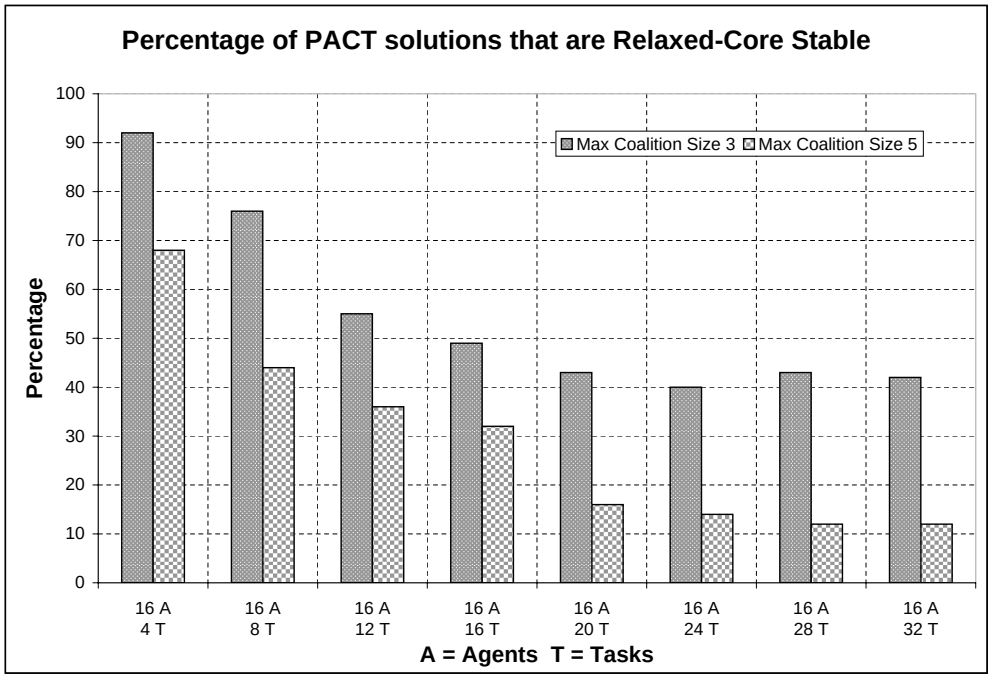

Figure 6.14: PACT solutions that are Core Stable

Note that the core can also be empty for a particular game. The likelihood for the

core being empty increases with higher values for the number of tasks, agents, and coalition sizes as the heavy constraints that the solution concept demands from a game get harder to satisfy. Our results show that for a high percentage of settings with just a handful of agents and tasks, the PACT algorithm leads to an equilibrium solution that is core-stable. Not surprisingly, these percentages go down with increasing number of tasks, agents, and coalition sizes.

6.5 Generalized Automated Negotiation Problem

The automated negotiation problem considered earlier in this chapter assumes that the coalition values for the negotiation network are known *apriori*. Another way of looking at this problem is that the environment has agents with single unique capabilities, and each coalition, which is a task that requires a certain set of capabilities to be performed, can be handled by exactly one group of agents. Now, most real world applications are more complex than that. Agents typically have multiple, overlapping capabilities, and there are typically multiple groups of agents that can successfully handle each coalition. Also, each agent knows about only a subset of other agents in the environment for practical settings. These agents are typically connected with some social network topology, such as a ring, star, or a random graph. In this section, we consider this generalized automated negotiation problem, and extend our PACT algorithm to address this problem (Goradia & Vidal, 2007d).

In this problem, we have a set of N self-interested, rational agents, $A = \{a_1, \ldots, a_n\}$, where each agent is a unique node in a social network. Each agent has a single fixed capability [1], $\gamma_i \in \Gamma$, assigned randomly from a uniform distribution of the interval $\Gamma = [1, \gamma]$, where γ is the number of different types of capabilities in the model. We

[1]Problems where individual agents have multiple capabilities can be transformed to this problem by having multiple agents, one for each capability, in the environment.

consider different types of social network topologies - agents organized in a ring, a random graph, and a star. Depending on the social network topology, each agent has a limited, static set of other agents in the system that it can interact and negotiate with. 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 a_i and a_j and $e_{ij} = 0$ otherwise. The social relationships among the agents are undirected, so $e_{ij} = e_{ji}$, and for all agents, $e_{ii} = 0$.

We have a set of M tasks, $T = \{t_1, \ldots, t_m\}$, which are globally announced to all the agents in the model. We consider situations where each task must be addressed by a group of agents that will perform the task. Each task $t_i \in T$ has an associated size requirement, $|t_i|$, and is defined as a pair $\langle cap(t_i), v(t_i)\rangle$, where $cap(t_i) \in \Gamma^{|t_i|}$ is a $|t_i|$-dimensional capabilities vector, and $v(t_i) \in \Re$ is the total payoff value for the task. The agents in our model have to propose valid coalitions that can fulfill every capability requirement for the tasks. Our problem can be modeled as a characteristic function game with n agents, and a characteristic function $v(S) \rightarrow \Re$ for each $S \subseteq A$. The worth of the tasks represent the coalition values of the dynamically formed agent sets for performing the tasks.

The outcome of the problem is represented as a coalition configuration, $\bar{C}$, which is a vector of tasks that were successfully assigned to the associated agent coalitions. Each element $C_i \in \bar{C}$ is a tuple $\left\langle t_i, \bar{A}_i, \bar{u_i}\right\rangle$, where t_i represents the task, $\bar{A}_i$ is the set of member agents that will jointly handle the task, and $\bar{u_i}$ is the payoff distribution vector, which defines the utility earned by each member by successfully completing the task. Here, $\sum u_i = v(t_i)$. We assume that an agent can participate in only one coalition at a time, so the coalition structure CS is a partition of agents A into $\left|\bar{C}\right|$ disjoint sets.

6.6 PACT Negotiation Mechanism for this Problem

Our approach to coalition formation for the above model is a two step process. First, the agents interact with each other and determine the potential coalitions that they can form. An agent's avenues for collaboration depend heavily on its social network structure, and agents with high degrees of connectivity can participate in more coalitions than others. Once the potential coalitions have been established, the second step in our approach involves agent negotiations to determine the most favorable coalitions for each of them. These negotiations yield both, the coalition structure, and the payoff distribution for each of its coalitions. We now present further details on both these steps.

6.6.1 Determining Potential Coalitions

Each agent in our system pro-actively seeks out as many coalitions as it can find to participate in. For each task in the environment that it has one of the required capabilities, it tries to determine if it's immediate neighbors in the social network can provide the rest of the capabilities. A task is considered feasible for an agent if it, by cooperating with it's neighbors, can fulfill all the capability requirements for the task successfully. If a candidate task is not feasible by its immediate neighbors, then the agent asks them for referrals from their own social network. If a task is still not feasible, then it is rejected. Thus each agent performs a search in its social network up to two levels deep while discovering potential avenues for cooperation. For every feasible task, the agent generates coalitions for all possible permutations for the remaining capabilities with its referral network without creating duplicates from both its own and other agent's proposals. Clearly, the more the coalitions an agent participates in, the higher its probability of being selected in the final coalition

```
FIND-POTENTIAL-COALITIONS(i)
 1  for t ∈ T
 2       do if γ_i ∈ cap(t)
 3             then new c
 4                  c^agents ← i
 5                  if |t| = 1 ∧ c ∉ C
 6                     then C ← C ∪ c
 7                     else  cap'(t) ← cap(t)\γ_i
 8                           feasible(t) ← TRUE
 9                           for γ' ∈ cap'(t)
10                                do if ∃j ∈ e_ij : γ_j = γ'
11                                      then ∀j ∈ e_ij
12                                           M ← M ∪ {j, γ'}
13                                   elseif ∃k ∈ e_jk : γ_k = γ'
14                                      then ∀k ∈ e_jk
15                                           M ← M ∪ {k, γ'}
16                                   else  feasible(t) ← FALSE
17                  if feasible(t) = TRUE
18                     then ∀J ∈ permutations(M)
19                          c^agents ← c^agents ∪ J
20                          if c ∉ C
21                             then C ← C ∪ c
```

Figure 6.15: Algorithm to find Potential Coalitions

structure. Figure 6.15 presents the complete algorithm that an agent uses to propose coalitions.

6.6.2 Determining the Coalition Configuration

The above algorithm yields us a negotiation network with all possible coalitions in the graph. The network is pruned, and only the coalitions with maximum value are retained for each set of agents. This presents us with the input that our PACT algorithm from section 6.3.4 requires to compute the feasible coalition configuration with (near-) optimal coalition structure and stable payoff division. The second step in our algorithm for the generalized automated negotiation problem is indeed the PACT

algorithm described above.

6.7 Experimental Analysis

We have developed a simulation system implementing our bargaining process and conducted a series of experiments to test the quality of the solutions that it leads to under different settings. First, we compare our algorithm's performance with that of a utilitarian algorithm. Specifically, we compare the coalition structures that the two algorithms propose in terms of the number of tasks allocated, and the total revenue generated as a result of that. Our next set of experiments concern with testing the scalability of our distributed algorithm. Specifically, we check for the duration of the bargaining process before the negotiations converge. We demonstrate the progressiveness of our algorithm by determining the rate of improvement in the quality of the solution produced with extended negotiations. Finally, we test the stability of the equilibrium solution produced by our bargaining process by evaluating whether it lies in the bargaining-set. As mentioned earlier, the network topology of the system affects its performance. We perform each test mentioned above for ring, random graph, and star topologies [2].

6.7.1 Experiments comparing our Bargaining Algorithm with a Utilitarian Solution

The goal of our first set of experiments is to determine how far the results generated by our bargaining algorithm are from a utilitarian solution. We test the coalition structure generated by our algorithm for the number of coalitions it comprises of,

[2]In order to test the relative performance of our algorithm with different topologies, we set the number of edges for random graphs the same as that for the ring topology. For the star topology, one hub node is connected to every other node in the network.

and the total value of these coalitions. We tested our system with a widely ranging distribution of agents and tasks, with scenarios where the agent/task ratio is 0.25, 0.5, 0.75, 1, 1.33, 2, and 4. Figure 6.16 shows our algorithm's performance as a ratio of that of the utilitarian solution.

