Chapter 6

Formal Analysis of Negotiation Protocols for Task Allocation

Victor Lesser[1] **, Jiaying Shen**[2] **, Ingo Weber**[3] **and Xiaoqin Shelley Zhang**[4]

[1] Computer Science Department
University of Massachusetts at Amherst
Email: lesser@cs.umass.edu

[2] Artificial Intelligence Center
SRI International
Email: shen@ai.sri.com

[3] School of Computer Science and Engineering
University of New South Wales, Sydney, Australia
Email: ingo.weber@cse.unsw.edu.au

[4] Computer and Information Science Department
University of Massachusetts at Dartmouth
Email: x2zhang@umassd.edu

Abstract. To formally understand the complex behaviors of negotiating agents so as to design appropriate mechanisms to approximate optimal performance, we have constructed a unified framework to model and analyze the task allocation problem in agent societies with different objectives. This OAR framework includes three aspects: agent's objective (O), it's negotiation attitude (A) and the reward splitting (R) among agents who cooperate to accomplish tasks. An agent's objective can span the spectrum from totally self-interested to completely cooperative, and there can be a mixture of agents with varying objectives in one agent society. This work focuses on understanding how these different aspects interact in order to achieve individual agent's objective and to produce effective system performance as well. Using the OAR framework, we develop a closed form statistical analysis to mathematically analyze the interaction between attitude parameters and reward splitting and their relationship with different objective functions

F. Lopes and H. Coelho (Eds.), *Negotiation and Argumentation in MAS*

for a simple scenario. Though the scenario is simple, it does allow us to show that being able to adjust the attitude parameter and the reward splitting is important to an agent, whether self-interested or cooperative, in order to optimize its objective function. We also present a graph model and optimality graphs, which are used for visualizing the relationships among different parameters. Additionally, we discuss how agents expected rewards are affected by changing the local attitude parameters, varying reward splitting, and the method of calculating the relational reward. This work shows that we can create a formal model to analyze interactions among agents ranging from self-interested to fully cooperative.

6.1 Introduction

An important characteristic of the next generation of complex multi-agent systems that operate in open environments will be the dynamic generation of multiple, concurrent and complex tasks. These tasks will be generated in response to emerging events in the environment, and further the character of these events may vary considerably over time. In such systems, effective task allocation is an important problem since it is not always the case that agents will have sufficient resources or expertise to fully complete all the tasks that they have generated in response to environmental events. We assume that these complex tasks can be further decomposed into subtasks. Thus, an agent may need to allocate some set of subtasks associated with one or more of its tasks to other agents in order to ensure their timely completions. Those agents who perform these subtasks refer to them as *non-local tasks*, as opposed to *local tasks* that directly arrive at those agents, and for whose completion they are responsible.

A centralized approach to task allocation is not always efficient or feasible given the large scale of the system, the dynamics of the environment, and the need to maintain the privacy of agents' local information. In such situations, agents often need to make decisions about how to allocate tasks in a distributed manner through negotiation. As part of this negotiation, when the resources do not suffice for both local tasks and the requested non-local tasks, the agent must decide which tasks to perform and which tasks to decline. This decision process can be complex and potentially needs to take into account both the objectives of the local agent and of the system as a whole.

Traditionally, cooperative agents – the agents whose goal is to maximize the social utility of a group – are assumed to always cooperate on non-local tasks requested by other agents in the group if they see that performing the requested task improves the social good more than if they use their resources

for some other tasks. On the other hand, self-interested agents whose goal is to maximize their own local reward, are assumed to put no weight on what reward the other agents may get if they help with non-local tasks, as they only care about the immediate reward they receive for helping on non-local tasks. However, recent experimental work [1, 2] has found that it is not always beneficial for an agent to cooperate with other agents about non-local tasks even if its goal is to achieve higher social utility solely based on the perceived importance of the non-local task as indicated by the requesting agent. Similarly, if an agent is interested only in its own local reward, it is still advantageous sometimes for that agent to perform a non-local task for another agent instead of its own local task even though the immediate reward of performing the local task is more than that of the non-local task. In this case, performing a non-local task that is non-optimal from a local reward perspective can lead to additional non-local tasks being offered by the system for this agent in the future, which would be beneficial for this agent in the long run [3]. Both examples indicate that what is in the social good and what is the long-term reward from doing a specific task are difficult to accurately predict without a detailed and encompassing view of the current system state and the likely future states. In a complex distributed system, the environment evolves over time. Thus, it is virtually impossible for the agents to always obtain and process all the necessary non-local information in order to achieve optimal performance, whether their goals are to maximize the social utility or local reward only.

6.1.1 Overview of the OAR Framework

To formally understand the complex behaviors of negotiating agents so as to design appropriate mechanisms to approximate optimal performance, we have constructed a unified framework to model and analyze the task allocation problem in agent societies with different objectives. This framework, named OAR, includes three aspects that can be varied:

1. **O**bjective functions: specify different goals of agents involved;
2. **A**ttitude parameters: reflect the negotiation attitude of each agent towards another agent;
3. **R**eward splitting: specifies how a contractor agent divides the reward received for finishing a specific task among itself and the agents who perform the subtasks.

A major focus of this chapter is on understanding how these different aspects interact in order to produce effective system performance in terms of the objectives of the agent society.

In this formulation, agents objectives can span the spectrum from totally self-interested to completely cooperative, and there can be a mixture of agents with varying objectives. It is very important to model the interactions of agents in a virtual society where their individual goals are not only focused on their own short-term local objectives, but include the success of the virtual society they operate in. For many multi-agent applications that we foresee, it is only through the success of the virtual society that individual agents will be able to optimize the level of achievement of their local objectives in the long term. An example of such a system is a virtual organization [4, 5] dynamically formed in an electronic marketplace [6] as illustrated in Figure 6.1. All agents within the big rectangle form a virtual organization in order to respond to customers' requests more efficiently. Besides performing individual tasks arriving directly to the agent, each agent is also motivated to help with the success of this virtual organization, because its long-term return also depends on the performance of this virtual organization. In this scenario, the objective function O of agent i is defined as the sum of its immediate reward gained by performing tasks and a share of the profit made by the virtual organization: $O_i = Reward_i + r_i * Profit(VO)$.

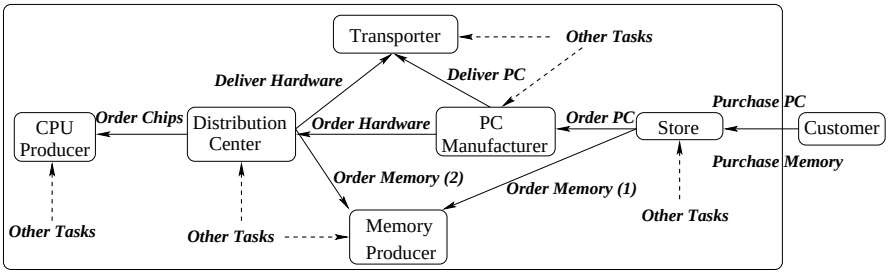


Fig. 6.1 A supply chain example

Given such an organizational setting, how should the agent evaluate the non-local task in the task allocation process? Should the agent consider not only the immediate reward it receives but also the task's influence on the utility of other agents and the performance of virtual organization? When there is conflict between a local task (which brings immediate reward) and a non-local task (which brings immediate reward and potentially also long-term reward), which one should the agent choose to perform? Additionally, how should the agent divide the reward of a task among itself and other agents who perform some subtasks of this task? The OAR framework has been developed to study these questions and understand how these decisions affect the utilities of individual agents and the agent society. In this framework,

we explicitly differentiate between the objective of an agent (which represents where the agent is on the spectrum from cooperative to self-interested) and its local decision process about whether to perform specific tasks requested by other agents. We introduce the notion of **self-directedness** and **external-directedness** for representing an agent's attitude towards another agent's request in task allocation. An agent is *completely self-directed* when it does not take into consideration how much utility the other agent can potentially gain if it commits to the requested task. In contrast, an agent is *completely externally-directed* if it sees the other agent's gain as its own when negotiating. The degree of cooperativeness/self-interestedness as defined in the agent's objective function represents the overall long-term goal of an agent, while the degree of self-directedness/external-directedness is a parameter of the local decision mechanism used to achieve the agent's long-term goal. The relationship between the degree of cooperativeness/self-interestedness and the degree of self-directedness/external-directedness is similar to the connection between a long-term strategy and short-term tactics. We represent them separately with objective functions and attitude parameters and make this distinction explicit. As we will show in the course of this chapter, the relationship between the two is often not obvious.

Using the OAR framework, we develop a closed form statistical analysis to mathematically analyze the interaction between attitude parameters and reward splitting and their relationship with different objective functions for a simple scenario. To make the math calculation tractable, a simple one-shot negotiation protocol is used in this work, which potentially can be extended such as to include de-commitment. Though the scenario is simple, it does allow us to show that being able to adjust the attitude parameter and the reward splitting is important to an agent, whether self-interested or cooperative, in order to optimize its objective function.

Using OAR and the formal model built for a simple multi-agent system, we are able to show the following.

- Reward splitting that is specific for the current environment setting is needed in addition to attitude parameters as a local mechanism to further improve the performance in both cooperative and non-cooperative systems.
- How the reward calculated by the requesting agent for an agent performing a subtask can affect overall performance.
- There are different ways to calculate relational reward, which represents "how important a task is". We proved that in a cooperative system,

one calculation of the relational reward is more expressive than others because it potentially brings a higher optimal expected social utility.

The formal analysis shows us that even simple parameters (information to transfer) can affect the optimal solution. So the meta-level control – what information to transfer – is very important. With this model we can begin to formally look at these issues. Though in this chapter we just consider comparatively simple cases, it does highlight a number of issues. The hope is that this work will encourage researchers to develop formal analysis of negotiation in more complex situation. The formal research method is very important in MAS community while not too much has been done yet.

6.1.2 Related Work

Research in distributed task allocation has been largely heuristic and experimental. [7] modeled a distributed meeting scheduling problem and is one of the first formal studies of a multi-agent system. Most formal work on negotiation is done in systems with self-interested agents [8, 9, 10]. [11] analyzes the need for meta-level communication in constructing a dynamic organizational structure. [12] studies the benefits of teaming and selflessness when using multi-agent search to solve task-oriented problems. Work on dynamic coalition formation studies the problem of finding a payoff distribution for a given game and coalition structure such that no agent has incentive to leave the coalition [13]. This is similar to the reward splitting issue we study in OAR. The difference is that the agents in the coalition find a payoff distribution through negotiation, while the reward splitting in OAR is a mechanism that is local to the manager agent, to better achieve its goal in the current environment. [14] introduced a model in which agents' utilities are linear in their own monetary income and their opponents', controlled by a parameter called altruism coefficient. This is similar to the calculations of both the objective function and the virtual utility in the OAR framework. However, their model is studied only in a competitive setting, and makes no distinction between the goal of an agent and the mechanism that an agent may employ to realize its goal. In OAR, we make this distinction explicit by representing these two related but distinct concepts with two different parameters: the objective parameter and the attitude parameter. We demonstrate that this clear distinction is important and necessary. Additionally, OAR enables us to study agents with different organizational goals in a unified setting by simply varying their objective parameters. The uniqueness of OAR lies in the fact that it represents the relationship among social welfare, agent's goal and its local

negotiation mechanisms formally, which allows us to model different multi-agent systems with it. It is the designer's concern to design mechanisms and different negotiation strategies in this framework and understand their performance in various environments. This work shows that we can create such a formal model to analyze interactions among agents ranging from self-interested to fully cooperative.

This chapter is organized as follows. We first present the details of the OAR model in Section 6.2, then we describe the general problem we are studying in Section 6.3. The statistical model and analysis work are presented in Sections 6.4, 6.5, and 6.6 respectively. Applications of the OAR model are described in Section 6.7. Finally, we conclude this work in Section 6.8.

6.2 OAR Framework

There are three components in OAR: *Objective functions* that specify the agents' goal, *Attitude parameters* that determine how an agent values each task, and *Reward splitting* that decides the reward allocation of a task that needs cooperation among agents.

6.2.1 Objective Functions

Traditionally, research on negotiation is categorized into two general classes: cooperative negotiation and competitive negotiation. In competitive negotiation, agents are self-interested and negotiate to maximize their expected local reward. In cooperative negotiation, agents work to find a solution that increases the sum of the expected reward of all involved agents. However there are other types of agents, whose goal is instead to reach a balance between their local gains and the reward of the rest of the system.

