Modeling and Analyzing Cooperation Parameters in a Multi-Agent System

Ingo M. Weber, Jiaying Shen, and Victor Lesser
University of Massachusetts
Deparment of Computer Science
140 Governor's Drive
Amherst, MA 01003-4601
{imweber, jyshen, lesser}@cs.umass.edu

February 2005

Abstract

In a multi-agent system, an environment imposes tasks on a group of agents. Upon successful completion, the agents get rewarded. Any interesting environment requires the agents to decide, which tasks they want to work on. By modeling a multi-agent system statistically and exploring the model, we can gain deep insights into the way such decisions should be made.

In preliminary work, a statistical model for a three-agent system has been developed. Two of the agents receive tasks from the environment, that require only their attention. The third agent, however, receives tasks that need the collaboration of all the three agents.

When the model was examined, there were three major simplifications: the reward for the shared task was divided between the participants by fixed ratios, an agent could only account for the other agents' gains in a certain way, and decommitment from a shared task was not communicated explicitly. This work analyzes how the mentioned simplifications impact the rewards. It also takes a close look at the statistical model and the structure of the resulting functions. A bi-product of the analysis is the conclusion that traditional approaches to take the other agents' rewards into account are in general suboptimal.

1 Introduction

In a multi-agent system several autonomous artificial agents work in a common environment. The environment imposes tasks on the agents, and by completing them the agents receive a reward. Certain tasks can be successfully fulfilled only by the joint action of a group of agents. Since there is a lot of concurrency, there can be complicated interaction between the tasks and agents.

Figure 3 shows an example multi-agent system, which will concern us throughout the rest of this work. It consist of three agents, A_1, A_2, and A_3. The environment imposes one type of task on each one of them, referred to as T_1, T_2, and T_3, respectively. While T_2 and T_3 are handled by A_2 and A_3 alone, agent A_1 requires their help on T_1: task T_1 consists of two subtasks, sub_2 and sub_3, which both have to be completed in order to successfully fulfill T_1. A_1, however, does not have the abilities or resources to work on any of the subtasks and needs to contract them out to the other two agents. sub_2 can only be executed by A_2, and sub_3 only by A_3.

When a task is successfully completed, the agent who received it from the environment gets rewarded. In general, this reward can be randomly distributed, but the agent knows the exact reward it will gain by the time it receives an instance of the task. A_1 has to decide how much to pay A_2 and A_3 for working on the subtasks. A_2 and A_3, on the other hand, can take into consideration how much A_1 gains in their decisions.

Every agent has a goal. For example, the goal can be that it gains as much as possible without caring about the other agents, in which case we call it *self-interested*. The other extreme is when it is only concerned about the other agents' gains and not at all about its own. Then the agent is called *altruistic*. Between these extremes there is a range of partial self-interest and partial altruism. Exactly in the middle are agents who try to maximize the combined gains of all of the agents. This could be the case in a company or a virtual organization. These the agents are called *cooperative*. Here, we are most interested in cooperative agents, but describe extensions to differing goals where applicable.

Thereby, our multi-agent system can be formulated as an *optimization problem*. In the case of a fully cooperative system, we want to maximize the social utility, i.e. the sum of the individual gains of the agents. Given a certain environment of tasks and their parameters, such as task duration, reward, and more, the agents work together to achieve the highest possible

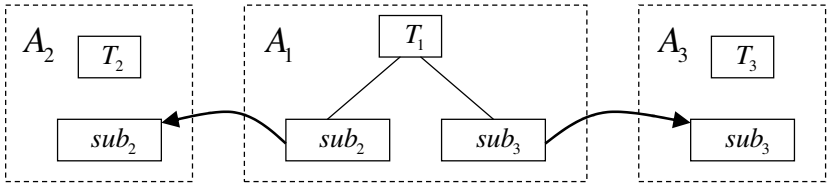

Figure 1: An example for a multi-agent system

sum of rewards.

Since every agent decides locally which tasks to take on and which ones to refuse, it has to have sufficient knowledge in order to make the optimal decisions. In the situation where a task which just arrived seems more beneficial than one that is already scheduled, but due to time constraints only one of them can be accomplished, the respective agent may decide to drop the task that is scheduled and instead schedule the new one. If the dropped task is part of a shared task, the other agents who work on the shared task are affected by this action, since they will not get their share of the reward. This issue falls into a class of problems called *multi-linked negotiation*, on which there has been some recent research [6, 3, 2].