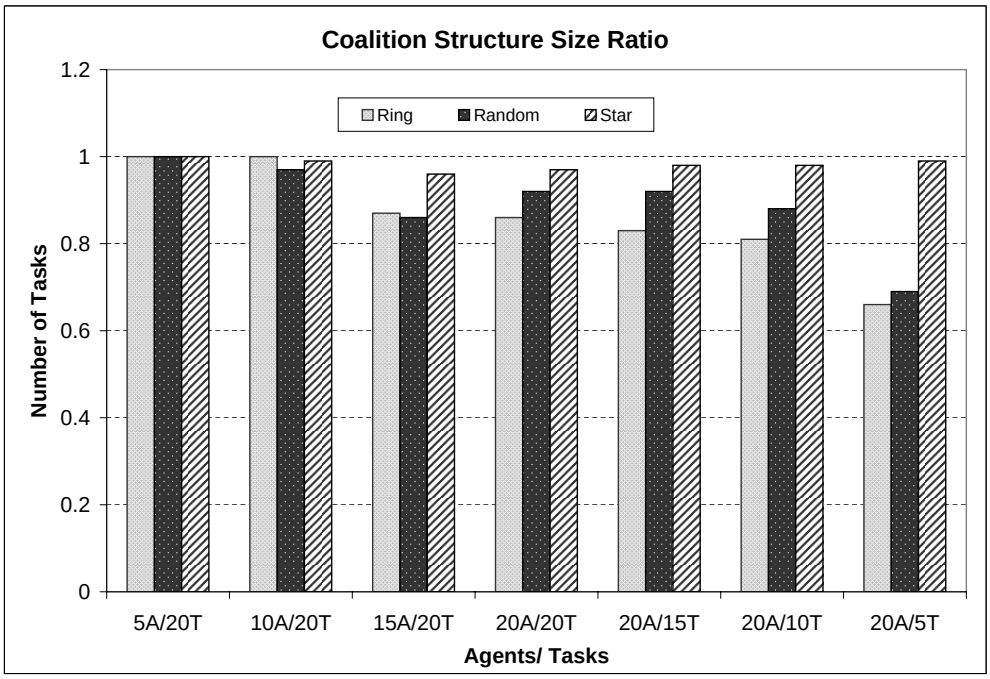

Figure 6.16: Comparison of the coalition structure size

Since our agents search for potential coalition partners two levels deep into their search tree, a star topology allows each agent to identify every other agent, and therefore has complete knowledge of the system. Our algorithm performs almost as a utilitarian algorithm in a star topology, suggesting that our negotiation scheme motivates the various agents towards coalitions that are mutually beneficial to all the concerned agents, and the payoff exchanges converge at an equilibrium point where it is in the best interest of all the participants to select those coalitions. Therefore,

although all agents adopt a myopic view in selecting the allocations most favorable to them in the post-negotiation phase, all agents end up selecting the favorable coalitions. With ring and random topologies, the limited network reachability is not a hindrance for smaller agent number and the algorithm performs remarkably well for low agent/task ratios. However, as the number of agents increase, the quality of the solution gets correspondingly lower.

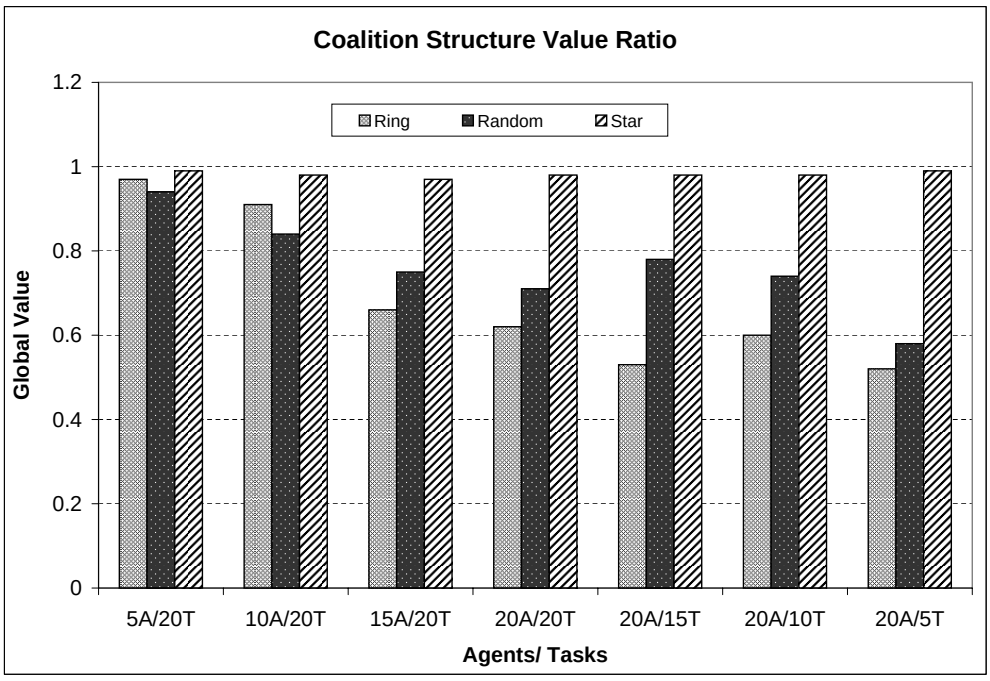

Figure 6.17: Comparison of the coalition structure value

Another issue related to the coalition structure is the total wealth generated by its winning coalitions. If the agents end up selecting tasks that are relatively less worthy, then their contribution towards the utilitarian solution will be low. Figure 6.17 shows the results for the total wealth generated by our algorithm in comparison to the highest possible value for the same set of agent/task distributions as in the

previous experiment. Simulation results show that the total values of the coalition structures generated by our bargaining algorithm are very close to those generated by the utilitarian solution for systems with fewer agents, but the quality steadily deteriorates with increasing agent populations sue to the lack of connectivity in the ring and random topologies. The star topology, however, always produces results close to the utilitarian solution, which suggests that the allocated tasks in our algorithm are mostly identical to those selected by the utilitarian solution. The agent negotiations drive them towards coalitions that have relatively higher values, and the negotiation process keeps them interested in those coalitions while also seeking a fair and stable payoff configuration.

6.7.2 Experiments to test the Scalability of our Bargaining Algorithm

We define scalability as the average number of negotiation rounds that the bargaining algorithm requires for a given setting before the negotiations terminate. In these experiments, we consider all the previous agent/task distributions to bring out any vagaries in the algorithm's performance. The results from our experiments are presented in figure 6.18.

The simulation results show that, for every topology, bargaining in systems with fewer agents terminates quickly. Negotiations in ring topologies and random graphs can continue for hundreds of rounds before terminating, while the star topologies always converge quickly. However, these results are deceptive - the actual negotiations that happen between agents in each round is directly proportional to the total number of coalitions that are proposed by all agents in the system. In rings and random graphs, due to the lower connectivity of each agent, the proposed coalitions are limited, while with star networks, these are typically higher by a magnitude of N. Consequently, while a negotiation round in rings and random graphs ends with

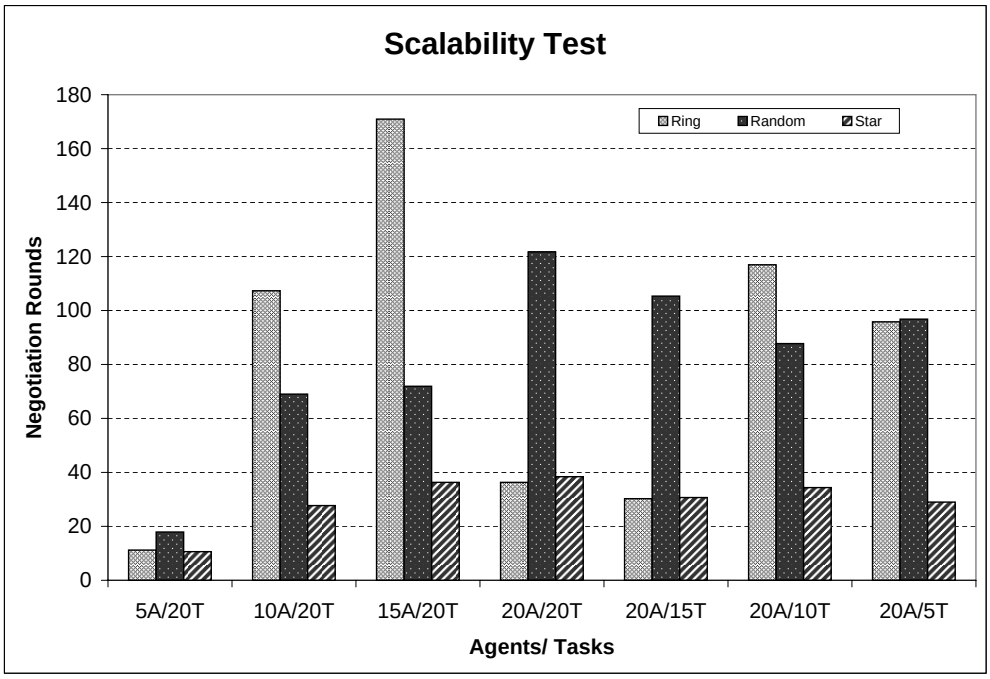

Figure 6.18: Scalability of our Bargaining Algorithm

a few message exchanges, for star structures it is prolonged for a much longer time. While optimizations are possible with star networks to limit the number of proposed coalitions, it beyond the scope for this paper as we are primarily interested in understanding the dynamics of the system under different topological settings.

6.7.3 Convergence in our Bargaining Algorithm

As we saw in the previous results, the bargaining process can run for hundreds of rounds before converging. This might be unacceptable for many competitive scenarios where an agent might be allowed only a few opportunities before it has to make its decision. Our current set of experiments evaluate the quality of the solution achieved over time in a negotiation process.

For an agent participating in our bargaining process, each round of negotiations involves setting the expectation for each of it's coalitions, and then computing the actual payoff for that round by considering the expectations of all the other members of that coalition. When the solution converges, the difference between the expectations and the actual payoffs diminishes. Therefore, the total change (sum of the absolute changes) in the expectations of all the agents for all its coalitions in the system is a good metric for testing convergence. In this set of experiments, we determine how long our algorithm takes to converge up to certain percentages of its best solution by measuring the total expectation changes for the negotiation rounds. Figure 6.19 shows the results of our tests.