The first component of OAR is *objective function*, which specifies the goal of each agent. A general form of objective function is:

$$O_i = w_i \cdot ER_i + (1 - w_i) \sum_{j \neq i} ER_j, \tag{6.1}$$

where ER_i is the expected reward of agent i. $w_i \in [0, 1]$ is called *objective parameter* and reflects how important its local reward is to A_i as compared to the reward received by the rest of the system. For a fully cooperative agent, $w_i = 0.5$. Its goal is to maximize the *total expected reward* of the entire system, i.e., $\sum_i ER_i$. A completely self-interested agent is interested only in its own expected reward, and $w_i = 1$. An agent with $w_i = 0$ is altruistic and

considers the gains of the other agents only. The objective function unites the traditionally separate views of cooperative systems and self-interested systems. By simply varying the objective parameter, we can study agents with different goals.

If all the agents are fully cooperative, the system is a (fully) cooperative system, and each agent achieves its optimal performance when the total expected reward of the entire system is maximized. If at least one of the agents is not fully cooperative, the system is not a (fully) cooperative system, where to maximize the total expected reward is not necessarily an objective for each agent. In such a system, it is the designer's concern to design mechanisms which promote "social welfare". For example, the system designer could adapt a negotiation protocol that is Pareto efficient to ensure all agreements reached by agents are Pareto optimal.

6.2.2 Attitude Parameters

The second component of OAR is the *attitude parameter* k, $0 \leqslant k \leqslant 1$. It specifies the negotiation attitude of each agent towards another agent. The attitude parameter was previously introduced in the integrative negotiation [2] mechanism that is briefly reviewed below.

Consider the following task allocation example. For each task *t* allocated from agent A to agent B, certain rewards are transferred from agent A to agent B. There are two types of reward that could be transferred with the successful accomplishment of task *t*: *real reward* and *relational reward*. *Real reward* has positive benefits to agent B's utility, the agent collects *real reward* for its own utility increase. To reflect agent B's attitude toward agent A's outcome, let Rr_t be the *relational reward* transferred from agent A to agent B when agent B performs task *t* for agent A. Suppose that by having task *t* accomplished, agent A's own utility increases by 20 units, there are 20 units Rr_t transferred with task t, representing the utility agent A gained by having agent B perform task t. Since Rr_t is a relational reward, its only purpose is to express how important the completion of task t is for agent A. The utility produced by relational reward can be considered a *virtual utility*, in the sense that it does not contribute to agent B's local utility increase, nor is it included in the social welfare computation. However, relational reward represents the meta-level information which allows an agent to better understand how its decision affects the other agent which it is negotiating with, but from a local, not a global view. Actually, how Rr_t is mapped into agent B's virtual utility depends on how cooperative agent B is with agent A. The utility curves of the relational reward can be adjusted by the agent dynamically to reflect its dynamic relationships

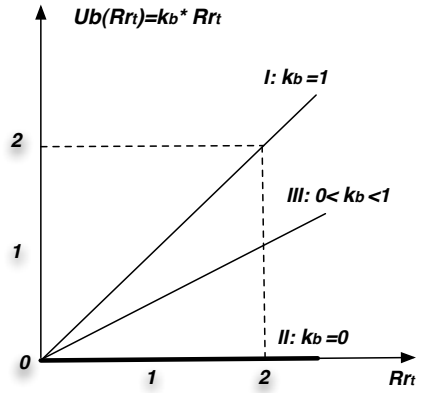


Fig. 6.2 Different mapping functions of Rr_t

with other agents. Figure 6.2 shows three different functions (**I, II, III**) for mapping Rr_t to agent B's virtual utility. Function **I**, **II** and **III** are all linear functions: $U_a(Rr_t) = k_b * Rr_t$.

If $k_b = 1$ (function **I**), $U_b(Rr_t) = Rr_t = U_a(t)$, where $U_a(t)$ denotes the utility agent A gained by transferring t[1], then agent B is completely externally-directed towards agent A;

If $0 < k_b < 1$ (function **III**), $U_b(Rr_t) < Rr_t = U_a(t)$, then agent B is partially externally-directed towards agent A;

If $k_b = 0$ (function **II**), $U_b(Rr_t) = 0$, then agent B is completely self-directed with respect to agent A. In this case, if agent A wants agent B to do task t, it needs to transfer significant real reward to agent B for performing the task.

The mapping function could also be a nonlinear function that describes a more complicated attitude of agent B to agent A, but in this research we focus on the linear mapping function $U_b(Rr_t) = k_b * Rr_t$. We choose this function because of its simplicity in application, the investigation of other functions is left for future work. k_b is referred to as the *attitude parameter* that reflects agent B's attitude toward agent A on task t.

Though relational reward provides agents the information of *how important this task is* to the other agent, this information may be inaccurate when this task requires cooperation from more than two agents. Thus it may not always be an accurate indicator of the effect on social welfare. In Section 6.7.3, we

[1]Here it is assumed that the relation reward is the same as the utility gain of agent A; there are in fact different ways of calculating relational reward, which are discussed in Section 6.7.3.

will discuss different ways to calculate relational reward and the impact of these choices. Also, in this work, it is assumed that agents do not lie about relational reward; in a more realistic world where this assumption does not hold, some reputation mechanisms [15] can be used to reduce the possibility of transferring untruthful information.

6.2.3 Reward Splitting

When an agent needs to subcontract subtasks to other agents in order to finish a task, it is this agent's decision how much to offer the other agents for accomplishing their subtasks. We call this issue *reward splitting*.

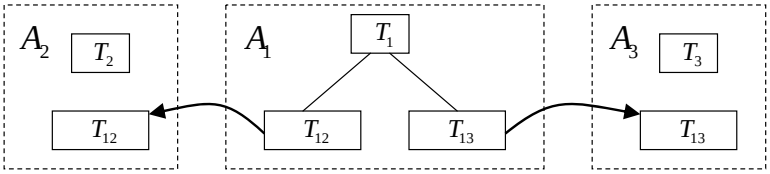


Fig. 6.3 The simplest organization structure with the necessary inter-agent interactions

Consider the example shown in Figure 6.3: in order to accomplish task T_1, agent A_1 needs to sub-contract two subtasks of T_1, T_{12} and T_{13}, to agent A_2 and agent A_3 respectively. Based on the local evaluation of A_1, if task T_1 is accomplished successfully, the utility of A_1 will increase by 20 units. Assuming the two subtasks have the same contribution to T_1, A_1 will send A_2 a task proposal on T_{12} with 10 units of relational reward (Rr_{12}), which means the accomplishment of T_{12} will generate 10 units local utility for A_1; and analogous for A_3 concerning T_{13}. Besides the relational reward, Agent A_1 has to figure out how much of the overall reward R_1 to pay to the other agents. The rewards for the subtasks are R_{12} and R_{13}, and A_1 keeps R_{11} for itself. The transferred rewards R_{12} and R_{13} are referred to as "real reward", since they actually contribute to agent A_2 and A_3's local utility increase. How to distribute the real reward R_1 among the three agents is the "reward splitting" issue.

Since agents A_2 and A_3 also receive other task proposals such as T_2 and T_3 from other agents, there may be resource conflicts between T_2 and T_{12}, or between T_3 and T_{13}, which force the agents to choose one task from these two proposals. This decision depends on the type and amount of the transferred reward associated with each task, the relational reward from A_1, and also

how the agent evaluates this relational reward (the mapping function for this relational reward).

It is a commonly used technique to use real reward as an incentive to manipulate the decisions of other agents. Similar to the use of attitude parameters as a mechanism for the contractee agents to achieve their goals, the reward splitting can be used as a local mechanism for the contractor agent to improve its performance, and is the third component of the OAR framework. The additional flexibility introduced by reward splitting is necessary to further improve an agent's performance, especially if its goal is not to maximize the total expected reward. We have seen up to 36% performance difference between different reward splitting settings for non-cooperative agents. Detailed results are presented in Section 6.7.2.

6.3 General Problem

In the previous section we introduced the OAR framework using a small Multi-Agent System as a conceivable example. Herein we discuss the general problem setting under investigation. Consider a group of agents $A_1, A_2, \ldots, A_n$ and a set of tasks $T_1, T_2, \ldots T_m$. Tasks arrive at agents randomly, T_i is not necessarily arriving at A_i. Each task has a number of parameters that observe a distribution, such as:

- dur_i: the duration of the task.
- R_i: the reward associated with each task if it is successfully completed.
- r_i: task arrives at an agent at time t with a probability of $1/r_i$.

It is assumed that each agent has one resource and the duration of the task indicates how long the task will require exclusive access to that resource. Each task $T_i, 1 \leqslant i \leqslant n$ can be decomposed into a set of subtasks: $T_{i1}, T_{i2}, \ldots, T_{im_i}$. All of the subtasks need to be completed in order for the agent A_s at whom T_i arrives to collect the reward R_i. The agent can contract out some or all of the subtasks to other agents or it can finish the task on its own. As a special case, A_s can contract out the entire task T_i. Each subtask $T_{ij}, 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m_i$ has a set of parameters as well, and they have to observe certain relationships with each other and with the original task T_i:

- dur_{ij}: the duration of the subtask.
- R_{ij}: the reward of the subtask if it is finished. $\sum_j R_{ij} + R_{is} = R_i$, where R_{is} is the net reward A_s gets after dispensing rewards for each subtask if all of the subtasks are completed. The subtask contractor gets paid when the subtask is completed though agent A_s only gets paid the whole task is accomplished. Agent A_s may cancel the request of a subtask before

it's execution starts if some other related subtasks were not successfully contracted out; in this case the contractor agent for this subtask will not get reward.
- r_{ij}: each subtask T_{ij} is generated at time t with a probability of $1/r_{ij}$, $r_{ij} = r_i$.

When a task T_i arrives at agent A_s, A_s needs to do the following for each subtask T_{ij}:

1. Decide with which agent(s) $NS(T_{ij})$ to negotiate in order to finish T_{ij}. $NS(T_{ij})$ refers to the set of agents to negotiate with concerning subtask T_{ij}.
2. Start a negotiation session with the agent(s) in $NS(T_{ij})$. Transmit the related parameters dur_{ij}, R_{ij} associated with subtask T_{ij}. In addition, it also transmits R_{is}, i.e., the reward A_s itself will get if T_i is finished successfully. By transmitting the information R_{is}, agent A_s informs other agents how important the task T_i is to itself, the other agents can take this factor into consideration when they make their local decisions. In this way, when each agent works towards maximizing its local utility, it can also consider other agent's outcome, and the implication on the organization performance it belongs to.

When agent A_l receives a request from agent A_s to do subtask T_{ij}, it does the following:

1. Decide whether T_{ij} can be fit into its own schedule or can be contracted out (contracting out a subtask follows the afore mentioned procedure of a regular task); if yes, reply committed.
2. If there is resource conflict with A_l's own schedule, compare the utilities of the conflicting tasks and commit to the one with highest utility, and de-commit from the other. When there is significant difference in task durations, a more complex model with opportunity cost is needed to make a more rational decision, such model is not included in this work.

There are three important questions that we need to answer for such a system:

- What is each agent's goal in the system? Does it want to optimize its local reward, or the total global reward, or a combination of the two?
- If there is a conflict between tasks, how does an agent evaluate each task and decide which to commit to?
- When an agent needs to contract out a task, how does it split the reward between itself and the subtasks?

Section 6.2 introduces OAR, a formal framework designed to answer these questions. In the OAR model, the utility of a subtask T_{ij} for A_l is calculated

as $U_l(T_{ij}) = R_{ij} + k_l * Rr_{ij}$, where k_l is A_l's attitude parameter; the relational reward Rr_{ij} represents the importance of accomplishing task T_{ij}; and the reward R_{ij}, the "real utility" that contributes to agent A_l's local utility increase. The "virtual utility" from $k_l * Rr_{ij}$ does not contribute to agent A_l's local utility increase, it is only being considered during the agent's decision-making process. We have explored different approaches to calculate Rr_{ij}, the details are described in Section 6.7.3.

6.4 Statistical Analysis and Verification

In this section we describe an analytical process using the simple multi-agent system depicted in Figure 6.3. This is a small agent organization with three agents A_1, A_2 and A_3. Three types of tasks T_1, T_2 and T_3 arrive at A_1, A_2 and A_3 respectively. Using this example we show in this section how one can analyze the relationship between the attitude parameter k, the environment parameters and the performance of the agent and the organization, given a specific reward splitting.

1. Each task T_i arrives at agent A_i every r_i time units, which is statistically equivalent to the following statement: T_i arrives at A_i with a probability of $1/r_i$ at each time unit.