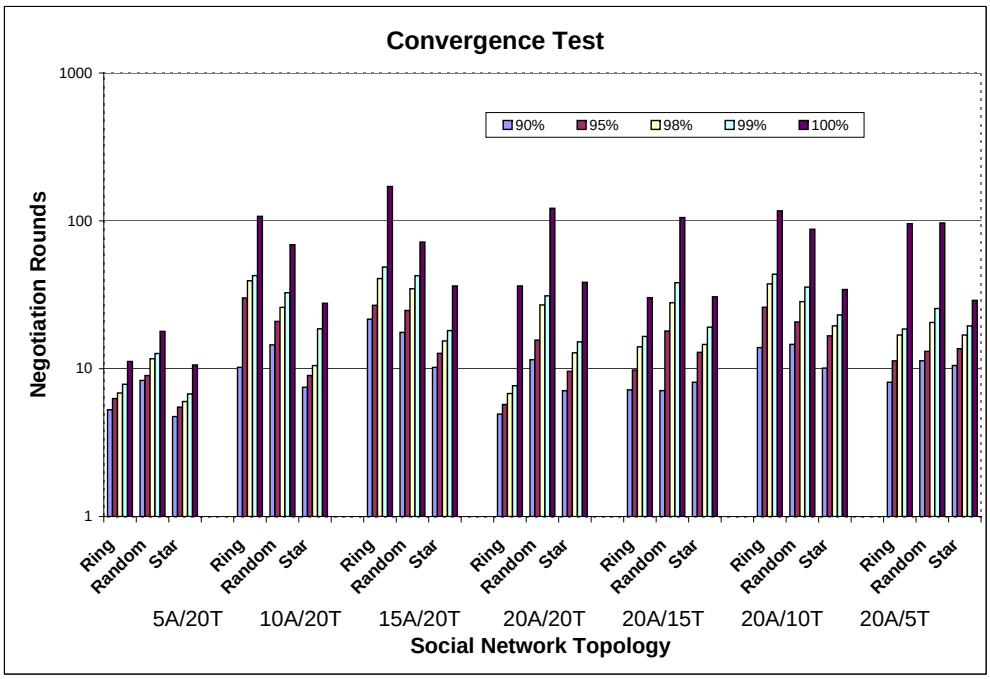

Figure 6.19: Convergence results for our Bargaining Algorithm

The most significant observation in these results is that our algorithm converges

up to 90% of its best solution within the first 10 rounds of negotiations. This is true even for the most competitive environments. The quality of the solution keeps improving as we proceed with the negotiations, with a solution 95%, 98%, and 99% close to its best arrived by approximately the $20^{th}, 30^{th}$, and 50^{th} rounds respectively. It is only the final 1% that holds back the vast majority of the runs from finishing earlier. This feature in our algorithm, whereby it converges to a close approximation of its best solution very early in the negotiations is critical for its application in various real-world domains.

6.7.4 Bargaining Set Stability Test

Our final set of experiments test the stability of the solutions arrived by our bargaining process. We choose the bargaining set as the stability solution concept primarily because for all coalition structures of all n-person games in characteristic function form, the bargaining set is always non-empty. Whenever a core solution exists in a coalitional game, it lies in the bargaining set. Also, the basic notions of objections and counterobjections are amenable to psychological interpretation, which strongly enhances the potential descriptive power of bargaining set theory (Kahan & Rapoport, 1984; Osborne & Rubinstein, 1999).

Figure 6.20 presents the stability results for our bargaining algorithm. Our bargaining algorithm is based on the idea of equal shares of the excess, a line of reasoning consistent with that of the bargaining set (Komorita, 1979). However, the stability results for the payoff configurations reached by our algorithm are poor for all topologies and distributions. One of the reasons for these negative results is that the agents in our system are allowed to terminate the negotiations before they converge according to the EET, thereby avoiding the irrational situations where they incur a loss by participating in a coalition. Also, there can be scenarios in our model where the system turns into a veto game and we have a monopolist agent(s) with exclusive ca-

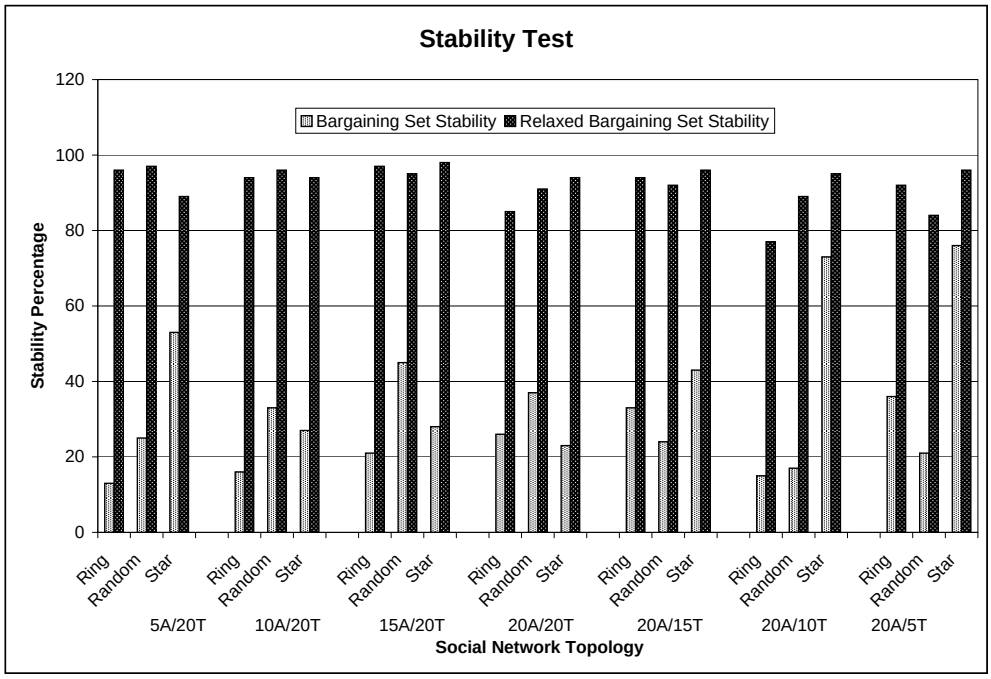

Figure 6.20: Stability results for our Bargaining Algorithm

pabilities to perform tasks in the system. The bargaining set and the core solution for such games is the irrational payoff configuration where the monopolist gets the entire revenue from the coalition, a situation that will not be allowed by other agents. This tendency reduces with increasing agent numbers and higher connectivity, as shown by the increase in the stability percentage for star topologies with higher agent/task distribution ratios.

For scenarios mentioned above, our algorithm does not yield bargaining set stable solutions because the monopolistic agent can raise objections against the weaker agents for their payoffs in the equilibrium solutions, which the latter cannot counter-object. To prevent such objections, we tested our solutions with a relaxed version of the bargaining set, where agents can object only against those with a higher payoff

than theirs. This prevents veto game-like scenarios in small coalitions, and is not a strong constraint in large coalitions where the variance in the agent payoffs is unlikely to be high. Under these constraints, our solution produces stable solutions, as shown in figure 6.20.

6.8 Summary

In this chapter, we built on our discussion about modeling automated negotiation problem as a negotiation network from the previous chapter, and defined the automated negotiation problem in terms of a negotiation network. We delved deeper into our proposition from the previous chapter that coalition formation is the most appropriate model for coordinating the agents in negotiation networks, and presented the computational complexity of finding the optimal solution for the automated negotiation problem. Since this is a hyper-exponentially complex problem, we need good heuristic approaches for solving this important problem in real-world situations. Towards this end, we presented PACT - our fully distributed, progressive, anytime algorithm for coalition formation. It addresses both coalition structure generation and payoff distribution among selfish agents simultaneously, cleverly handling the constraints of the two subproblems to find a satisfactory solution in polynomial time. We conducted extensive experiments to test the various properties of the algorithm in order to ascertain its usefulness. Our results show that for most cases, PACT generates coalition structures and maximizes social welfare to the extent that the optimal solution does. Being a hill-climbing algorithm, PACT is very efficient, and we show experimentally that it scales very well with respect to the number of agents, the number of tasks, and the coalition size. The PACT algorithm extends to larger number of negotiation rounds when the experimental settings are very competitive. However, we showed experimentally that the negotiations are extended only at the latter stages

of the algorithm where it is very close to convergence. On average, even for extremely competitive settings PACT generates a solution close to 90% of its best solution after only a few tens of negotiation rounds, and it increases to 98% of the best solution well before 100 rounds. Therefore, we believe that PACT is very applicable to many real-life domains. We showed that over 90% of the PACT solutions are core-stable for environment settings with few tasks and agents, and it steadily decreases with increasing tasks, agents, and coalition sizes as many such coalition games do not have a core solution.

We also introduced the generalized automated negotiation problem, which allows agents to have capabilities. Tasks in the system can be addressed by multiple groups of agents, thereby forming multiple coalitions in the system for each task. This adds an additional layer of exponential complexity to the original automated negotiation problem, as coalition values are not known now - they have to be computed for all coalitions. Also, in the generalized automated negotiation problem, we removed the assumption from the original problem that each agent knows about every other agent in the system. Instead, the agents in the general problem have social networks, which constrain the total number of other agents that it can collaborate with to jointly handle tasks. We discuss several network topologies for the social network - ring, star, and random. We test our PACT algorithm for this general problem, appending it to our other algorithm determining the coalition values in this problem. Our experimental results show similar results to the original problem for optimality. We also show that the computed solution is modified-bargaining set stable for most cases.

Chapter 7

Nucleolus-stable Solution for the Automated Negotiation Problem

In this chapter, we focus on the stable payoff division problem. We assume that the coalition values and the optimal coalition structure have already been computed. Many approaches have been proposed recently for addressing these problems (Sandholm et al., 1999; Rahwan & Jennings, 2005; Rahwan, Ramchurn, Dang, Giovannucci, & Jennings, 2007). Our work is complementary to these efforts and can be used in conjunction with them to address the coalition formation problem in multiagent systems. As is common practice in the literature, we model coalition formation problems as *characteristic form games (CFGs)*. The worth of every coalition $S \subseteq N$ in a CFG is given by a characteristic function, $v(S)$, and the actions of all agents who are not part of S do not influence $v(S)$. As mentioned earlier, game-theorists have devised many solution concepts over the years - the core, the Shapley value, and the nucleolus being some of the most prominent of them. The *core* is arguably the most significant solution concept for defining stability in CFGs. Like the Nash equilibrium in non-cooperative games, the core has this desirable property that no coalition can unilaterally deviate from the proposed solution and obtain a better payoff for all its agent members. Unfortunately, many CFGs have empty cores, and some have

multiple core solutions. Also, computing the core solution is exponentially complex. All these factors make it difficult to design multiagent systems that are core-stable. The *Shapley value* addresses some of the issues with the core solution. It is always nonempty and unique for every CFG. It is also deemed to be fair as the agent payoffs are determined according to their marginal contributions to the selected coalition structure in the CFG. However, determining the Shapley value for a CFG is known to be NP-complete. Also, the solution proposed by the Shapley value does not always lie in the core for games where the latter is nonempty - the agents are therefore very likely to reject this solution. The *nucleolus* solution concept has many desirable properties that make it the most appropriate stability concept for engineering multiagent systems in our view. Like the Shapley value, it is nonempty and unique for every coalition structure in every CFG. It is also generally accepted as a fair way of distributing the payoffs among the participating agents. Additionally, the nucleolus is proved to be a subset of the core, and generally lies in the center of the core region for games with nonempty cores. The computational cost for finding the nucleolus solution is also known to be considerably less than that for any other solution.

Given the above background, we present an efficient, distributed algorithm for determining the nucleolus-stable payoff division of any CFG (Goradia & Vidal, 2007a). The rest of the paper is organized as follows. We provide some basic definitions, including the nucleolus, and discuss existing approaches for computing it in the next section. This is followed by a detailed description of our algorithm for computing the nucleolus in section 3. In section 4, we empirically evaluate our algorithm and present some test results. Finally, we conclude in section 5.

7.1 Basic Definitions and Related Work

Let (N, v) be a *CFG with transferable utility* (i.e. there are no restrictions on how the payoff may be divided among coalition members). This game has N agents ($n = |N|$) and a characteristic function $v : S \rightarrow \Re^+$.

For a vector $x \in \Re^n$ and $S \subseteq N$ we denote $x(S) = \sum_{i \in S} x_i$. The vector x is an *allocation* if $x(N) = v(N)$. The allocation x is an *imputation* if $x_i \geq v(\{i\})$ for all $i \in N$.

Given an imputation $x \in \Re^n$, the *excess* of a coalition S (with respect to x) is defined as the number

$$e(S, x) = v(S) - x(S)$$

Given an imputation $x \in \Re^n$ for the game (N, v), let $\Theta(x)$ denote the $(2^n - 2)$-dimensional vector of all nontrivial excesses (the *excess vector*) $e(S, x), \phi \neq S \neq N$, arranged in nonincreasing order. For two such imputations x and y, $\Theta(x)$ is said to be *lexicographically greater* than $\Theta(y)$ if there exists an integer $q, 1 \leq q \leq 2^n - 2$, such that

$$\begin{aligned} &\forall p < q, \quad \Theta_p(x) = \Theta_p(y) \\ &\text{and} \qquad \Theta_q(x) > \Theta_q(y) \end{aligned}$$

If $\Theta(x)$ is not lexicographically greater than $\Theta(y)$, we write $\Theta(x) \preccurlyeq_{lex} \Theta(y)$.

With this terminology, the *nucleolus* (Schmeidler, 1969) of a game (N, v), denoted as $\mathcal{N}$, is defined to be the (unique) imputation x^* that lexicographically minimizes Θ over the set of all imputations. Formally,

$$\mathcal{N} = \left\{ \; x^* \in \Re^n \quad | \quad \forall x, \quad \Theta(x^*) \preccurlyeq_{lex} \Theta(x) \; \right\}$$

The nucleolus is a piecewise linear function of the characteristic function with a finite number of different linear pieces (Kohlberg, 1971). All approaches towards computing the nucleolus to date involve solving a sequence of linear programming

problems. To the best of our knowledge, (Potters, Reijnierse, & Ansing, 1996) provides the most efficient algorithm for finding the nucleolus of a general CFG. This algorithm finds the nucleolus after solving at most $n-1$ linear problems with at most $2^n + n - 1$ rows and $2^n - 1$ columns. So, each of the $n-1$ linear programs is large, and the number of constraints in them also grows exponentially. Also, such approaches require the usage of commercial linear programming packages and are inherently centralized. For multiagent settings, we need a decentralized approach where the individual agents can collaborate by sharing the computational load and resolving the inconsistencies through communication to jointly derive the nucleolus solution for the system. We present a novel algorithm for computing the nucleolus in a CFG that meets these requirements.

7.2 Our Distributed Algorithm for Computing the Nucleolus

We present our algorithm for finding the nucleolus in a CFG. The nucleolus solution is a single point in the entire imputation set for the CFG (see figure 7.1 for an example).

Since the nucleolus is a piecewise linear function of the values for the various agent coalitions in a CFG, it is possible to transition through the imputation set of the game such that we always arrive at a lexicographically smaller imputation than the previous one, eventually reaching equilibrium at the imputation that is lexicographically minimal. The heart of our algorithm is this transition function which allows us to start from any initial imputation in a CFG and eventually reach the nucleolus solution.

We have devised the transition function discussed above in the form of a sliding window heuristic. This heuristic allows us to scan the imputations adjacent to our current position and transition to a new imputation that takes us closer to the nucleo-

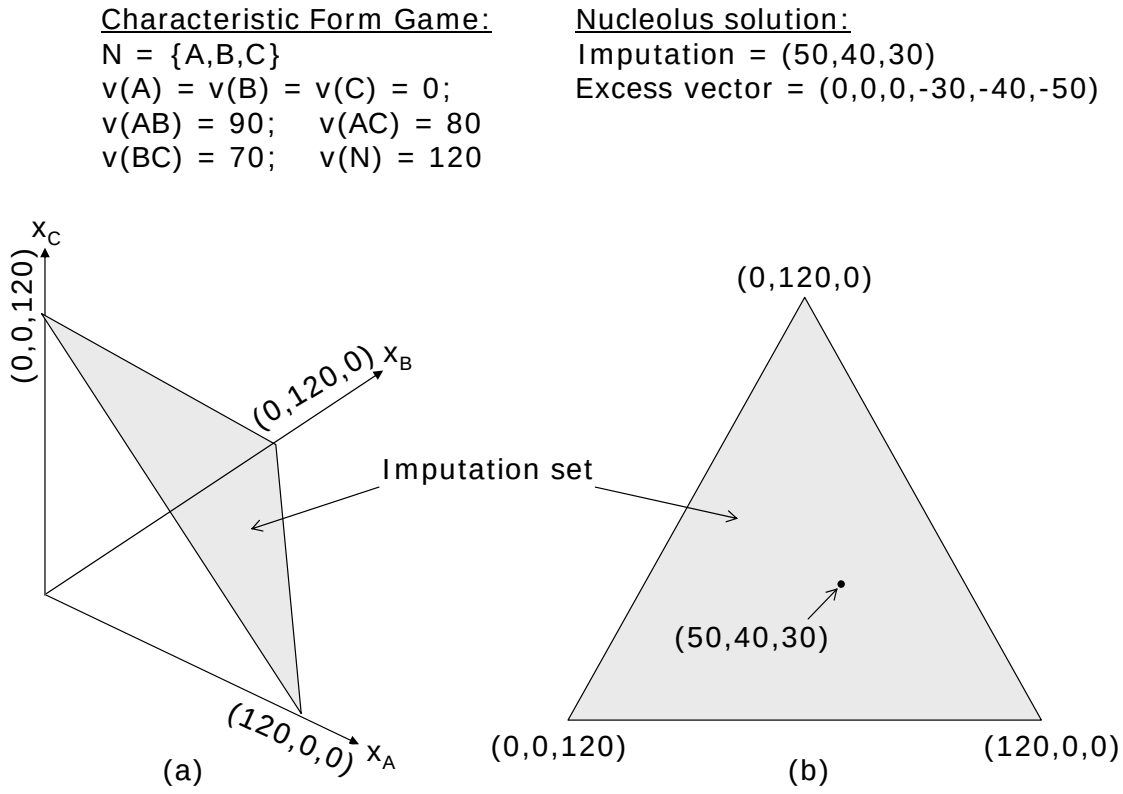

Figure 7.1: Geometric representation of imputation set and nucleolus solution for a 3-agent CFG. (a) shows the imputation set, and (b) shows the nucleolus solution in the set.

lus. The size of the window in the heuristic determines the adjacent imputations that are tested for potential transitions. Our algorithm for computing the nucleolus has two distinct stages that search through the imputation set at different levels of granularity - the coarse-level search yields an imputation close to the nucleolus (sometimes it returns the nucleolus solution itself), and the fine-level search gives the nucleolus solution with the desired precision level. Both stages use the sliding window heuristic discussed above with different window sizes and inter-imputation distances.

Having introduced the high-level idea of our approach, we now discuss the algorithm (see figure 7.2) in detail. We will use the following CFG as a running example

FIND-NUCLEOLUS(i, N, v, p)

if i is last in the preset agent order
  then $x_i^0 \leftarrow v(N) - (n-1)\lfloor \frac{v(N)}{n} \rfloor$
  else $x_i^0 \leftarrow \lfloor \frac{v(N)}{n} \rfloor$
$x^0 \in \Re^n = \bigcup_{j \in N} x_j^0$ ▷ Starting imputation
$\mathcal{N} \leftarrow$ COMPUTE-NUCLEOLUS(i, N, v, p, x^0)

COMPUTE-NUCLEOLUS(i, N, v, p, x)

$r \leftarrow 0$ ▷ Begin coarse-level search
$x^r \leftarrow x$
$x^{r+1} \leftarrow$ SLIDE-WINDOW$(i, N, v, 0, x^r, 1)$
if $x^{r+1} \neq x^r$
  then $r \leftarrow r + 1$
    go to 3
if $p = 1$
  then return x^r
$x^* \leftarrow x^r$
$r \leftarrow 0$ ▷ Begin fine-level search
$x^r \leftarrow x^*$
$x^{r+1} \leftarrow$ SLIDE-WINDOW$(i, N, v, p, x^r, 1)$
if $x^{r+1} \neq x^r$
  then $r \leftarrow r + 1$
    go to 12
$x^* \leftarrow x^r$
return x^*

SLIDE-WINDOW(i, N, v, p, x^r, w)