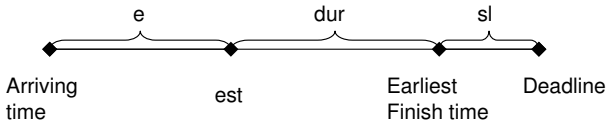


Fig. 6.4 The relationship of the different parameters of a task

2. For each task T_i, there is an earliest start time est_i, a deadline dl_i and a duration dur_i. There is a relationship between est_i, dl_i and dur_i. We introduce a new variable slack time sl_i to represent the time difference between the earliest possible finish time and the deadline: $est_i + dur_i + sl_i = dl_i$. e_i represents the difference between the arrival time of a task and its earliest start time est_i. The relationship of the different parameters of a task T_i is expressed in Figure 6.4. e_i, dur_i and sl_i are all uniformly distributed in a specific range (i.e. e_i is uniformly distributed in range $[ae_i, be_i]$). More formally,

$$P_{e_i}(x) = \begin{cases} \frac{1}{be_i - ae_i}, & ae_i < x \leqslant be_i \\ 0, & otherwise \end{cases}$$

$$P_{dur_i}(x) = \begin{cases} \frac{1}{bd_i - ad_i}, & ad_i < x \leqslant bd_i \\ 0, & otherwise \end{cases}$$

$$P_{sl_i}(x) = \begin{cases} \frac{1}{bs_i - as_i}, & as_i < x \leqslant bs_i \\ 0, & otherwise \end{cases}$$

3. For the two subtasks T_{12} and T_{13}, we have the corresponding parameters e_{1i}, dur_{1i} and sl_{1i}, where $i = 2, 3$. Each of them is uniformly distributed as well.
4. When a task T_1 arrives at A_1, the agent will start the negotiation processes with both A_2 and A_3. Associated with each T_1 is a reward R_1. Upon completion of T_1, A_1 will collect a part of the total reward R_{11} for itself, and hand the rest R_{12} and R_{13} to A_2 and A_3: $R_1 = R_{11} + R_{12} + R_{13}$. In order for the reward to be collected, both T_{12} and T_{13} have to be completed.
5. During the negotiation process with agent A_i about a subtask T_{1i}, A_1 communicates some relational reward Rr_{1i} for completing the task and thus tells A_i about the reward that A_1 itself will gain if the task is completed. Relational reward indicates how important this task is to the rest of the agents, and can be calculated in various ways in order to express different issues, as discussed in Section 6.7.3. In this section, we choose the formula $Rr_{1i} = \frac{1}{2} \cdot R_{11}$ assuming that the utility increase of A_1 is the only concern in the relational reward and the two subtasks have the same importance to T_1, therefore being assigned the same relational reward. A_i has its attitude mapping function toward A_1: $f(Rr_{1i}) = k_i \cdot Rr_{1i}$. As a result, for A_i the utility of the subtask T_{1i} is $Rn_i = R_{1i} + f(Rr_{1i})$. Currently, R_1, R_{11}, R_{12} and R_{13} are all assumed as constants, however, they also can be dynamically adjusted if needed.
6. T_2 and T_3 both have rewards, R_2 and R_3 respectively, which are uniformly distributed.

$$P_{R_i}(x) = \begin{cases} \frac{1}{br_i - ar_i}, & ar_i < x < br_i \\ 0, & otherwise \end{cases}$$

7. On receiving a subtask (T_{1i}) request, the agent A_i sees whether there is a conflict between the new task and other tasks (both the previous commitment to A_1 and its local task T_i). These other tasks include the tasks that came in before the new one and those that will come in after it. If there is no conflict, A_i will commit to the task. Otherwise, it will choose the task with higher reward. If the subtask T_{1i} is not selected, agent A_i could inform agent A_s about the rejection of the subtask. In the implementation of our simulation, Agent A_s would cancel other relate

subtask requests if the execution for these subtasks have not started. However, formally modeling such cancellation and analyzing its impact is beyond the scope of this study and subject to future study.

The only time an agent needs to choose between tasks to execute is when there is a conflict between tasks. Based on the above model, we can calculate the probability of conflict. As seen in Figure 6.5, a task of type i is in conflict with a task of type j (whether it came before task i or after) if and only if there exists a task of type j such that the following two inequalities are both true:

$$dl_i - est_j \leqslant dur_i + dur_j \ \textbf{and} \ dl_j - est_i \leqslant dur_i + dur_j \tag{6.2}$$

Rewriting (6.2) in terms of est, dur and sl, we get

$$sl_i - dur_j \leqslant est_j - est_i \leqslant dur_i - sl_j \tag{6.3}$$

For a task of type i that arrives at a given time, we define Pc_{ij} as the probability of there being a task of type j that has conflict with it. Notice that for task i, we only know its arriving time, not its other relevant parameters. In addition, we do not know any parameter of task j.

$$\begin{aligned} Pc_{ij} &= P(sl_i - dur_j \leqslant est_j - est_i \leqslant dur_i - sl_j) \\ &= \sum_{z=-\infty}^{+\infty} \sum_{y=z}^{+\infty} (1 - \prod_{x=z}^{y} (1 - P_{est_j - est_i}(x))) \\ &\quad \cdot P_{dur_i - sl_j}(y) P_{sl_i - dur_j}(z) \end{aligned} \tag{6.4}$$

First let us look at $P_{est_j - est_i}(x)$, the probability of the difference between the earliest start time of tasks T_i and T_j being x. Since the arrival time of task i is fixed, without loss of generality, let us define the arriving time of task i as 0. As a result, $est_i = e_i$, and est_j can range from $-\infty$ to $+\infty$. Therefore, $P_{est_j - est_i}(x) = P(est_j - e_i = x)$, i.e., the probability of there existing a task j that satisfies $est_j - e_i = x$. We first solve the probability of there being a task whose est is at a specified time t, which we write as $P(est = t)$:

$$\begin{aligned} P(est = t) &= \sum_{x=-\infty}^{+\infty} Pa(t - x) P_e(x) = \sum_{x=a^e+1}^{b^e} \frac{1}{r} \cdot \frac{1}{b^e - a^e} \\ &= \frac{1}{r} \end{aligned} \tag{6.5}$$

Then we can further calculate $P_{est_j - est_i}(x)$:

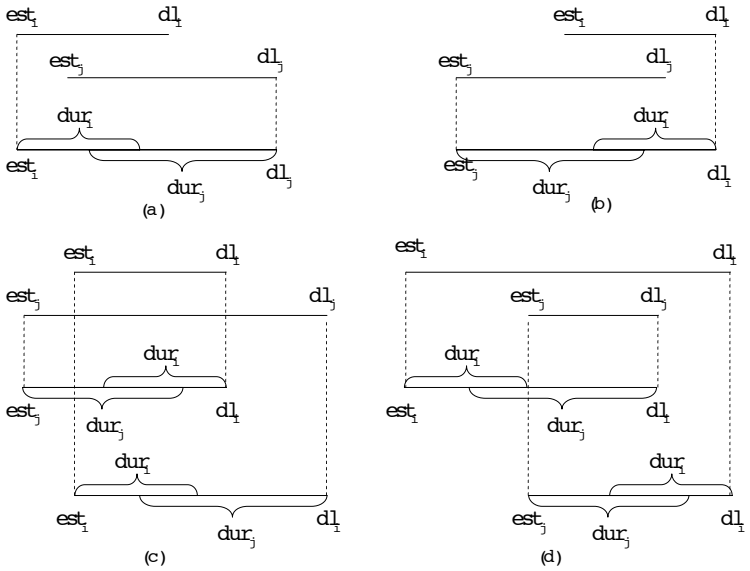


Fig. 6.5 Conflicts between tasks. Two tasks are in conflict with each other when and only when they cannot be shifted around within est and dl in order to fit both on to the schedule.

$$\begin{aligned} P_{est_j - est_i}(x) &= \sum_{y=-\infty}^{+\infty} P_{e_i}(y) P(est_j = y + x) = \sum_{y=a_i^e+1}^{b_i^e} \frac{1}{b_i^e - a_i^e} \cdot \frac{1}{r_j} \\ &= \frac{1}{r_j} \end{aligned} \tag{6.6}$$

Now let us see what $P_{dur_i - sl_j}(y)$ is.

$$\begin{aligned} & P_{dur_i - sl_j}(y) \\ = & \sum_{x=-\infty}^{+\infty} P_{dur_i}(x) \cdot P_{sl_j}(x - y) \\ = & \begin{cases} \frac{b_i^d - a_j^s - y}{(b_i^d - a_i^d)(b_j^s - a_j^s)}, & \max(a_i^d - a_j^s, b_i^d - b_j^s) < y < b_i^d - a_j^s; \\ \frac{1}{b_i^d - a_i^d}, & a_i^d - a_j^s \leqslant y \leqslant b_i^d - b_j^s; \\ \frac{1}{b_j^s - a_j^s}, & b_i^d - b_j^s \leqslant y \leqslant a_i^d - a_j^s; \\ \frac{b_j^s + y - a_i^d}{(b_i^d - a_i^d)(b_j^s - a_j^s)}, & a_i^d - b_j^s < y < \min(a_i^d - a_j^s, b_i^d - b_j^s); \\ 0, & otherwise. \end{cases} \end{aligned} \tag{6.7}$$

Similarly, we get

$$\begin{aligned} & P_{sl_i - dur_j}(z) \\ = & \begin{cases} \frac{b_i^s - a_j^d - z}{(b_i^s - a_i^s)(b_j^d - a_j^d)}, & \max(a_i^s - a_j^d, b_i^s - b_j^d) < z < b_i^s - a_j^d; \\ \frac{1}{b_i^s - a_i^s}, & a_i^s - a_j^d \leqslant z \leqslant b_i^s - b_j^d; \\ \frac{1}{b_j^d - a_j^d}, & b_i^s - b_j^d \leqslant z \leqslant a_i^s - a_j^d; \\ \frac{b_j^d + z - a_i^s}{(b_i^s - a_i^s)(b_j^d - a_j^d)}, & a_i^s - b_j^d < z < \min(a_i^s - a_j^d, b_i^s - b_j^d); \\ 0, & otherwise. \end{cases} \end{aligned} \tag{6.8}$$

Now, we can put (6.6), (6.7) and (6.8) back to (6.4) and get the probability of there being a conflict for a task that comes in at a given time. Please note that this calculation of $Pc_{i,j}$ is an approximation, since we are only considering the probability of two tasks conflicting with each other. In reality, there might be three or more tasks that can not be scheduled successfully at the same time but any two of them can be. Therefore, the real probability of conflict may be higher than our approximation.

Given the probability of conflict, we can calculate the expected reward for each agent. For A_2 and A_3, there may be two types of tasks coming in at any moment: the local task T_i and the non-local task T_{1i} with a probability of $1/r_i$ respectively, where $i = 2, 3$. Let us look at them one by one.

When a local task T_i for A_i arrives, it accumulates reward only under one of the following circumstances:

1. There is a conflict between it and one non-local task T_{1i} and there is no conflict with other local tasks. In addition, the local task reward is greater than the utility of the non-local task that it is in conflict with, i.e., $R_i > Rn_i = R_{1i} + k_i \cdot \frac{1}{2} \cdot R_{11}$. The part of expected reward gained by executing the new task in this case is then:

$$ER_i^{(1)} = Pc_{1i,i} \cdot (1 - Pc_{ii}) \cdot E(R_i | R_i > Rn_i) \tag{6.9}$$

where

$$\begin{aligned} E(R_i | R_i > Rn_i) &= \sum_{x=\lfloor Rn_i \rfloor + 1}^{b_i^r} P_{R_i}(x) \cdot x \\ &= \begin{cases} \frac{a_i^r + b_i^r + 1}{2}, & \lfloor Rn_i \rfloor < a_i^r; \\ \frac{(b_i^r - \lfloor Rn_i \rfloor)(b_i^r + \lfloor Rn_i \rfloor + 1)}{2(b_i^r - a_i^r)}, & a_i^r \leqslant \lfloor Rn_i \rfloor < b_i^r; \\ 0, & \lfloor Rn_i \rfloor \geqslant b_i^r. \end{cases} \end{aligned} \tag{6.10}$$

2. The only conflict caused by this task is with another local task T_i'. In addition, the new reward is higher than that of T_i'. The expected reward gained by executing this task under this condition is:

$$ER_i^{(2)} = (1 - Pc_{1i,i}) \cdot Pc_{ii} \cdot [E(R_i | R_i > R_i') + \frac{1}{2} E(R_i | R_i = R_i')] \tag{6.11}$$

where

$$\begin{aligned} E(R_i | R_i > R_i') &= \sum_{y=a_i^r+1}^{b_i^r} \sum_{x=y+1}^{b_i^r} x P_{R_i}(x) P_{R_i}(y) \\ &= \sum_{y=a_i^r+1}^{b_i^r} \sum_{x=y+1}^{b_i^r} \frac{x}{(b_i^r - a_i^r)^2} \end{aligned} \tag{6.12}$$

and

$$\frac{1}{2}E(R_i|R_i = R_i') = \sum_{x=a_i^r+1}^{b_i^r} x(P_{R_i}(x))^2 = \frac{a_i^r + b_i^r + 1}{4(b_i^r - a_i^r)} \tag{6.13}$$

3. There is a conflict with both another local task and a non-local task. In addition, the reward gained by the new local task is the highest.

$$\begin{aligned} ER_i^{(3)} &= Pc_{1i,i} \cdot Pc_{ii} \cdot [E(R_i|R_i > Rn_i \& R_i > R_i') \\ &\quad + \frac{1}{2}E(R_i|R_i > Rn_i \& R_i = R_i')] \end{aligned} \tag{6.14}$$

where

$$\begin{aligned} & E(R_i|R_i > Rn_i \& R_i > R_i') \\ = & \sum_{y=a_i^r+1}^{b_i^r} \sum_{x=\max(\lfloor Rn_i \rfloor+1, y+1)}^{b_i^r} P_{R_i}(x)P_{R_i}(y)\, x \\ = & \frac{1}{(b_i^r - a_i^r)^2} \sum_{y=a_i^r+1}^{b_i^r} \sum_{x=\max(\lfloor Rn_i \rfloor+1, y+1)}^{b_i^r} x \end{aligned} \tag{6.15}$$

and

$$\begin{aligned} & \frac{1}{2}E(R_i|R_i > Rn_i \& R_i = R_i') \\ = & \frac{1}{2} \sum_{x=\lfloor Rn_i \rfloor+1}^{b_i^r} [P_{R_i}(x)]^2 \cdot x \\ = & \begin{cases} 0, & \lfloor Rn_i \rfloor \geqslant b_i^r; \\ \frac{(b_i^r - \lfloor Rn_i \rfloor)(b_i^r + \lfloor Rn_i \rfloor + 1)}{4(b_i^r - a_i^r)^2}, & a_i^r \leqslant \lfloor Rn_i \rfloor < b_i^r; \\ \frac{a_i^r + b_i^r + 1}{4(b_i^r - a_i^r)}, & \lfloor Rn_i \rfloor < a_i^r. \end{cases} \end{aligned} \tag{6.16}$$

With the above equation, it is assumed that the non-local task will be selected if it has the same reward as the local task, i.e. when $R_i = Rn_i$. If a different assumption is used, this equation can be adjusted accordingly.

4. There is no conflict caused by the new task.

$$ER_i^{(4)} = (1 - Pc_{1i,i})(1 - Pc_{ii}) \cdot \frac{ar_i + br_i}{2} \tag{6.17}$$

Similarly, when a subtask T_{1i} arrives at A_i, it will choose to commit to it under four conditions, but it can accumulate this reward only when the other agent decides to commit to the other subtask as well. The request of subtask may be cancelled (before the execution of this subtask starts) if some of subtasks were not successfully contracted out. Therefore the expected reward will be:

$$ER_i^{(5)} = Pcommit_2 \cdot Pcommit_3 \cdot R_{1i} \tag{6.18}$$

where $Pcommit_i$ is the probability of agent A_i commits to the subtask T_{1i} ($i = 2, 3$), which can be calculated as the following:

$$\begin{aligned} Pcommit_i &= Pc_{1i,i}(1 - Pc_{1i,1i})P(Rn_i \geqslant R_i) \\ &\quad + \frac{1}{2}Pc_{1i,i} \cdot Pc_{1i,1i}P(Rn_i \geqslant R_i) \\ &\quad + \frac{1}{2}(1 - Pc_{1i,i})Pc_{1i,1i} + (1 - Pc_{1i,i})(1 - Pc_{1i,1i}) \end{aligned} \tag{6.19}$$

and

$$\begin{aligned} P(Rn_i \geqslant R_i) &= \sum_{x=a_i^r+1}^{\lfloor Rn_i \rfloor} P_{R_i}(x) \\ &= \begin{cases} 1, & \lfloor Rn_i \rfloor \geqslant b_i^r \\ \frac{\lfloor Rn_i \rfloor - a_i^r}{b_i^r - a_i^r}, & a_i^r \leqslant \lfloor Rn_i \rfloor \leqslant b_i^r \\ 0, & \lfloor Rn_i \rfloor \leqslant a_i^r \end{cases} \end{aligned} \tag{6.20}$$

Now we have the expected reward that A_2 or A_3 collects at each time unit:

$$ER_i = \frac{1}{r_i}(ER_i^{(1)} + ER_i^{(2)} + ER_i^{(3)} + ER_i^{(4)}) + \frac{1}{r_1}ER_i^{(5)} \tag{6.21}$$

Let us have a look at the expected reward that A_1 collects at each time unit. There is only one type of task coming in to A_1. The reward can be collected if and only if both of the other two agents commit to the subtasks. As a result,

$$ER_1 = \frac{1}{r_1} \cdot R_{11} \cdot Pcommit_2 \cdot Pcommit_3 \tag{6.22}$$

Now that we have the expected reward for each of the agents, we can calculate the k_i that will maximize the total expected reward given the set of

	r	est	dur	dl	R
T_2	t2	14	6	26+td2*[1,3]	2+tr2*[1,3]
T_3	t3	24	7	34+td3*[1,3]	2+tr3*[1,3]
sub_2	15	12	7	20+[0,2]	3
sub_3	15	23+[0,2]	6	35+[0,2]	3
T_1	15	12		35+[0,2]	25

Table 6.1 Simulation parameter setting

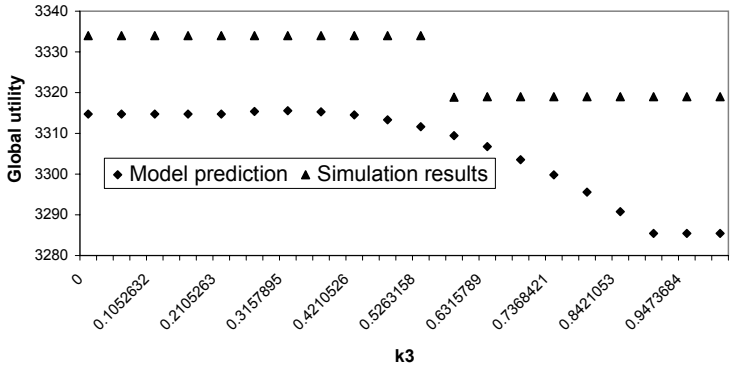


Fig. 6.6 Comparison of the model prediction and the simulation results. $t2 = t3 = 10$, $td2 = td3 = 1$, $tr2 = tr3 = 6$.

the parameters. More formally, we set k_2 and k_3 to be:

$$\arg\max_{k_2,k_3}(ER_1 + ER_2 + ER_3).$$

The above description shows how this statistical model can be used to analyze how the attitude parameter affects agent utility.

To verify this model, we have run a set of simulations, we ran a set of simulations in the integrative negotiation framework with different parameter settings (Table 6.1) to verify the model. We vary the arrival rate, deadline and reward of the tasks and record the social utility generated by the system after 950 time units for different attitude parameters k_2 and k_3. As seen in Figure 6.6, the simulation results and the theoretical prediction match well with each other, with a utility difference of around 1%. The difference in the two curves are mainly caused by the two major differences between the simulator and our theoretical model. First, the tasks in the simulator arrives at the agents every r_i time step instead of with a probability of $1/r_i$ at each

step. Though these two settings are statistically equivalent, the simulator has less chance of the same type of tasks conflicting with each other, and results in a higher utility generated by the simulation. Second, the simulator uses a scheduler that schedules all the tasks in a fixed time window together and resolves the conflicts among them. Once a task is successfully scheduled, it will not be removed from the schedule or shifted to accomodate tasks arriving in the next time window. As a result, the simulator is not as sensitive to slight parameter changes as the model is, which leads to the gradual drop in utility in the theoretical model versus the step function drop in the simulator. Other parameter settings show a similar correlation between the simulation results and the model prediction. As tasks become less flexible (varied by r and dl), conflicts become increasingly likely and global utitlity is reduced. The higher a local task's reward is compared to that of the subtask, the less likely T_1 will be finished and the more self-directed the other agent should be for the system to collect more reward. These behaviors are both predicted and explained by the model and the resultant equations.

This model has been extended and used in the OAR framework to analyze the influences of different parameters, the results are described in the following sections.

6.5 Adjusting local attitude parameters

In a real system, the environment may evolve over time. In such situations, it is unlikely that a static organization will remain optimal as the environment changes. Furthermore, it is impractical for the agents to always have a global view of the system without significant communication cost. Fortunately, an agent can often learn the other agents' behavior through past interactions with them. If agents can dynamically adjust their relationships with other agents (represented by the local attitude parameters) based on observations of each other, then the system can achieve more total expected reward than a static system. The agents A_2 and A_3 can learn the probability of the reward being actually collected from A_1 by recording the interaction history between them. From these statistics, they are able to choose their own attitude parameters (k_i) in order to maximize the total expected reward that may be collected by them and agent A_1. Expressed more formally, if agent A_2 observes the probability of A_1 handing out the reward for T_{12} as P_2, then $ER_2^{(5)}$ and ER_1 are written as follows:

$$ER_2^{(5)} = R_{12} \cdot Pcommit_2 \cdot P_2 \tag{6.23}$$

$$ER_1 = \frac{1}{r_1} \cdot R_{11} \cdot Pcommit_2 \cdot P_2 \tag{6.24}$$

In order to maximize the total expected reward of agent A_1 and A_2, as its vision of the environment allows, A_2 should set k_2 as:

$$k_2 = \arg\max_{k_2}(ER_1 + ER_2). \tag{6.25}$$

It is analogous for A_3.

There are two cases of environmental change to consider. First, there can be changes happening at A_2 or A_3 which makes the corresponding agent adjust its k_i. Second, there is a change of the local parameters at A_1 that leads to a change in k_i in one or both of the agents' attitude. When such change happens, one or both of the agents initiate the adjustment in their attitude parameters k_i in response, which leads to a change in the other agent's observation of P_i and further adjustment of k_i. We proved the following theorem [16]:

Theorem 6.1 For the small example system described in Figure 6.3, the local adjustment of the attitude parameters is stable, i.e., the process will converge.

Proof. If we fix the parameters other than k_2 and denote the utility that A_2 is trying to maximize as U_2, we can write it as a function of $x_2 = \lfloor Rn_2 \rfloor$: $U_2 = ER_1 + ER_2 = -a \cdot x_2^2 + (b + d \cdot P_2) \cdot x_2 + c$, when $a_2^r \leqslant \lfloor Rn_2 \rfloor \leqslant b_2^r$, where a, b, c, d are all constants. Then we have the optimal $\overline{\lfloor Rn_2 \rfloor}$ as $\overline{x_2} = \frac{b+d \cdot P_2}{2a}$. Since $x_2 = \lfloor Rn_2 \rfloor = R_{12} + k_2 \cdot R_{11}$, the optimal $\overline{k_2}$ changes monotonically as P_2 changes (shown in Figure 6.7(a)). When A_2 sets its new k_2, A_3's observation of P_3 changes accordingly: $P_3 = e \cdot \lfloor Rn_3 \rfloor + f$ when $a_3^r \leqslant \lfloor Rn_3 \rfloor \leqslant b_3^r$, where e and f are constants. As shown in Figure 6.7(b), P_3 changes monotonically as k_2 changes.

No matter what change in the environment causes the change in local parameter k_i, the value of k_i either increases or decreases. If the changes of both agents are towards the same direction, i.e., both of them increase, both decrease, or one of them stays the same, then as Figures 6.7(a) and 6.7(b) show, both k_2 and k_3 change monotonically without oscillation. Since there are only limited number of different values for $\lfloor Rn_i \rfloor$, k_i will converge to a certain value.