$\overline{ws_i} \leftarrow \{x_i^r + w, x_i^r + w - 1, \ldots, x_i^r, \ldots, x_i^r - w\}$
if $\exists r \in \overline{ws_i}$ such that $r > v(N)$ or $r < v(i)$
  then Remove r from $\overline{ws_i}$
for $j \in N \setminus i$
  do send $\overline{ws_i}$; receive $\overline{ws_j}$
▷ Generate i's share of the imputation set
$\overline{ws_i} \leftarrow \{x_i^r + w, x_i^r + w - 1, \ldots, x_i^r + 1\}$
$\overline{I_i} = \{x^1, x^2, \ldots, x^k\}$ such that $\forall_{x^\alpha, x^\beta \in I_i}$
  $x^\alpha = \bigcup_{j \in N} \binom{\overline{ws_j}}{1}$, $x^\alpha(N) = v(N)$, $x^\alpha \neq x^\beta$
▷ Determine i's best imputation
$x_i^* \leftarrow \{x^\alpha \in \overline{I_i}$ such that $\forall_{x^\alpha, x^\beta \in I_i} \; x^\alpha \preccurlyeq_{lex} x^\beta \}$
for $j \in N \setminus i$
  do send $\overline{x_i^*}$; receive $\overline{x_j^*}$
$\overline{I^*} \leftarrow \bigcup_{j \in N} x_j^*$
▷ Determine the globally best imputation
$x^* \leftarrow \{x^\alpha \in \overline{I^*}$ such that $\forall_{x^\alpha, x^\beta \in I^*} \; x^\alpha \preccurlyeq_{lex} x^\beta \}$
return x^*

Figure 7.2: Our distributed algorithm for computing the nucleolus

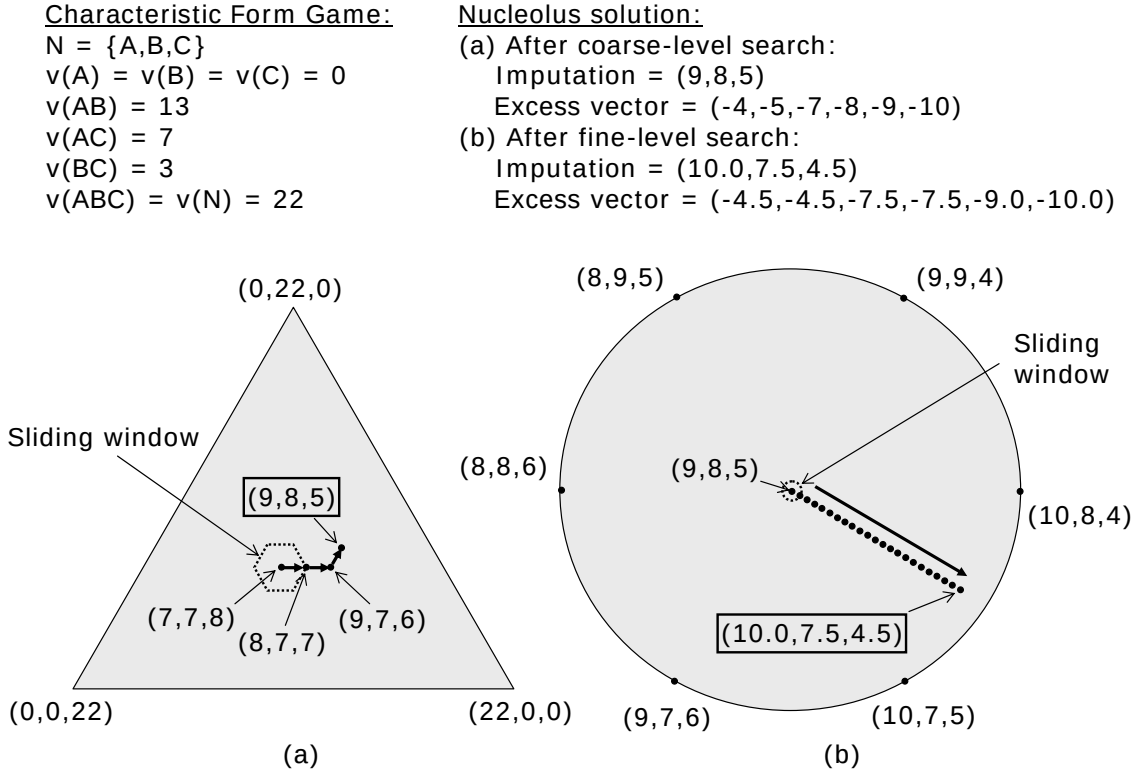

Figure 7.3: Graphical description of our algorithm for computing the nucleolus. (a) shows the coarse-level search stage, and (b) shows the fine-level search stage of the algorithm.

in our discussions (see figure 7.3 for a graphical description):

$$N = \{A, B, C\}$$
$$v(A) = v(B) = v(C) = 0$$
$$v(AB) = 3; v(AC) = 7; v(BC) = 13$$
$$v(ABC) = v(N) = 22$$

The characteristic function is assumed to be common knowledge, so all agents are aware of their relative strengths (and weaknesses) with respect to other agents. Also, it is assumed that a coalition structure involving all $n = |N|$ agents is formed and a nucleolus-stable solution dividing $v(N)$ units among the agents is desired.

The algorithm begins with the invocation of the FIND-NUCLEOLUS(i, N, v, p) procedure. Here, i represents the agent that executes this procedure, (N, v) is the CFG, and p represents the desired precision level for the outcome produced by the algorithm. All n agents in the system execute this code concurrently. Each agent begins by determining the initial imputation from which the search for the nucleolus imputation will begin. The agents fix a precedence order among themselves based on a unique key such their UID. The agents pick an equal share of $v(N)$, rounding to the closest integer, according to their preference order. The agent that is preceded by all other agents in the system gets the remaining part of $v(N)$ after all other agents have picked their share of $\lfloor v(N)/n \rfloor$. Each agent then uses this knowledge to generate the initial imputation x^0. For our example game, imputation $x^0 = (7, 7, 8)$ is formed. This is the starting point in our algorithm, and is plotted in figure 7.3. Each agent computes the excess vector for this imputation and moves on to the next stage of the algorithm.

The COMPUTE-NUCLEOLUS(i, N, v, p, x) procedure begins the agents' search for the nucleolus of the game from the initial position x. Depending on the desired precision level p, the search for the imputation that lies in the nucleolus can run upto two stages. We perform a coarse-level search in the first stage of the search process by setting the precision level to 0(lines 1-6). Thus, the inter-imputation distance (given by 10^{-p}) is 1, and only the imputations that are at unit distance from each other in the imputation set are considered. Therefore, this first stage only yields the closest integer imputation to the nucleolus (i.e. the lexicographically minimal imputation with integer payoffs for all agents). Having said that, note that the nucleolus solution for many games is an integer imputation. So, for such cases, our algorithm will produce the nucleolus imputation after the coarse-level search itself. The actual search for the imputation with lexicographically minimal excess vector is performed by the SLIDE-WINDOW procedure. The procedure is invoked repeatedly

until it returns the same imputation in successive executions, which suggests that the imputation that minimizes the excess vector over the entire imputation set is reached. This is the termination condition for the search process.

The SLIDE-WINDOW(i, N, v, p, x, w) procedure allows the agents to scan all imputations within a distance of w from the central imputation x, and determine the lexicographically minimal imputation in that subset $\overline{I}$ of the imputation set. The window size w determines how large (or small) this subset of imputations will be. Higher values of w result in more imputations being handled by the agents during each invocation, which directly results in lesser communication between the agents during the search process. However, this gain in communication bandwidth is achieved at the expense of higher redundancy (i.e. the same imputation is checked multiple times by different agents). We set $w = 1$ in all our experiments. Upon invocation, each agent first determines its share of imputations to be tested in the current procedure call. Depending on its payoff x_i from the current central imputation x, and the window size w, agent i computes its working set $\overline{ws_i}$ - the range of values $[x_i + w, x_i - w]$ - for the current invocation. All values in this range that lead to search points beyond the imputation set (i.e. $x_i > v(N)$, and imputations that are not individual rational to i (i.e. $x_i < v(\{i\})$ are removed from the working set $\overline{ws_i}$ (lines 1-3). The agents exchange their working sets (lines 4-5) so that they can all compute their share of imputations, $\overline{I_i}$, for the current invocation (lines 6-9). To reduce redundancy, each agent only checks the imputations that are more beneficial for itself than the central imputation x. For this, each agent revises its working set by eliminating all values that are less than or equal to x_i. It then generates $\overline{I_i}$ by combining the values from the working sets of all agents, such that for any allocation $x^\alpha \in \overline{I_i}$, we have $x(N) = v(N)$. Once the set $\overline{I_i}$ is generated, agent i computes the excess vectors for them all and determines the one, x_i^*, which is lexicographically minimal (line 11). All agents exchange this value to generate the set $\overline{I^*}$ of the preferred imputations by all agents, and

then determine the best imputation among them all, x^*, for the current invocation. This value is returned by the procedure, and forms the central imputation for the subsequent procedure call (lines 12-17).

The coarse-level search for our example game is shown graphically in figure 7.3(a). The hexagonal region in the figure around the central imputation $(7, 7, 8)$ is the sliding window for the first procedure call. The working sets for all agents in the system during invocation would be

$$\overline{ws_A} = \{6, 7, 8\} \quad \overline{ws_B} = \{6, 7, 8\} \quad \overline{ws_C} = \{7, 8, 9\}$$

The following set of adjacent imputations fall under the sliding window:

$$\overline{I} = \begin{array}{l} \{(6, 7, 9), (6, 8, 8), (7, 6, 9), (7, 7, 8), \\ (7, 8, 7), (8, 6, 8), (8, 7, 7)\} \end{array}$$

The distribution of the search space I^1 among the agents upon selecting the most beneficial imputations for themselves is as follows:

$$\overline{I_A} = \{(8, 6, 8), (8, 7, 7)\}$$
$$\overline{I_B} = \{(6, 8, 8), (7, 8, 7)\}$$
$$\overline{I_C} = \{(6, 7, 9), (7, 6, 9)\}$$

Note that imputation $(7, 7, 8)$ was already checked previously, so redundant calculations are avoided here. The agents check the excess vectors for their share of the imputations and determine the one that is lexicographically smaller than the rest.

These would be

$$x^*_A = (8, 7, 7) \quad \text{Excess vector} = \{-2, -7, -7, -8, -8, -11\}$$
$$x^*_B = (7, 8, 7) \quad \text{Excess vector} = \{-2, -7, -7, -7, -8, -12\}$$
$$x^*_C = (7, 6, 9) \quad \text{Excess vector} = \{0, -6, -7, -9, -9, -12\}$$

The agents exchange these imputations and unambiguously select the best one for the current round,

$$x^* = (8, 7, 7) \quad \text{Excess vector} = \{-2, -7, -7, -8, -8, -11\}$$

The window slides to this imputation for the next invocation of SLIDE-WINDOW procedure, and the process is repeated until it eventually converges. We plot these transitions in figure 7.3. The process converges to the following imputation:

$$x^* = (9, 8, 5) \quad \text{Excess vector} = \{(-4, -5, -7, -8, -9, -10\}$$

Once the agents have computed the integer-level lexicographically minimal imputation, they enter the second stage of searching in the COMPUTE-NUCLEOLUS procedure (lines 7-17) of our algorithm. Here, we perform a fine-level search resulting in a nucleolus imputation that is accurate upto the desired precision level. The precision level for the invocations of SLIDE-WINDOW procedure is set to p, so the inter-imputation distance is 10^{-p}. Everything else for fine-level search is similar to that for coarse-level search.

Getting back to our example game, figure 7.3(b) show the graphical representation of the fine-level search. This stage of computation starts with the imputation $(9, 8, 5)$. For a desired precision level of 2, the first invocation of SLIDE-WINDOW procedure in

this stage will lead to the following working sets and imputation sets for the agents:

$$\overline{ws_A} = \{8.99, 9.0, 9.01\}$$
$$\overline{ws_B} = \{7.99, 8.0, 8.01\}$$
$$\overline{ws_C} = \{4.99, 5.0, 5.01\}$$

$$\overline{I_A} = \{(9.01, 7.99, 5.0), (9.01, 8.0, 4.99)\}$$
$$\overline{I_B} = \{(8.99, 8.01, 5.0), (9.0, 8.01, 4.99)\}$$
$$\overline{I_C} = \{(8.99, 8.0, 5.01), (9.0, 7.99, 5.01)\}$$

Again, agents check the excess vectors for all their imputations, and exchange their best imputations:

$$x^*_A = (9.01, 8.0, 4.99)$$
$$\text{Excess vector} = \{-4.01, -4.99, -7.0, -8.0, -9.01, -9.99\}$$
$$x^*_B = (9.0, 8.01, 4.99)$$
$$\text{Excess vector} = \{-4.01, -4.99, -6.99, -8.01, -9.0, -10.0\}$$
$$x^*_C = (9.0, 7.99, 5.01)$$
$$\text{Excess vector} = \{-3.99, -5.01, -7.01, -7.99, -9.0, -10.0\}$$

Finally, they determine the lexicographically smallest imputation for the current procedure call:

$$x^* = (9.01, 8.0, 4.99)$$
$$\text{Excess vector} = \{-4.01, -4.99, -7.0, -8.0, -9.01, -9.99\}$$

The window slides to this imputation for the subsequent Slide-Window proce-

dure call, and the process is repeated until it converges to the imputation:

$$x^* = (10.0, 7.5, 4.5)$$
$$\text{Excess vector} = \{-4.5, -4.5, -7.5, -7.5, -9.0, -10.0\}$$

This is the nucleolus solution returned by our algorithm.

7.3 Test Results

We evaluate the performance of our algorithm by measuring some of its key properties. Specifically, we focus on the following:

Percentage of search space scanned during computation As mentioned previously, the size of the imputation set - the search space for our algorithm - grows rapidly in our environment. With this performance metric, we measure the size of the imputation set that is tested by our algorithm while finding the nucleolus imputation.

Each agent's share in the distribution We distribute the imputations equally among all the agents. However, as mentioned earlier, there is redundancy in our algorithm - the same imputation is tested by more than one agent. We measure this redundancy with this metric.

Total execution time Here we measure the actual running time of a Java implementation of the algorithm on an Intel Pentium M, 3GHz machine with 1GB of RAM.

We measure the above properties in different environment settings, where all the factors that contribute towards the exponential growth of the imputation space are considered - namely, number of agents, $v(N)$, and precision level.

Agent Set Size	Imputations				Execution Time (msec)
	Total	Tested	Percentage	Distribution	
3	1326	44.456	3.352639517	12.4160	2.5000
4	23426	108.844	0.464629045	34.0560	15.9400
5	316251	267.424	0.084560681	83.5840	76.0680
6	3478761	708.256	0.020359433	225.7200	403.3120
7	32468436	1360.78	0.004191086	435.96	3625.3120
8	264385836	4207.6	0.001591462	1356.6000	45974.2500
9	1916797311	11991.98	0.000625626	3881.1200	179279.2600

Table 7.1: Performance against agent set sizes

For our first experiment, we vary the number of agents in the system from 3 to 9. The value of the grand coalition is fixed to 50, and the precision level is set to 0 (i.e. we only seek integer solutions). The simulation results are presented in table 7.1. We see that the search space increases from 1326 for 3 agents to 19167973311 for 9 agents. However, the share of the imputation set that our algorithm requires to search before finding the nucleolus reduces with increasing agent sizes - it drops from 3.35% for 3 agents to 0.0006% for 9 agents. The starting imputation from which we begin our search plays a significant role here. The nucleolus is the lexicographically minimal imputation, and the excess vector of all imputations in the search space grows as we move further away from the nucleolus. Our algorithm simply traces all imputations (approximately) along the straight line from the starting point to the nucleolus. Each agent computes approximately 30% of all imputations that are tested in all settings because the sliding window size is set to 1 in all our experiments - the working set of candidate payoffs for all agents is of size 3 everytime and each agent opts to test the case where its payoff is the highest (1 of 3 possibilities in the working set). The execution time of our algorithm is still exponential in the number of agents.

In our second experiment, we fix the number of agents to 5 and the precision level is still 0, but we vary the value of the grand coalition, $v(N)$, from 20 to 100. Simulation results in table 7.2 show that our algorithm handles high loads of $v(N)$ very well. From $v(N) = 20$ to $v(N) = 100$, the imputation set increases more than 40

$v(N)$	Imputations				Execution
	Total	Tested	Percentage	Distribution	Time (msec)
20	10626	138.712	1.305401845	43.18	60.056
40	135751	239.068	0.176107727	74.688	80.384
60	635376	308.224	0.048510488	96.384	72.632
80	1929501	346.576	0.01796195	108.416	83.864
100	4598126	477.34	0.010381186	149.44	98.74

Table 7.2: Performance with respect to $v(N)$

times over, but our algorithm searches only 4 times more imputations - the percentage of the imputation set that is searched drops from 1.3% in the former case to 0.01% in the latter. Moreover, the execution time increases by only a little over 50% and is under 100 msec for the largest case.

So far we have tested the performance of our algorithm for settings where the agent set size or the characteristic function work in isolation. However, many multiagent settings have both these factors varying simultaneously. We test such cases in our third experiment. We set the precision level to 0 and vary the agent size / $v(N)$ settings from 3 / 20 upto 7 / 100. Refer to table 7.3 for the simulation results. Expectedly, the search space grows faster than those in the earlier experiments - while it is a meager 231 for the simplest setting, it grows to 1705904746 for the worst setting tested. Our algorithm still maintains the desirable property of reduction in the percentage of imputations in the search space that are actually tested at runtime for larger settings. From slightly over 9% for 3 / 20 setting, it goes down all the way to 0.000132% for the 7 / 100 setting. The execution times increase exponentially as in experiment 1.

In our final experiment, we test the effect of the precision level (p) on the performance of our algorithm, fixing agent set size to 5 and $v(N) = 50$. As mentioned earlier, a precision level of z is equivalent to increasing $v(N)$ by a factor of 10^z. Table 7.4 shows the simulation results of our algorithm for $p = 0, \ldots, 3$. We see that the imputation set size increase phenomenally with increasing precision requirements, as

Agents / $v(N)$	Imputations				Execution Time (msec)
	Total	Tested	Percentage	Distribution	
3 / 20	231	20.824	9.014718615	5.664	2.564
4 / 40	12341	104.056	0.843173163	32.544	13.248
5 / 60	635376	308.224	0.048510488	96.384	72.632
6 / 80	32801517	878.02	0.002676766	279.9	499.364
7 / 100	1705904746	2252.104	0.000132018	721.728	5369.62

Table 7.3: Performance for various agent set size / $v(N)$ combinations

Precision Level	Imputations				Execution Time (msec)
	Total	Tested	Percentage	Distribution	
0	316251	269.464	0.085205738	84.224	33.624
1	2656615626	499.692	1.88093E-05	162.876	73.696
2	2.60938E+13	2630.144	1.00796E-08	801.496	420.312
3	2.60469E+17	21837	8.38373E-12	6546.944	3437.996

Table 7.4: Performance against precision levels

does the execution times of our algorithm. The percentage of search space tested goes down from 0.085% for $p = 0$ to 8.38E-12 for $p = 3$.

7.4 Summary

Stable payoff division is an important problem in coalition formation, and we provide a distributed algorithm for computing a nucleolus-stable solution for any CFG. Our algorithm is particularly suited to multiagent settings where centralized solutions are often infeasible, and where the individual agents would want to play a significant role in determining their payoffs. We adopt a heuristic approach where we constantly transition to successively better solutions, eventually ending with the nucleolus solution. The computational load is equally distributed among all agents and they all play a significant role in identifying the best solution at every stage. Empirical results for our algorithm show that it performs well with increasing loads in the form of agent population, total value of the coalition, and precision requirements for the proposed solution. While the execution time of any algorithm for this hard problem will always

remain exponential, the bounds on its average case scenario can always be improved. There is high redundancy in the system with the current approach, so limiting that can lead to further improvements in this algorithm's performance. We would like to address this in the future.

Part IV

Conclusions

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this dissertation, we study the automated negotiation problem in multiagent systems with self-interested agents. We are interested in situations where there is a conflict of interest among the agents in a system, i.e. an agent has to cooperate with other agents and work as a team to accomplish tasks and make profit, but one agent's gain is always at the expense of others in the team. Clearly, we frequently encounter situations like these in many real-world domains. Our research focus in this dissertation is on devising automated negotiation mechanisms that facilitate agents in settings with conflicting interests to make decisions in their best interest and yield them optimal rewards. We believe that this work will be of interest to researchers and practitioners not only in computer science, but also in other fields such as social sciences and economics.