On the other hand, if k_2 and k_3 start changing towards different directions, they will both oscillate, as the directions of change caused by the two agents are different. Fortunately, the oscillation is bounded by the curves of change in

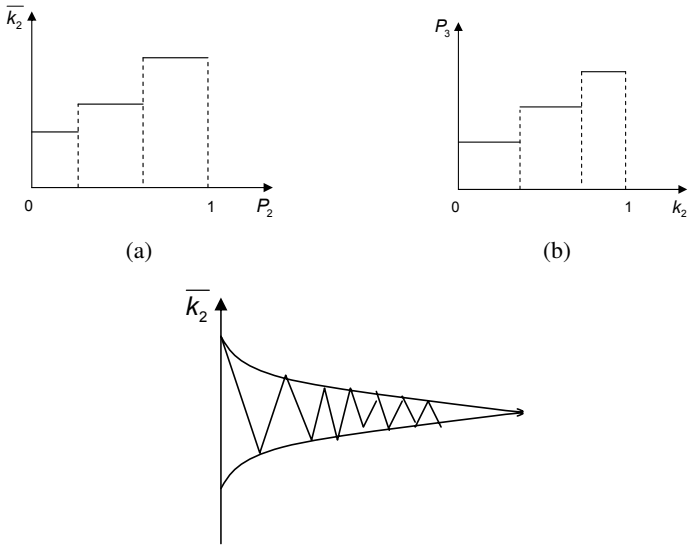


Fig. 6.7 (a) $\overline{k_2}$ changes monotonically as P_2 changes. (b) P_3 changes monotonically as k_2 changes. (c) $\overline{k_2}$ converges over time even when k_2 and k_3 change in different directions at the same time.

k_i in Figure 6.7(a) (as shown in Figure 6.7(c)), and the process will converge in the end. □

Theorem 6.1 tells us that it is safe for the agents to adjust their attitude parameters locally and reach a global equilibrium. We can add a simple learning component to each agent A_i which observes the probability of A_1 handing out the reward for non-local task T_{1i} as P_i and adjust k_i to the optimal value related to P_i dynamically. In an environment with uncertainty, the information provided by other agents may be inaccurate and prove a distraction for an agent's goal [17]. [17, 18] suggest that mechanisms that appropriately handle distraction in a complex multi-agent system are important to improving the overall system performance. In the three agent multi-linked negotiation system we are modeling in this chapter, there is uncertainty related to the rewards that A_1 promises to A_2 and A_3 and may prove distracting. P_2 and P_3 are good measures of this uncertainty. The proof of Theorem 6.1 shows that the level of uncertainty in the external information received from A_1 directly affects the amount of self-directness that an agent should have in order to optimize the total expected reward. The greater the value of P_i, the higher the optimal k_i, which means the more externally-directed A_i should be towards

	r	dur	sl	e	R
T_1	40	[20,40]	[0,20]	[0,20]	30
T_{12}	40	[20,40]	[0,20]	-	-
T_{13}	40	[20,40]	[0,20]	-	-
T_2	50	[70,80]	[0,20]	[0,20]	[14,18]
T_3	50	[70,80]	[0,20]	[0,20]	[14,18]

Table 6.2 Environmental parameters of Scenario 1

A_1 regarding task T_{1i}. Likewise, when there is more uncertainty related to the external information, an agent should be more self-directed. Therefore, the attitude parameter of an agent can be seen as an effective way to handle distraction introduced by uncertain external information.

Unfortunately, there are cases where an attitude parameter $k \in [0, 1]$ cannot fully deal with such uncertainty and guarantee the optimal system performance. As an example environment[2], consider Scenario 3 as specified in Table 6.4 and the case where R_{12} is much bigger than R_2. In this situation, A_3 is unlikely to commit to T_{13} because R_3 is much bigger than R_{13}. Therefore there is very little chance that A_2 will actually get the R_{12} since the reward is awarded only if both T_{12} and T_{13} are completed. Thus A_2 is much better off ignoring the subtask and doing its local task. In this case, even if we set k_2 to its lowest 0, A_2 will still choose to commit to T_{12} instead.

One way to deal with such uncertainty is to extend the range of the attitude parameters: $k_i \in [a^{k_i}, b^{k_i}]$, $a^{k_i} \leqslant 0$, $b^{k_i} \geqslant 1$. As shown in Figure 6.8(a), this extended range can potentially lead to better total expected reward. This optimality graph is produced based on Scenario 3 with $k_i \in [-1, 2]$. The square in this graph denotes the original range of $k_i \in [0, 1]$. We can see that the optimal total expected reward can be achieved only with a $k_3 > 1.3$ and not the original range.

6.6 Analysis Mechanism Using the OAR Framework – Numerical Optimization Process

OAR is a formal framework that can be used to model different negotiation mechanisms and study various negotiation strategies. When a certain

[2]Each environment can be modeled using the statistical model described on Section 6.4 with a set of environmental parameters specified, as shown in Table 6.4. More explanation can be found in the beginning of Section 6.6.

	r	dur	sl	e	R
T_1	40	[20,40]	[0,20]	[0,20]	30
T_{12}	40	[20,40]	[0,20]	-	-
T_{13}	40	[20,40]	[0,20]	-	-
T_2	40	[10,20]	[0,20]	[0,20]	[12,15]
T_3	40	[10,20]	[0,20]	[0,20]	[12,15]

Table 6.3 Environmental parameters of Scenario 2

environment is given, the issue of how to choose the local control parameters for each agent can be described as an optimization problem. Each environment, that can be modeled using our statistical system as described in Section 6.4, is determined by a set of environmental parameters. These parameters include the task arrival time, earliest start time, duration, deadline, and reward R_i. The value of each parameter is drawn from a uniform distribution within a specified range. Tables 6.2, 6.3, and 6.4 are examples of scenarios given by fixed environmental parameters. Besides those environmental parameters, there are some local control parameters that can be adjusted by agents, such as the partial rewards for the shared task, R_{11}, R_{12}, R_{13}, and the local attitude parameters, k_2, and k_3.

Given a fixed set of environmental parameters, there is a mapping from the local control parameters to the expected reward in each agent. The mapping function is a step function, that is neither convex nor continuous, thus the optimization problem of choosing the best values for local parameters does not fit into standard optimization problem formulation with continuous functions. There is an algorithm for the optimization of non-continuous functions that works similarly and is called the sub-gradient method [19, 20]. However, it cannot be applied to this problem since it requires the function to be convex. Again, this cannot be guaranteed in our case.

However, we have found that for fully cooperative agents, taking samples of the function values on each of the steps enables us to find the maxima and their position. We have analyzed the structure of the function and developed the mathematical mechanisms that are necessary for a seamless sampling grid, therefore enabling a numerical optimization process [21]. Due to the discreet nature of the expected reward function in focus, the number of points which have to get sampled in order to guarantee capturing the optimal value, is finite and computationally feasible. Thus, we suggest this technique and use it in the rest of this work to evaluate and compare different outcomes.

	r	dur	sl	e	R
T_1	30	[20,40]	[0,20]	[0,20]	30
T_{12}	30	[20,40]	[0,20]		
T_{13}	30	[20,40]	[0,20]		
T_2	30	[10,20]	[0,20]	[0,20]	[2,4]
T_3	50	[70,80]	[0,20]	[0,20]	[30,50]

Table 6.4 Environmental parameters of Scenario 3

The numerical optimization process is used to find the optimal values for parameters in order to maximize the expected utilities. Given the fact that $R_1 = R_{11} + R_{12} + R_{13}$ and R_1 is given and constant, there really are only four dimensions along which the expected utilities vary: R_{12}, R_{13}, k_2, and k_3. Within the four dimensional space spanned by k_2, k_3, R_{12}, and R_{13}, there is typically more than one optimal solution. This is due to the fact that there is some redundancy in the decisions, such that one optimal value of Rn_i (more precisely $\lfloor Rn_i \rfloor$) can be computed from a set of internal parameter choices. One way to select an optimal value is by calculating the set of (R_{12}, R_{13}), for which the optimum can be achieved by some setting of (k_2, k_3). From it, we choose one (R_{12}, R_{13})-pair by some metric. Then, we decide on the (k_2, k_3) which maximize the objective function. That way, secondary goals can be introduced as the metric mentioned above, e.g. fairness in the reward splitting for the shared task. Another way to select an optimal value is by first fixing the values of (k_2, k_3) such that the optimum can be reached, and then to find an optimal reward splitting (R_{12}, R_{13}).

6.7 Applications of OAR Framework

In this section, we will first present a graph model, *optimality graphs*, which we use for visualizing the relationships among different parameters. Then we will discuss how agents' expected rewards are affected by changing the local attitude parameters, varying reward splitting, and the method of calculating the relational reward.

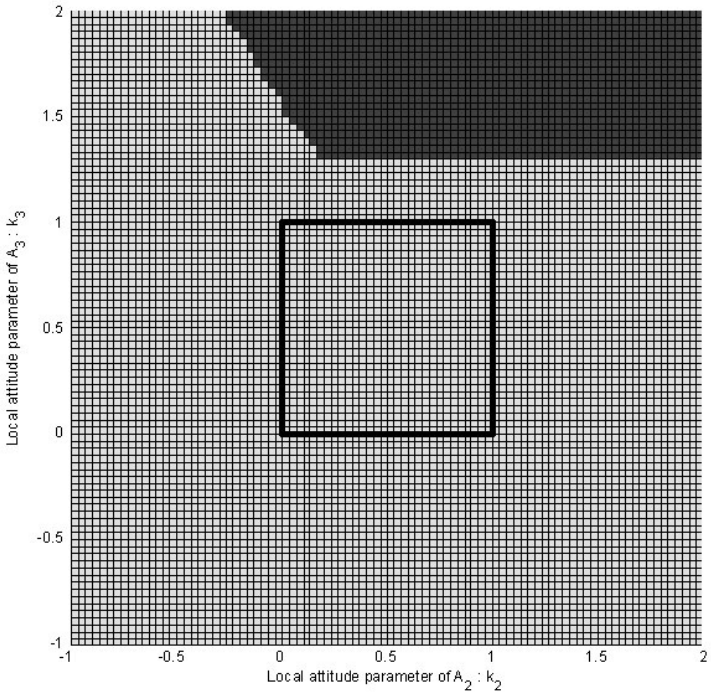
6.7.1 Optimality Graphs

As we discussed in Section 6.6, the relationship between the expected reward of an agent and the local parameters can be derived from the mathematical

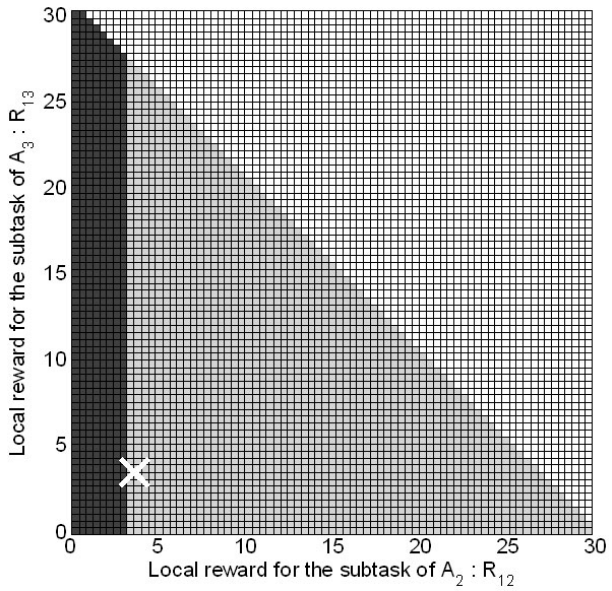
model we developed using OAR. In this model, an agent's expected reward is affected by the attitude parameters and reward splitting settings. Those particular settings of local control parameters which lead to the maximum expected reward can be found using the numerical optimization process. Oftentimes, we want to show the optimality of a setting of the attitude parameters only, or for the reward splitting alone. An *optimality graph* is designed for this purpose. Its dimensions are labeled as the different attitude parameters *or* as the different reward splitting settings. Each optimality graph has an optimization goal, which can be the expected reward of individual agent, or the sum of all agents' expect reward (total expected reward of the system: $\sum_i ER_i$). Figures 6.8(a) and 6.8(b) show examples of optimality graphs with the optimization goal being the total expected reward: $\sum_i ER_i$.

In order to examine the optimality of a certain setting of attitude parameters in the system, we first find the global optimum by varying all the different parameters. Then we fix the attitude parameters, vary the reward splitting and record the maximum total expected reward calculated using the mathematical model. If this local maximum value is the same as the global optimum, then we call this attitude parameter setting optimal. The corresponding point of this setting in the optimality graph is colored dark gray. If the setting is not optimal, the corresponding point is shown in medium gray. When a setting of attitude parameters is chosen that lies within a dark gray area, it is possible to have a reward splitting such that the system achieves the optimal total expected reward in the current environment. If the setting of the attitude parameters is chosen in a medium gray area, the optimal total expected reward cannot be achieved, regardless of the chosen reward splitting.