We now present a summary of research contributions from this dissertation:

- We introduce a novel problem - the automated negotiation problem - in multiagent systems, and show its scope and significance in a wide range of real-world applications.

- We describe a model for multiagent settings with conflicting agent interests -

a negotiation network. We show that the model possesses all the characteristics, and therefore appropriately represents the automated negotiation problem under study.

- We present a detailed theoretical analysis of the automated negotiation problem, and show that solving the problem optimally has hyper-exponential complexity.
- We present a negotiation mechanism - PACT - that provides an approximate solution to the automated negotiation problem.
- We evaluate the PACT mechanism empirically, and present the simulation results. These show that the algorithm is distributed, efficient, stable, scalable, simple, convergent, and can be terminated anytime during execution to yield the best found result until then.
- We extend our negotiation mechanism to a generalized automated negotiation problem, and present detailed experimental results for the same.
- We present an algorithm that can be juxtaposed to our negotiation mechanism to yield the payoff distribution that lies in the nucleolus of any negotiation network.

8.2 Ideas for Future Research

The real world is inherently decentralized. Examples of inherently decentralized applications include mobile computing, large-scale operations scheduling, electronic commerce over the Internet, teams of autonomous robots performing complex tasks, and distributed sensor networks. In all these applications, the information needed to make good decisions and the control necessary to implement those decisions is distributed among system components. While many current solutions for various applications

assume that all the necessary information can be aggregated at some central computer system that can then make global decisions, such approaches will no longer be appropriate for creating complex computer systems in the future. Building distributed systems comprised of autonomous or semi-autonomous components entails devising mechanisms whereby the individual components can communicate to coordinate their actions and effectively resolve issues of conflict via negotiations. My dissertation presents an automated negotiation mechanism for multiagent settings with conflicting interests. In the future, I am interested in building on this work to address other problems in various real world application domains.

8.2.1 Automated Multilateral Multiple-Issue Negotiations

My current research addresses automated negotiations for multilateral settings where the agents negotiate over the value of only a single issue (e.g. price, quality of service, etc.). The mechanism also works for situations where the agent negotiations cover multiple, mutually independent attributes, where the agents' utility functions over the possible outcomes are still linear. However, most real-world settings involve negotiations among multiple attributes that are interrelated. For example, when buying a car, price is not the only issue to be negotiated over (although it might be the dominant one). Typically, a buyer is also interested in the length of the guarantee, the terms of after-sales service, the extras that might be included (such as air conditioning, stereos and global positioning system), etc. and these issues cannot be handled independently. Multiple issues lead to an exponential growth in the negotiation space, and the buyers' utilities over the negotiation outcomes is also non-linear. Hill climbing approaches for such settings typically get stuck at some local optima and are therefore inadequate. How do we address this challenge? A possible approach worth considering is the usage of machine learning techniques, either in isolation or in combination with other techniques.

8.2.2 Electronic Business and Electronic Commerce

As we all know, electronic commerce is rapidly gaining acceptance in every industry today, and it will only become more widespread in the future. It offers opportunities to significantly improve the way that businesses interact with both their customers and their suppliers. Current (first-generation) e-commerce systems such as Amazon.com allow a user to browse an online catalog of products, choose some, and then purchase these selected products using a credit card. However, agents can lead to the second-generation e-commerce systems where many aspects of consumer buying behaviors (in B2C systems) and business-to-business transactions (in B2B systems) are automated. Sophisticated automation on both the buyer's and the seller's side can lead to applications that are more dynamic and personalized. Both buyers and sellers gain from these changes - the buyers can expect the agents to search for and retrieve the best deals available, while the sellers can have the agents that automatically customize their offerings to customers based on various parameters, such as the customer type, current seller competition, and current state of the seller's own business. This advanced degree of automation can be achieved by modeling the e-commerce systems as interacting agents. Indeed, there is a tremendous interest among the research community in devising efficient algorithms for winner determination in combinatorial auctions and exchanges. Most of the proposed solutions to date are centralized and therefore have exponential run-time complexity. My current research can potentially lead to the kind of agent negotiation mechanisms necessary for the second-generation e-commerce systems. A huge limitation of the current mechanism, which makes it inappropriate for e-business settings is that the agent strategy proposed by the mechanism is not incentive-compatible (i.e. the may not be the best move for the agent).

8.2.3 Web Services, Service-Oriented Computing, and Business Process Management

Workflow management systems, which aim to automate the processes of a business, have been around for decades. With the advent of the service-oriented computing paradigm and the Web services technology, we have means for addressing the heterogeneity that currently exists across enterprises. Today we have standard languages such as WS-BPEL and WS-CDL for defining business processes that span across enterprise boundaries. We need business process management systems to handle such cross-enterprise workflows. The current incarnations of such systems, such as IBM's BPEL4J engine, leave a lot to desire. Typically, these systems are centralized and lack the adaptability to cope with unpredictable events. Systems for handling workflows involving multiple businesses must adopt a decentralized, peer-to-peer architecture to avoid privacy and trust issues. There have been many proposals for modeling business process management systems as multiagent systems as they naturally address many issues that plague the current workflow systems. For example, adopting an agent-based approach naturally addresses the issue of aggregating information from distributed data sources owned by different parties. The inherent dependencies and conflicts of interest among the participants in the business processes can be resolved through agent interactions. Such systems could also respond more rapidly to changing circumstances in business environments. Once the integration issues between businesses are resolved, there is a need for automatically resolving higher level issues such as contractual agreements. Automated negotiation mechanisms can potentially address these concerns.

8.2.4 Semantic Web Services and the Semantic Web

Web services technology is an emerging paradigm for architecting and implementing today's business solutions. Web service providers can publish descriptions of service interfaces on the Web using the XML-based Web Services Description Language (WSDL). These descriptions include information about the message forms used to invoke the services, which can be serialized using HTTP and SOAP protocols, among others. WSDL does not, however, have a systematic way to associate meanings with the messages and message arguments that appear in those descriptions. Now, the goal of the Semantic Web is to develop XML-based languages for expressing information in a machine processable way. This Semantic Web vision takes Web-publishing of descriptions to the next level by introducing semantic description languages built on XML, which lets people publish and share ontologies - set of conceptual terms labeled by URLs - that can be used in describing other published materials. Semantic Web services are Web services in which semantic Web ontologies ascribe meanings to published service descriptions so that software systems representing prospective service clients can interpret and invoke them. This is a very appealing research area for me and I plan to contribute towards it in the future.

Bibliography

Aknine, S., Pinson, S., & Shakun, M. F. (2004). An Extended Multi-Agent Negotiation Protocol. *Autonomous Agents and Multi-Agent Systems*, *8*(1), 5–45.

Andersson, M. R., & Sandholm, T. (1998). Contract Types for Satisficing Task Allocation: II Experimental Results. In *AAAI Spring Symposium: Satisficing Models*.

Aumann, R. J., & Mashler, M. (1964). The Bargaining Set for Cooperative Games. In Dresher, M., Aumann, R. J., & Shapley, L. S. (Eds.), *Advances in Game Theory*, pp. 443–476, Princeton, N.J. Princeton University Press.

Buhler, P., & Vidal, J. M. (2005). Towards Adaptive Workflow Enactment Using Multiagent Systems. *Information Technology and Management Journal*, *6*(1), 61–87.

Cramton, P., Shoham, Y., & Steinberg, R. (Eds.). (2006). *Combinatorial Auctions*. MIT Press.

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

Davis, M., & Mashler, M. (1965). The Kernel of a Cooperative Game. *Naval Research Logistics Quarterly*, *12*, 223–259.

Davis, R., & Smith, R. G. (1983). Negotiation as a Metaphor for Distributed Problem Solving. *Artificial Intelligence*, *20*, 63–109.

Dias, B., Zlot, R., Kalra, N., & Stentz, A. (2006). Market-based multirobot coordination: a survey and analysis. *Proceedings of the IEEE*, *94*(7), 1257–1270.

eBay (2001). eBay - Your Personal Trading Community. http://www.ebay.com..

Fujishima, Y., Leyton-Brown, K., & Shoham, Y. (1999). Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pp. 548–553. Morgan Kaufmann Publishers Inc.

Gillies, D. B. (1953). *Some Theorems on n-Person Games*. Ph.D. thesis, Princeton University.

Goldberg, K., Newman, M., & Haynsworth, E. (1972). Combinatorial Analysis – Stirling Numbers of the Second Kind. In Abramowitz, M., & Stegun, I. A. (Eds.), *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, 9th printing.*, pp. 824–825, New York. Dover Publications.

Goradia, H. (2006). Towards B2B Automation Via Coalition Formation Among Service Agents. In *Proceedings of the IBM PhD Symposium at the Fourth International Conference on Service Oriented Computing*, pp. 43–48.

Goradia, H. J., & Vidal, J. M. (2005). Multiagent Workflow Enactment Using Adaptive Pricing Mechanisms. In *AAAI Planning and Scheduling for Web Services Workshop*.

Goradia, H. J., & Vidal, J. M. (2007a). A Distributed Algorithm for Finding Nucleolus-Stable Payoff Divisions. In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*.

Goradia, H. J., & Vidal, J. M. (2007b). An Equal Excess Negotiation Algorithm for Coalition Formation. Tech. rep. USC CSE TR-2007-007, University of South Carolina.

Goradia, H. J., & Vidal, J. M. (2007c). An Equal Excess Negotiation Algorithm for Coalition Formation. In *Proceedings of the Autonomous Agents and Multi-Agent Systems Conference*.

Goradia, H. J., & Vidal, J. M. (2007d). Negotiation-Based Coalition Formation among Selfish Agents. Tech. rep. USC CSE TR-2007-008, University of South Carolina.

Guttman, R. H., Moukas, A. G., & Maes, P. (1998). Agent-Mediated Electronic Commerce: A Survey. *Knowledge Engineering Review*, *13*(3), 147–159.

Harsanyi, J. C. (1956). Approaches to the Bargaining Problem Before and After the Theory of Games: A Critical Discussion of Zeuthen's, Hicks', and Nash's Theories. *Econometrica*, *24*(2), 144–157.

Harsanyi, J. C. (1977). *Rational Behavior and Bargaining Equilibrium in Games and Social Situations*. Cambridge University Press.