Similarly, an optimality graph can be plotted for reward splitting to examine the optimality of different reward splitting settings, as depicted in Figure 6.8(b). In such optimality graphs, the dark gray areas mark reward splittings which allow for an optimal total expected reward. That is, when a setting of R_{12}, and R_{13} is chosen within a dark gray area, it is possible to set k_2 and k_3 in a way that the whole multi-agent system achieves the maximal total expected reward in the current environment. For any reward splitting in a medium gray area, that is not the case: k_2 and k_3 cannot be set in a way that makes up for the suboptimal reward splitting. The light gray area – in optimality graphs of R_{12} and R_{13} for the system at hand this is always the area above the diagonal – denotes reward splittings that are invalid due to a fixed R_1 with $R_1 = R_{11} + R_{12} + R_{13}$ and $R_1, R_{11}, R_{12}, R_{13} \geqslant 0$.



(a)



(b)

Fig. 6.8 (a) An attitude parameter optimality graph for Scenario 3 in Table 6.4 showing the need for larger attitude parameter ranges. Optimization goal: $\sum_{i=1}^{3} ER_i$. (b) The reward splitting optimality graph for Scenario 3 in Table 6.4. Optimization goal: $\sum_{i=1}^{3} ER_i$. The white cross denotes the setting from [16] that lies in the suboptimal area, marked in medium gray. This shows that different reward splittings can lead to higher total expected reward for the system. **Dark gray**: optimal setting. **Medium gray**: sub-optimal setting. **Light gray**: invalid setting.

6.7.2 Varying the reward splitting and its effect

Given a set of environmental parameters, reward splitting affects each agent's expected reward and also the total expected reward. Figure 6.9 shows a typical, symmetric optimality graph with the total expected reward $\sum_{i=1}^{3} ER_i$ as the optimization goal, produced on the basis of Scenario 2 in Table 6.3. For some environments we tested, the splitting from [16], $R_{11} = \frac{3}{4}R_1$ and $R_{12} = R_{13} = \frac{1}{8}R_1$, turned out to be within the optimal region. However, in certain environments this is not the case and the fixed strategy fails to achieve the optimal total expected reward. For one of these environments, Figure 6.8(b) shows the optimality graph with optimization goal as the total expected reward $\sum_{i=1}^{3} ER_i$. The underlying environment is Scenario 3 in Table 6.4.

As we can see in Figure 6.8(b), the density of optimal solutions is fairly high which means that, in most environments, a number of reward splittings may lead to the optimal behavior. This is particularly beneficial for a distributed system where the agent only has a local view of the system and can decide on the reward splitting with limited information. With the high density of optimal settings, the agent is more likely to choose an optimal reward splitting even with its limited local knowledge.

We could exploit this fact by introducing a secondary goal to choose among the optimal solutions. Examples for such goals include fairness, a minimal reward for subtasks, balanced local gains and more.

From a global, system designer's point of view, there exists some redundancy given that one has control over all the variables. But from a more local, agent-bound perspective, this is not the case, since the agent has a limited scope of influence. As a result, the agent can make an optimal local decision for a lot of settings of the other agents' local control parameters.

Another observation from all optimality graphs in the (R_{12}, R_{13})-space with the optimization goal $\sum_{i=1}^{3} ER_i$ is that the lower left corner element is always dark gray, i.e. optimal. It turns out that this setting, which is $(R_{12}, R_{13}) = (0, 0)$, in fact always yields the optimum and therefore lends itself as a canonical solution to the reward splitting – detailed proof is presented in Appendix A. Though this solution is optimal in terms of total expected reward, it is very inflexible when it comes to secondary goals such as fairness. In a lot of cases it will result in a highly skewed distribution of the reward among the agents.

6.7.3 Different Formulae for Relational Reward Calculation

By taking into account the relational reward instead of just the local reward, an agent that is requested to do a subtask bases its decision whether to do the

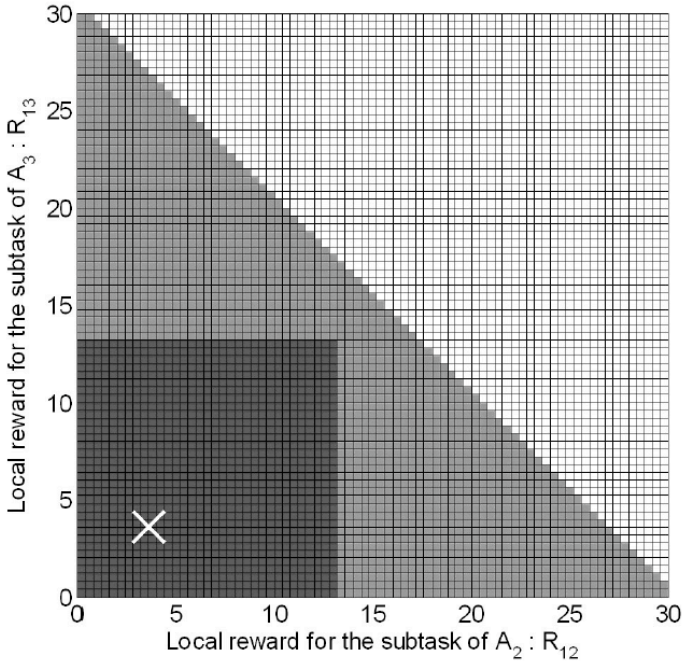


Fig. 6.9 An optimality graph in the (R_{12},R_{13})-space for Scenario 2 in Table 6.3, optimization goal: $\sum_{i=1}^{3} ER_i$. The setting that would have been selected by the fixed strategy from [16] is marked with a white cross and lies within the dark gray area. Therefore, it would have been optimal here.

task not solely on the real reward it may receive, but also on the other agents' rewards. We examine three different ways to calculate the relational reward and their expressiveness.

In the example from Fig. 6.3, A_1 needs to complete task T_1 in order to collect a reward of R_1 and asks A_2 to complete one of its subtasks. It promises a real reward of R_{12} and a relational reward of Rr_{12}. If all the subtasks of T_1 are finished successfully, A_1 itself can collect R_{11} after handing out all the real rewards. Rr_{12} can be calculated in one of the following three ways:

- $Rr_{12}^{(1)} = \frac{1}{2}R_{11}$;
- $Rr_{12}^{(2)} = R_{11}$;
- $Rr_{12}^{(3)} = R_1 - R_{12}$.

Each of the three calculations has its own motivation. Using $Rr_{12}^{(2)}$ as the relational reward, A_2 considers the reward A_1 may receive if it commits to the subtask. The motivation behind $Rr_{12}^{(1)}$ is that A_2 alone committing to the task is not going to get A_1 the reward, but only with some probability, and $\frac{1}{2}$ is a fair estimate. On the other hand, $R_1 - R_{12}$ is a more accurate measure of the reward the rest of the system would get if A_2 chooses to do the subtask. As a result, by using $Rr_{12}^{(3)}$, k_2 would be reflecting A_2's attitude towards the rest of the system instead of its attitude towards A_1 alone.

The choice of relational reward calculation depends on the goal of the agent as well as the control mechanisms available to the system. If the goal is to maximize the total expected reward in the system, one calculation of the relational reward is more expressive than another when it potentially allows a higher optimal total expected reward. We have the following proposition about the expressiveness of the three relational reward calculations in a cooperative system. Its formal proof is presented in Appendix B, based on the close form mathematical model we developed in [21].

Proposition 6.1 With the goal to maximize the total expected reward, when the reward splitting is fixed and attitude parameter k_2 can be varied between 0 and 1, $Rr_{12}^{(3)}$ is at least as expressive as $Rr_{12}^{(2)}$, and $Rr_{12}^{(2)}$ is at least as expressive as $Rr_{12}^{(1)}$. If the reward splitting is adjustable as well, then the expressiveness of $Rr_{12}^{(2)} \times Rr_{13}^{(2)}$ (denotes $Rr_{12} = R_{11}$ and $Rr_{13} = R_{11}$) and $Rr_{12}^{(3)} \times Rr_{13}^{(3)}$ (denotes $Rr_{12} = R_1 - R_{12}$ and $Rr_{13} = R_1 - R_{13}$) are the same, while $Rr_{12}^{(1)} \times Rr_{13}^{(1)}$ (denotes $Rr_{12} = \frac{1}{2}R_{11}$ and $Rr_{13} = \frac{1}{2}R_{11}$) is less expressive.

6.8 Conclusion

In this chapter we introduced OAR, a formal framework to study different issues related to negotiation. It is designed to answer the question of how agents should interact in an evolving environment in order to achieve their different goals. There are three components in OAR. Objective functions specify different goals of the agents involved in a negotiation. Attitude parameters reflect the negotiation attitude of each agent towards another agent. Reward splitting specifies how a contractor agent divides the reward between the subtasks it needs to contract out. The traditional categorization of self-interested and cooperative agents is unified by adopting a utility view. Both attitude parameters and reward splitting can be used as effective local mechanisms for the agents to realize their goals. We proved that it is safe to adjust agents' local attitude parameters since the process will converge. We also examined the effect of varying the reward splitting among agents. We studied different ways to calculate relational reward and their expressiveness. The goal of this work is not to find how to set a particular local control parameter – the optimization process is based on the knowledge of other agents' control parameters and the environmental parameters, which are not generally available in real applications. The main focus of this work is to understand the limitation and efficiency of a specific negotiation mechanism, and whether it pays off to use more complicated mechanisms. For example, how important is it to choose an appropriate reward splitting? Is it worthwhile to introduce relational reward into negotiation, and how should it be calculated? Given a particular objective, how to choose the negotiation mechanism? Such questions are asked more from the system/agent designer's perspective, who needs to decide what kind of negotiation mechanism should be adopted for an agent.

In our future work, we intend to use OAR to model and evaluate different systems with various negotiation strategies. Specifically, we are currently studying the role of de-commitment penalty as a new parameter in OAR. Another topic of interest is how the dynamic adjustment of local parameters may play out in a non-cooperative system. The study of such issues will help us to understand the behavior of a system with complex agent interactions and guide us in the design process. In OAR, the objective function represents the goal of each agent from a local perspective. We are looking into explicitly representing other criteria that may be used to measure a system's overall performance such as fairness and load balancing. We are also interested to explore whether the same optimization process can be used for agents who have a non-cooperative objective function.

Appendix A

To see why the total expected reward is always maximized with this solution, we have to take a look into the image space of the relational rewards: (Rn_2, Rn_3). As described above, we use the formula $Rn_i = R_{1i} + k_i R_{11}$ with the following constraints: $k_i \in [0,1]$, $R_1, R_{11}, R_{12}, R_{13} \geqslant 0$, and $R_1 = R_{11} + R_{12} + R_{13}$.

Hypothesis 1: The (Rn_2, Rn_3)-space with the free variables $R_{11}, R_{12}, R_{13}, k_2$, and k_3 is $[0, R_1] \times [0, R_1]$.

Hypothesis 2: The assignment $R_{11} = R_1$, $R_{12} = R_{13} = 0$ with k_2 and k_3 as free variables allows the (Rn_2, Rn_3)-space to be $[0, R_1] \times [0, R_1]$, too, and thus always allows for the optimal result.

Proof of Hypothesis 2: If $R_1 = 0$, the (Rn_2, Rn_3)-space collapses to $(0, 0)$, because $R_1, R_{11}, R_{12}, R_{13} \geqslant 0$ and $R_1 = R_{11} + R_{12} + R_{13}$ together allow only $R_{11} = R_{12} = R_{13} = 0$. For any arbitrary value of k_i, $Rn_i = R_{1i} + k_i R_{11} = 0 + k_i \cdot 0 = 0$.

If $R_1 > 0$ and we set $R_{11} = R_1, R_{12} = R_{13} = 0$, the relational reward formula reduces to $Rn_i = k_i R_1$. Therefore, Rn_2 and Rn_3 can be set independently to any value between 0 and R_1 by choosing k_i respectively.

In all cases, the setting of $R_{11} = R_1, R_{12} = R_{13} = 0$ results in the space of (Rn_2, Rn_3) being $[0, R_1] \times [0, R_1]$.

□

Proof of Hypothesis 1: The above proof includes the statement that the (Rn_2, Rn_3)-space is at least $[0, R_1] \times [0, R_1]$. We still have to show that it is not bigger for other settings. Again, for $R_1 = 0$, that is apparent and the same arguments as above hold true.

For $R_1 > 0$, Rn_i still can never be negative, because it results from multiplication and summation of non-negative terms. At the same time, Rn_i cannot be larger than R_1, due to the assignment $R_{11} = R_1, R_{1i} = R_{1j} = 0$, and $k_i = 1$ results in $Rn_i = R_1$, and any allowed change to this assignment at most decreases Rn_i:

- Increasing R_{1j} does not change anything.
- Increasing R_{1i} does not change anything, since at the same time we decrease R_{11} and there is the constraint $R_1 = R_{11} + R_{12} + R_{13}$.
- Increasing R_{11} is impossible, because of $R_1 = R_{11} + R_{12} + R_{13}$ and $R_{11}, R_{12}, R_{13} \geqslant 0$.
- $k_i \in [0, 1]$ and therefore can only be decreased. Doing so decreases Rn_i.

Altogether, we can summarize the above as $Rn_i \geqslant 0$ and $Rn_i \leqslant R_1$. Combined with the inclusion from the proof of Hypothesis 2, that the (Rn_2, Rn_3)-space is at least $[0, R_1] \times [0, R_1]$, this is equal to Hypothesis 1.

□

Due to Hypotheses 1 and 2, any optimal (Rn_2, Rn_3) can be achieved with $R_{11} = R_1$ and $R_{12} = R_{13} = 0$. Therefore, this setting is a canonical optimal solution. Using it, the optimization problem can be reduced to two dimensions only: k_2 and k_3. However, it might not be favorable under other criteria to not pay A_2 and A_3 anything, even when the main focus is on maximizing the total expected reward. Although it can be used, we would encourage other, more sophisticated approaches.

Appendix B

Hypothesis

There is an order between the image sets of the ways to calculate Rn_i and thus their expressiveness.

Say, $I(T)$ is the (infinite) set of values that can be expressed with the term T, its mathematical image. With control over the k_i only, we denote $Rn_i^{(l)}$ as $Rn_i^{(l)}(k_i)$, and get the following relationship

$$\begin{aligned} &\forall R_1, R_{11}, R_{1i}, R_{1j} \geqslant 0 \; s.t. \; R_1 = R_{11} + R_{1i} + R_{1j} \quad with \;\; k_i \in [0,1] : \\ &\qquad I(Rn_i^{(1)}(k_i)) \subset I(Rn_i^{(2)}(k_i)) \subset I(Rn_i^{(3)}(k_i)) \end{aligned} \tag{6.26}$$

When having control over the reward splitting as well, namely R_{11}, R_{1i}, and R_{1j} within certain bounds, the relations change. Since here the relational rewards for agents A_2 and A_3 are not independent of each other anymore, we have to look at both of them at the same time, resulting in the two-dimensional space spanned by Rn_2 and Rn_3. We denote $Rn_i^{(l)}$ as $Rn_i^{(l)}(p_1, p_2, ...)$ where p_m are the relevant parameters for this term. The altered relationships are the following:

$$\begin{aligned} &\forall \, R_1 \geqslant 0, with \;\; k_2, k_3 \in [0,1] \;\; and \;\; R_{11}, R_{12}, R_{13} \geqslant 0 \\ &\quad s.t. \; R_1 = R_{11} + R_{12} + R_{13} : \\ &\qquad I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13})) \\ \subset \;\; &\qquad I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13})) \\ \equiv \;\; &\qquad I(Rn_2^{(3)}(k_2, R_{11}, R_{12}, R_{13}) \times Rn_3^{(3)}(k_3, R_{11}, R_{12}, R_{13})) \end{aligned} \tag{6.27}$$

In general, the subset relations are proper subsets, but for some environments the images can be equal.

Proof

We will first prove hypothesis (6.26), by showing $I(Rn_i^{(1)}(k_i)) \subseteq I(Rn_i^{(2)}(k_i))$ and $I(Rn_i^{(2)}(k_i)) \subseteq I(Rn_i^{(3)}(k_i))$. In particular, $I(Rn_i^{(1)}(k_i)) \subseteq I(Rn_i^{(3)}(k_i))$ can be concluded from the transitivity of the relation $\subseteq$, which we will not prove here. Then, we will show that the relations are strict subsets in general. Next, we will prove hypothesis (6.27), by assuring
$I(Rn_2^{(1)} \times Rn_3^{(1)}) \subseteq I(Rn_2^{(2)} \times Rn_3^{(2)})$,
$I(Rn_2^{(1)} \times Rn_3^{(1)}) \not\supset I(Rn_2^{(2)} \times Rn_3^{(2)})$,
$I(Rn_2^{(2)} \times Rn_3^{(2)}) \subseteq I(Rn_2^{(3)} \times Rn_3^{(3)})$, and
$I(Rn_2^{(2)} \times Rn_3^{(2)}) \supseteq I(Rn_2^{(3)} \times Rn_3^{(3)})$
In order to keep this list legible, we omitted the parameters. We will continue to do so occasionally, when appropriate. In the rest of this proof, we will write $k_i^{(l)}$ to refer to the k_i in $Rn_i^{(l)}(k_i)$. For the R_{1i} we will use the analogous notation, but only when we have control over it.

Proof of $I(Rn_i^{(1)}(k_i)) \subseteq I(Rn_i^{(2)}(k_i))$:

For any arbitrary but fixed assignment of $k_i^{(1)}, R_1, R_{11}, R_{1i}$, and R_{1j} such that $R_1, R_{11}, R_{1i}, R_{1j} \geqslant 0$, $k_i^{(1)} \in [0,1]$ and $R_1 = R_{11} + R_{1i} + R_{1j}$, we can prove

$$Rn_i^{(1)}(k_i) \in I(Rn_i^{(2)}(k_i)).$$

That is, $\exists k_i^{(2)} \in [0,1] : Rn_i^{(1)}(k_i) = R_{1i} + \frac{1}{2}k_i^{(1)}R_{11} = Rn_i^{(2)}(k_i) = R_{1i} + k_i^{(2)}R_{11}$ and we can find this $k_i^{(2)}$.

$$\begin{aligned} Rn_i^{(1)}(k_i) &= Rn_i^{(2)}(k_i) \\ R_{1i} + \frac{1}{2}k_i^{(1)}R_{11} &= R_{1i} + k_i^{(2)}R_{11} \\ \frac{1}{2}k_i^{(1)}R_{11} &= k_i^{(2)}R_{11} \end{aligned}$$

Since $R_{11} \geqslant 0$, there are two cases.

Case 1: $R_{11} = 0$

Then, for any arbitrary $k_i^{(2)}$ the statement is true:

$$0 = 0$$

Case 2: $R_{11} > 0$

$$\frac{1}{2}k_i^{(1)} = k_i^{(2)} \tag{6.28}$$

Since $k_i^{(1)} \in [0,1]$ and $\frac{1}{2}k_i^{(1)} = k_i^{(2)}$, it is obvious that $k_i^{(2)} \in [0,0.5] \subset [0,1]$, where any valid setting for $k_i^{(2)}$ has to obey $k_i^{(2)} \in [0,1]$.

Proof of $I(Rn_i^{(2)}(k_i)) \subseteq I(Rn_i^{(3)}(k_i))$:
For any arbitrary but fixed assignment of $k_i^{(2)}, R_1, R_{11}, R_{1i}$, and R_{1j} such that $R_1, R_{11}, R_{1i}, R_{1j} \geqslant 0$, $k_i^{(2)} \in [0,1]$ and $R_1 = R_{11} + R_{1i} + R_{1j}$, we can prove

$$Rn_i^{(2)}(k_i) \in I(Rn_i^{(3)}(k_i)).$$

That is, $\exists k_i^{(3)} \in [0,1] : Rn_i^{(2)}(k_i) = R_{1i} + k_i^{(2)} R_{11} = Rn_i^{(3)}(k_i) = R_{1i} + k_i^{(3)}(R_{11} + R_{1j})$ and we can find this $k_i^{(3)}$.

$$\begin{aligned} Rn_i^{(2)}(k_i) &= Rn_i^{(3)}(k_i) \\ R_{1i} + k_i^{(2)} R_{11} &= R_{1i} + k_i^{(3)}(R_{11} + R_{1j}) \\ k_i^{(2)} R_{11} &= k_i^{(3)}(R_{11} + R_{1j}) \end{aligned}$$

Since $R_{11}, R_{1j} \geqslant 0$, there are two cases.

Case 1: $R_{11} = R_{1j} = 0$
Then, for any arbitrary $k_i^{(3)}$ the equality is established:

$$0 = 0$$

Case 2: $R_{11} > 0 \vee R_{1j} > 0$

$$k_i^{(3)} = k_i^{(2)} \frac{R_{11}}{R_{11} + R_{1j}} \tag{6.29}$$

We can be certain that $k_i^{(3)} \in [0,1]$: In case 1, we can choose $k_i^{(3)}$ arbitrarily, in particular within the desired interval. In case 2, we have to prove that $\frac{R_{11}}{R_{11}+R_{1j}} \in [0,1]$. However, we know: $R_{11}, R_{1j} \geqslant 0 \wedge (R_{11} > 0 \vee R_{1j} > 0)$.
If $R_{11} = 0$, then $R_{1j} > 0$ and $\frac{R_{11}}{R_{11}+R_{1j}} = \frac{0}{R_{1j}} = 0$.
If $R_{1j} = 0$, then $R_{11} > 0$ and $\frac{R_{11}}{R_{11}+R_{1j}} = \frac{R_{11}}{R_{11}} = 1$.
If $R_{11} > 0 \wedge R_{1j} > 0$, then $\frac{R_{11}}{R_{11}+R_{1j}} > 0$[3] and $\frac{R_{11}}{R_{11}+R_{1j}} < \frac{R_{11}}{R_{11}} = 1$.

[3] In the limit, $\frac{R_{11}}{R_{11}+R_{1j}} \xrightarrow{R_{1j} \to \infty} 0$.

Now that we can be certain of $\frac{R_{11}}{R_{11}+R_{1j}} \in [0,1]$ and $k_i^{(2)} \in [0,1]$, we can conclude that $k_i^{(3)} = k_i^{(2)} \frac{R_{11}}{R_{11}+R_{1j}} \in [0,1]$.

In order to show that the image sets are not completely equal, we have to show that certain values can be achieved by the one formula but not the other. It is sufficient to pick exactly one such value per pair of formulae. Since the $\not\supset$ relation is not necessarily transitive, we will conduct the proof for all three pairs.

Proof of $I(Rn_i^{(1)}(k_i)) \not\supset I(Rn_i^{(2)}(k_i))$:
For an arbitrary but fixed setting of R_{11} and R_{1i}, and a $k_i^{(2)} > 0.5$, $Rn_i^{(2)}(k_i) = R_{1i} + k_i^{(2)} R_{11}$ takes on a value that cannot be the outcome of $Rn_i^{(1)}(k_i) = R_{1i} + \frac{1}{2} k_i^{(1)} R_{11}$, with a $k_i^{(1)} \in [0,1]$.
One specific instance: Let $R_{11} = 10$, $R_{1i} = 10$, and $k_i^{(2)} = 1$. Then, $Rn_i^{(2)}(k_i) = R_{1i} + k_i^{(2)} R_{11} = 10 + 1 \cdot 10 = 20$. In order to construct a $k_i^{(1)}$, such that $20 = R_{1i} + \frac{1}{2} k_i^{(1)} R_{11} = 10 + \frac{1}{2} k_i^{(1)} 10$, $k_i^{(1)}$ would have to be 2, with violates the constraint $k_i^{(1)} \in [0,1]$. Thus, there is an element in $I(Rn_i^{(2)}(k_i))$ which is not in $I(Rn_i^{(1)}(k_i))$.

Proof of $I(Rn_i^{(2)}(k_i)) \not\supset I(Rn_i^{(3)}(k_i))$:
For an arbitrary but fixed setting of R_{11}, R_{1i}, and R_{1j}, $k_i^{(3)}$ can be set in a way that causes $Rn_i^{(3)}(k_i) = R_{1i} + k_i^{(3)}(R_{11} + R_{1j})$ to take on a value which cannot be the outcome of $Rn_i^{(2)}(k_i) = R_{1i} + k_i^{(2)} R_{11}$, with a $k_i^{(2)} \in [0,1]$.
One specific instance: Let $R_{11} = 10$, $R_{1i} = 10$, $R_{1j} = 10$, and $k_i^{(3)} = 1$. Then, $R_{1i} + k_i^{(3)}(R_{11} + R_{1j}) = 10 + 1 \cdot (10 + 10) = 30$. In order to construct a $k_i^{(2)}$, such that $30 = R_{1i} + k_i^{(2)} R_{11} = 10 + k_i^{(2)} 10$, $k_i^{(2)}$ would have to be 2, with violates the constraint $k_i^{(2)} \in [0,1]$. Thus, there is an element in $I(Rn_i^{(3)}(k_i))$ which is not in $I(Rn_i^{(2)}(k_i))$.