He, M., Jennings, N. R., & Leung, H.-F. (2003). On Agent-Mediated Electronic Commerce. *IEEE Transactions on Knowledge and Data Engineering*, *15*(4), 985–1003.

Jennings, N. R. (2001). An agent-based approach for building complex software systems. *Communications of the ACM*, *44*(4), 35–41.

Jennings, N. R., Faratin, P., Johnson, M. J., Norman, T. J., O'Brien, P., & Wiegand, M. E. (1996). Agent-based Business Process Management. *International Journal of Cooperative Information Systems*, *5*(2–3), 105–130.

Jennings, N. R., Faratin, P., Lomuscio, A. R., Parsons, S., Wooldridge, M., & Sierra, C. (2001). Automated Negotiation: Prospects Methods and Challenges. *Group Decision and Negotiation*, *10*(2), 199–215.

Kahan, J. P., & Rapoport, A. (1984). *Theories of Coalition Formation*. L. Erlbaum Associates.

Kalai, E., & Smorodinsky, M. (1975). Other Solutions to Nash's Bargaining Problem. *Econometrica*, *43*, 513–518.

Ketchpel, S. (1994). Forming coalitions in the face of uncertain rewards. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp. 414–419, Menlo Park, CA, USA. AAAI Press.

Klemperer, P. (1999). Auction theory: A guide to the literature. *Journal of Economics Surveys*, *13*(3), 227–286.

Kohlberg, E. (1971). On the Nucleolus of a Characteristic Function Game. *SIAM Journal of Applied Mathematics*, *20*, 62–66.

Komorita, S. S. (1979). An Equal Excess Model for Coalition Formation. *Behavioral Science*, *24*(6), 369–381.

Kraus, S. (1996). An overview of incentive contracting. *Artificial Intelligence*, *83*(2), 297–346.

Kraus, S. (1997). Negotiation and Cooperation in Multi-Agent Environments. *Artificial Intelligence*, *94*(1-2), 79–98.

Kraus, S. (2001). *Strategic Negotiation in Multiagent Environments*. MIT Press.

Kraus, S., Shehory, O., & Taase, G. (2003). Coalition Formation with Uncertain Heterogeneous Information. In *Proceedings of the Second International Joint Conference on Autonomous Agents and MultiAgent Systems*, pp. 1–8. ACM.

Kraus, S., Shehory, O., & Taase, G. (2004). The Advantages of Compromising in Coalition Formation with Incomplete Information. In *Proceedings of the Third International Joint Conference on Autonomous Agents and MultiAgent Systems*, pp. 588–595. ACM.

Kreps, D. M. (1990). *A Course in Microeconomic Theory*. Prentice Hall.

Lehmann, D., Muller, R., & Sandholm, T. W. (2006). The Winner Determination Problem. In Cramton, P., Shoham, Y., & Steinberg, R. (Eds.), *Combinatorial Auctions*, pp. 297–318. MIT Press.

Mas-Colell, A., Whinston, M. D., & Green, J. R. (1995). *Microeconomic Theory*. Oxford University Press.

Monderer, D., & Tennenholtz, M. (2000). Optimal auctions revisited. *Artificial Intelligence Journal*, *120*, 29–42.

Moulin, H. (1995). *Cooperative Microeconomics*. Princeton University Press.

Myerson, R. B. (1997). *Game Theory: Analysis of Conflict*. Harvard University Press.

Nash, J. F. (1950). The Bargaining Problem. *Econometrica*, *18*, 155–162.

Neumann, J. V., & Morgenstern, O. (1944). *Theory of Games and Economic Behavior*. Princeton University Press.

Osborne, M. J. (2004). *An Introduction to Game Theory*. Oxford University Press.

Osborne, M. J., & Rubinstein, A. (1999). *A Course in Game Theory*. MIT Press.

Potters, J. A. M., Reijnierse, J. H., & Ansing, M. (1996). Computing the nucleolus by solving a prolonged simplex algorithm. *Mathematics of Operations Research*, *21*(3), 757–768.

Rahwan, T., & Jennings, N. R. (2005). Distributing coalitional value calculations among cooperating agents. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pp. 152–157, Pittsburgh, USA.

Rahwan, T., & Jennings, N. R. (2007). An algorithm for distributing coalitional value calculations among cooperating agents. *Artificial Intelligence*, *171*(7–8).

Rahwan, T., Ramchurn, S. D., Dang, V. D., Giovannucci, A., & Jennings, N. R. (2007). Anytime Optimal Coalition Structure Generation. In *Proceedings of the*

22nd National Conference on Artificial Intelligence, pp. 1184–1190, Vancouver, Canada.

Raiffa, H. (1982). *The Art and Science of Negotiation*. Harvard University Press.

Rosenschein, J. S., & Zlotkin, G. (1994). *Rules of Encounter*. The MIT Press, Cambridge, MA.

Rothkopf, M. H., Pekec, A., & Harstad, R. M. (1998). Computationally Manageable Combinational Auctions. *Management Science*, *44*(8), 1131–1147.

Rubinstein, A. (1982). Perfect Equilibrium in a Bargaining Model. *Econometrica*, *50*(1), 97–110.

Sandholm, T. (1993). An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pp. 256–262.

Sandholm, T. (1996). Limitations of the Vickrey auction in computational multiagent systems. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-96)*, pp. 299–306, Menlo Park, CA. AAAI Press.

Sandholm, T. (1997). Necessary and sufficient contract types for optimal task allocation. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*.

Sandholm, T. (1998). Contract Types for Satisficing Task Allocation: I Theoretical Results. In *AAAI Spring Symposium: Satisficing Models*.

Sandholm, T. (2002). An Algorithm for Winner Determination in Combinatorial Auctions. *Artificial Intelligence*, *135*(1-2), 1–54.

Sandholm, T., Larson, K., Anderson, M., Shehory, O., & Tohmé, F. (1999). Coalition Structure Generation with Worst Case Guarantees. *Artificial Intelligence*, *111*(1-2), 209–238.

Sandholm, T., & Lesser, V. (2002). Leveled-Commitment Contracting: A Backtracking Instrument for Multiagent Systems. *AI Magazine*, *23*(3), 89–100.

Sandholm, T., & Zhou, Y. (2002). Surplus equivalence of leveled commitment contracts. *Artificial Intelligence*, *142*(2), 239–264.

Sandholm, T. W. (1999). Distributed Rational Decision Making. In Weiss, G. (Ed.), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, pp. 201–258. MIT Press, Cambridge, MA.

Sandholm, T. W. (2006). Optimal Winner Determination Algorithms. In Cramton, P., Shoham, Y., & Steinberg, R. (Eds.), *Combinatorial Auctions*, pp. 337–368. MIT Press.

Schmeidler, D. (1969). The Nucleolus of a Characteristic Function Game. *SIAM Journal of Applied Mathematics*, *17*, 1163–1170.

Selten, R. (1972). Equal Share Analysis of Characteristic Function Experiments. In Sauermann, H. (Ed.), *Contributions to Experimental Economics, vol. 3.*, pp. 130–165. Tubingen: J. C. B. Mohr (Siebeck).

Shapley, L. S. (1953). A Value for n-Person Games. In Kuhn, H. W., & Tucker, A. W. (Eds.), *Contributions to the Theory of Games, vol. 2. Annals of Mathematical Studies No. 28*, pp. 307–318, Princeton, N.J. Princeton University Press.

Shehory, O., & Kraus, S. (1998). Methods for Task Allocation via Agent Coalition Formation. *Artificial Intelligence*, *101*(1-2), 165–200.

Shehory, O., & Kraus, S. (1999). Feasible Formation of Coalitions among Autonomous Agents in Nonsuperadditve Environments. *Computational Intelligence*, *15*, 218–251.

Sierra, C., & Dignum, F. (2001). Agent-Mediated Electronic Commerce: Scientific and Technological Roadmap. In *Agent Mediated Electronic Commerce: The*

European AgentLink Perspective, Vol. 1991/2001, pp. 1–18. Springer Berlin / Heidelberg.

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

Smith, R. G., & Davis, R. (1981). Frameworks for Cooperation in Distributed Problem Solving. *IEEE Transactions on Systems, Man and Cybernetics*, *11*(1), 61–70.

Stearns, R. E. (1968). Convergent transfer schemes for n-person games. *Transactions of the American Mathematical Society*, *134*(3), 449–459.

Vickrey, W. (1961). Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance*, *16*(1), 8–37.

Vidal, J. M. (2006). *Fundamentals of Multiagent Systems: Using NetLogo Models*. Unpublished - http://www.multiagent.com/?q=fmas.

Vidal, J. M., Buhler, P., & Stahl, C. (2004). Multiagent Systems with Workflows. *IEEE Internet Computing*, *8*(1), 76–82.

Wellman, M. P. (1993). A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, *1*, 1–23.

Wellman, M. P. (1996). Market-oriented programming: Some early lessons. In Clearwater, S. (Ed.), *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific.

Wellman, M. P., & Wurman, P. R. (1998). Market-aware agents for a multiagent world. *Robotics and Autonomous Systems*, *24*, 115–125.

Willer, D. (Ed.). (1999). *Network Exchange Theory*. Praeger Publishers, Westport CT.

Wilson, R. B. (1985). Incentive efficiency of double auctions. *Econometrica*, *53*(5), 1101–1116.

Wooldridge, M. (2002). *An Introduction to MultiAgent Systems*. John Wiley and Sons.

Wooldridge, M., & Jennings, N. R. (1995). Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, *10*(2), 115–152.

Wu, L. (1977). A dynamic theory for the class of games with nonempty cores. *SIAM Journal on Applied Mathematics*, *32*(2), 328–338.

Wurman, P. R., Walsh, W. E., & Wellman, M. P. (1998). Flexible double auctions for electronic commerce: Theory and implementation. *Decision Support Systems*, *24*, 17–27.

Yeh, D. Y. (1986). A Dynamic Programming Approach to the Complete Set Partitioning Problem. *BIT Numerical Mathematics*, *26*(4), 467–474.

Zeuthen, F. (1930). *Problems of Monopoly and Economic Warfare*. Routledge.

Zlotkin, G., & Rosenschein, J. S. (1994). Coalition, Cryptography, and Stability: Mechanisms for Coalition Formation in Task Oriented Domains. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pp. 32–437. AAAI Press.