Proof of $I(Rn_i^{(1)}(k_i)) \not\supset I(Rn_i^{(3)}(k_i))$:
Above, we have shown that $I(Rn_i^{(1)}(k_i)) \not\supset I(Rn_i^{(2)}(k_i))$ and $I(Rn_i^{(2)}(k_i)) \subseteq I(Rn_i^{(3)}(k_i))$. Therefore, we can conclude that the specific element of $I(Rn_i^{(2)}(k_i))$ which is not in $I(Rn_i^{(1)}(k_i))$, is in $I(Rn_i^{(3)}(k_i))$. Thus, there is at least one elements of $I(Rn_i^{(3)}(k_i))$ which is not in $I(Rn_i^{(1)}(k_i))$.

Proof of $I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13}))$
$\subseteq I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$:

For each $R_1 \geqslant 0$ and an arbitrary but fixed setting of $k_2^{(1)}, k_3^{(1)} \in [0,1], R_{11}^{(1)}, R_{12}^{(1)}, R_{13}^{(1)} \geqslant 0$ with $R_1 = R_{11}^{(1)} + R_{12}^{(1)} + R_{13}^{(1)}$, we can find a valid assignment of $k_2^{(2)}, k_3^{(2)}, R_{11}^{(2)}, R_{12}^{(2)}, R_{13}^{(2)}$, such that $(Rn_2^{(1)}, Rn_3^{(1)}) = (Rn_2^{(2)}, Rn_3^{(2)})$. The assignment strategy here is rather simple: $R_{11}^{(2)} = R_{11}^{(1)}, R_{12}^{(2)} = R_{12}^{(1)}, R_{13}^{(2)} = R_{13}^{(1)}, k_2^{(2)} = \frac{1}{2}k_2^{(1)}, k_3^{(2)} = \frac{1}{2}k_3^{(1)}$. As shown above (equation (6.28) and the proofs around it), the resulting values are the same: $Rn_i^{(1)} = Rn_i^{(2)}$ for $i \in \{2,3\}$. Thus, every element in $I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13}))$ is also in $I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$

Proof of $I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13}))$
$\not\supset I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$:

In order to prove the above statement, we have to provide one element of $I(Rn_2^{(2)} \times Rn_3^{(2)})$ which is not in $I(Rn_2^{(1)} \times Rn_3^{(1)})$. Note that for $R_1 = 0$, the image sets are equal: $\{0\}$. But for a fixed arbitrary value of $R_1 > 0$, we set $k_2^{(2)} = k_3^{(2)} = 1$, $R_{11}^{(2)} = R_1$, and $R_{12}^{(2)} = R_{13}^{(2)} = 0$. This assignment satisfies the constraints $k_2^{(2)}, k_3^{(2)} \in [0,1]$ and $R_1 = R_{11}^{(2)} + R_{12}^{(2)} + R_{13}^{(2)}$. The result is $I(Rn_2^{(2)} \times Rn_3^{(2)}) = (R_1, R_1)$.
For now, let us assume that $(R_1, R_1) \in I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13}))$. In particular, that means $R_1 = Rn_2^{(1)}(k_2, R_{11}, R_{12}) = R_{12}^{(1)} + \frac{1}{2}k_2^{(1)}R_{11}^{(1)}$. Since $k_2^{(1)}$ has to be in $[0,1]$ and $R_1 = R_{11}^{(1)} + R_{12}^{(1)} + R_{13}^{(1)}$ has to be fulfilled, that means essentially that we have to set $R_{12}^{(1)} = R_1$ and thus $R_{11}^{(1)} = R_{13}^{(1)} = 0$, where $k_2^{(1)}$ can be set arbitrarily. No other possible reward splitting than the above one would satisfy $R_1 = Rn_2^{(1)}(k_2, R_{11}, R_{12})$. But then, $Rn_3^{(1)}(k_3, R_{11}, R_{13}) = R_{13}^{(1)} + \frac{1}{2}k_3^{(1)}R_{11}^{(1)} = 0 + \frac{1}{2}k_3^{(1)} \cdot 0 = 0$, specifically $R_1 > 0 = Rn_3^{(1)}(k_3, R_{11}, R_{13})$. Thus, our assumption had to be false and we can conclude that (R_1, R_1) is not in $I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13}))$.

Proof of $I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$
$\subseteq I(Rn_2^{(3)}(k_2, R_{11}, R_{12}, R_{13}) \times Rn_3^{(3)}(k_3, R_{11}, R_{12}, R_{13}))$:

For each $R_1 \geqslant 0$ and an arbitrary but fixed setting of $k_2^{(2)}, k_3^{(2)} \in [0,1], R_{11}^{(2)}, R_{12}^{(2)}, R_{13}^{(2)} \geqslant 0$ with $R_1 = R_{11}^{(2)} + R_{12}^{(2)} + R_{13}^{(2)}$, we can find a

valid assignment of $k_2^{(3)}, k_3^{(3)}, R_{11}^{(3)}, R_{12}^{(3)}, R_{13}^{(3)}$, such that $(Rn_2^{(2)}, Rn_3^{(2)}) = (Rn_2^{(3)}, Rn_3^{(3)})$. The assignment strategy here, again, is rather simple: $R_{11}^{(3)} = R_{11}^{(2)}, R_{12}^{(3)} = R_{12}^{(2)}, R_{13}^{(3)} = R_{13}^{(2)}, k_2^{(3)} = k_2^{(2)} \frac{R_{11}^{(2)}}{R_{11}^{(3)}+R_{13}^{(3)}}, k_3^{(3)} = k_3^{(2)} \frac{R_{11}^{(2)}}{R_{11}^{(3)}+R_{12}^{(3)}}$. As shown above (equation (6.29) and around it), the resulting values are the same: $Rn_i^{(2)} = Rn_i^{(3)}$ for $i \in \{2,3\}$. Thus, every element in $I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$ is also in $I(Rn_2^{(3)}(k_2, R_{11}, R_{12}, R_{13}) \times Rn_3^{(3)}(k_3, R_{11}, R_{12}, R_{13}))$.

Proof of $I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$
$\supseteq I(Rn_2^{(3)}(k_2, R_{11}, R_{12}, R_{13}) \times Rn_3^{(3)}(k_3, R_{11}, R_{12}, R_{13}))$:

For $R_1 = 0$, both image sets only contain the element $(0,0)$. Specifically, the statement is true. For an arbitrary but fixed $R_1 > 0$ and an arbitrary but fixed setting of $k_2^{(3)}, k_3^{(3)} \in [0,1], R_{11}^{(3)}, R_{12}^{(3)}, R_{13}^{(3)} \geqslant 0$ with $R_1 = R_{11}^{(3)} + R_{12}^{(3)} + R_{13}^{(3)}$, we can find a valid assignment of $k_2^{(2)}, k_3^{(2)}, R_{11}^{(2)}, R_{12}^{(2)}, R_{13}^{(2)}$, such that $(Rn_2^{(3)}, Rn_3^{(3)}) = (Rn_2^{(2)}, Rn_3^{(2)})$. The assignment strategy here is the following: $R_{11}^{(2)} = R_1, R_{12}^{(2)} = R_{13}^{(2)} = 0, k_2^{(2)} = \frac{Rn_2^{(3)}}{R_1}, k_3^{(2)} = \frac{Rn_3^{(3)}}{R_1}$. This setting of $k_i^{(2)}$ is valid, due to $Rn_i^{(3)} = R_{1i}^{(3)} + k_i^{(3)}(R_{11}^{(3)} + R_{1j}^{(3)})$ and $k_i^{(3)} \in [0,1]$. Therefore, $Rn_i^{(3)} \in [R_{1i}, R_1]$, because $R_1 = R_{11}^{(3)} + R_{12}^{(3)} + R_{13}^{(3)}$. Thus, $\frac{Rn_i^{(3)}}{R_1} \in [0,1]$. Given the assignment above, the resulting values are the same: $Rn_i^{(2)} = R_{1i}^{(2)} + k_i^{(2)} R_{11}^{(2)} = 0 + \frac{Rn_i^{(3)}}{R_1} R_1 = Rn_i^{(3)}$ for $i \in \{2,3\}$. Thus, every element in $I(Rn_2^{(3)}(k_2, R_{11}, R_{12}, R_{13}) \times Rn_3^{(3)}(k_3, R_{11}, R_{12}, R_{13}))$ is also in $I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$.

□

References

1. H. Jung, M. Tambe, and S. Kulkarni, "Argumentation as distributed constraint satisfaction: Applications and results," in *Proceedings of the International Conference on Autonomous Agents*, 2001.
2. X. Zhang, V. Lesser, and T. Wagner, "Integrative negotiation among agents situated in organizations," *IEEE Transactions on Systems, Man, and Cybernetics: Part C, Special Issue on Game-theoretic Analysis and Stochastic Simulation of Negotiation Agents*, vol. 36, no. 1, pp. 19–30, January 2006.

3. A. Glass and B. Grosz, "Socially conscious decision-making," in *Proceedings of Agents 2000 Conference*, Barcelona, Spain, June 2000, pp. 217 – 224.
4. E. Oliveira and A. P. Rocha, "Agents advanced features for negotiation in electronic commerce and virtual organisations formation processes," in *Agent Mediated Electronic Commerce, The European AgentLink Perspective.*, C. Sierra and F. Dignum, Eds. London, UK: Springer-Verlag, 2001, pp. 78–97.
5. Q. Zheng and X. Zhang, "Automatic formation and analysis of multi-agent virtual organization," *Journal of the Brazilian Computer Society: Special Issue on Agents Organizations*, vol. 11, no. 1, pp. 74–89, July 2005.
6. T. J. Norman, A. Preece, S. Chalmers, N. R. Jennings, M. Luck, V. D. Dang, T. D. Nguyen, V. Deora, J. Shao, A. Gray, and N. Fiddian, "Conoise: Agent-based formation of virtual organisations," *Int. J. Knowledge Based Systems*, vol. 17, no. 2-4, pp. 103–111, 2004.
7. S. Sen and E. H. Durfee, "A formal study of distributed meeting scheduling," *Group Decision and Negotiation*, vol. 7, pp. 265–289, 1998.
8. T. Sandholm, S. Sikka, and S. Norden, "Algorithms for optimizing leveled commitment contracts," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Stockholm, Sweden, 1999, pp. 535–540.
9. S. Sen, "Believing others: Pros and cons." in *Artificial Intelligence,*, 2002, pp. 142(2):179–203.
10. S. Saha, S. Sen, and P. S. Dutta, "Helping based on future expectations," in *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*. Melbourne, Australia: ACM Press, 2003, pp. 289–296.
11. K. Decker and V. Lesser, "An Approach to Analyzing the Need for Meta-Level Communication," *International Joint Conference on Artificial Intelligence*, vol. 1, January 1993. [Online]. Available: http://mas.cs.umass.edu/paper/29
12. J. M. Vidal, "The effects of cooperation on multiagent search in task-oriented domains," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 16, no. 1, pp. 5–18, 2004.
13. M. Klusch and A. Gerber, "Dynamic coalition formation among rational agents," *IEEE Intelligent Systems*, vol. 17, no. 3, pp. 42–47, 2002.
14. D. K. Levine, "Modeling altruism and spitefulness in experiments," *Review of Economic Dynamics*, vol. 1, pp. 593–622, 1998.
15. W. T. L. Teacy, N. R. J. J. Patel, S. C. M. Luck, N. Oren, T. J. Norman, A. Preece, P. M. D. Gray, G. Shercliff, P. J. Stockreisser, J. Shao, W. A. Gray, N. J. Fiddian, and S. Thompson, "Monitoring, policing and trust for grid-based virtual organisations," in *Proc. 4th UK e-Science Meeting, Nottingham*, 2005.
16. J. Shen, X. Zhang, and V. Lesser, "Degree of Local Cooperation and its Implication on Global Utility," *Proceedings Of Third International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2004)*, July 2004.
17. M. H. Chia, D. E. Neiman, and V. R. Lesser, "Poaching and distraction in asynchronous agent activities," in *Proceedings of the Third International Conference on Multi-Agent Systems*, 1998, pp. 88–95.
18. V. R. Lesser and L. D. Erman, "Distributed interpretation: A model and an experiment," vol. C-29, no. 12, pp. 1144–1163, Dec. 1980.
19. S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient Methods. Stanford University, October 1, 2003." [Online]. Available: http://www.stanford.edu/class/ee392o/subgrad_method_slides.pdf
20. A. Nedic and D. Bertsekas, "Incremental Subgradient Methods for Nondifferentiable Optimization," *Report LIDS-P-2460, Dec. 2000, SIAM*

J. on Optimization, Vol. 12, pp. 109–138, 2001. [Online]. Available: http://web.mit.edu/dimitrib/www/Increm_LIDS.pdf

21. I. Weber, J. Shen, and V. Lesser, "Modeling and analyzing cooperation parameters in a multi-agent system," Computer Science Department, University of Massachusetts, Amherst, Tech. Rep. 05-29, 2005.