In order to balance out the impacts of multi-linked negotiation and other complex interactions, the notion of *integrative negotiation* has been developed [7]. The idea in terms of our example system is the following. When agent A_1 requests A_2 and A_3 to perform sub_2 and sub_3, it tells them how much reward they will get and how much it itself will gain upon successful completion. A_2 and A_3 consider the reward of A_1 partially as their own in their decisions by adding the A_1's reward times the *local attitude parameter* to their local reward for the comparison. The factor with which they weigh A_1's local reward is also called the *degree of local cooperation*.

Based on the degree of local cooperation, a statistical model for the system has been developed and evaluated [5]. The main focus of this model was to find the optimal local attitude parameters and to provide the agents with a local mechanism to do so. The three major simplifications in the model were

- a fixed reward splitting for the shared task T_1,
- a negotiation protocol without explicit decommitment, and
- a single way of performing the integrative negotiation in terms of the

meta-level information that is transmitted.

The first statement means, that A_1 did not reason about how much of the reward it should hand out to A_2 and A_3 in terms of the optimization. Rather, it kept paying A_2 and A_3 a fixed rate of task T_1's reward without adaptation to different environments. In this work we will show that, in fact, it is potentially beneficial to reason about the impacts of reward splitting. In certain environments, a cooperative system can achieve higher gains by choosing the rewards for the subtasks systematically.

In [5], only implicit decommitment was used. That is, if an agent decided to drop the subtask it previously committed to, it simply let the deadline pass without sending a completion message to the other agents. A_1 would not receive any reward for this instance of T_1 and, in turn, not pay A_2 and A_3. In this way, the other agents learned implicitly that one of them decommitted from the task. Explicit decommitment would be if an agent, who decides to drop the subtask it scheduled, sends a decommitment message to the other agents. Thus, they can potentially free the time slot they assigned to their part of the shared task and schedule other tasks.

The third point means that, when an agent decided whether or not to accept a subtask, it took into account the local reward it would receive and A_1's full reward. From a global perspective, one could say that the reward of A_1 should not be accounted for twice, thus A_1 should tell both A_2 and A_3 only about half of its reward. On the other hand, a local point of view suggests that each agent should know about the importance of a task to the social utility, and thus an agent should know how much it will be paid and how much all the other agents will gain if it finishes the subtask. This leads to transmitting slightly different information as part of the negotiation policy.

In this work, we are concerned about how these three simplifications effect the optimality of the model. In order to evaluate this, we first look into the structure of the statistical model and see, how one can find the highest value of its output, i.e. the maximal expected utility. Then, we analyze the effects of optimizing this model in terms of the reward splitting for the shared task T_1, and find out that, in fact, certain situations yield higher rewards than can be realized with the fixed reward splitting strategy from [5]. Next, we examine whether altering the information in the integrative negotiation or extending it towards more flexibility impacts the expected utilities significantly. This is related to broadening the scope of uncertainty

about the reward payment for the subtasks, as well as allowing the system to overrate the benefits of a shared task. The increased flexibility indeed pays off in certain environments. We then present possible extensions to the statistical model for adding explicit decommitment, and the according preliminary effects on the maximal social utility. In the end we give an outlook on ideas for future work within and around this multi-agent system.

The rest of this paper is organized in the following way. Section 2 is a summary of [5]. In Section 3, we present a mathematical analysis of the statistical model. Section 4 is concerned with the reward splitting and its effects, while Section 5 looks into the different ways to perform and enhance the integrative negotiation. The extensions of the statistical model for explicit decommitment are introduced and evaluated in Section 6. Section 7 concludes and suggests directions of further research on this problem. Appendix A contains the parameters for the scenarios used throughout the course of this work.

2 A Summary of Preliminary Work

This paper extends the work in [5], which will be introduced in the following section.

2.1 A Formal Model of a Multi-Agent System

As described in [5], a multi-agent system consists of several agents, each receiving tasks from the environment. A task is defined by its arrival time, earliest start time, duration, deadline, and reward, as well as its internal structure: a task can contain several other tasks. We will call a task with no subtasks a low-level task, and one with subtasks a high-level task. Low-level tasks that the environment imposes are called local tasks, while local tasks from other agents are subtasks, a part of a shared task. Shared tasks yield utility only if all of its subtasks are successfully accomplished. Although several layers of high-level tasks can be imagined - i.e. a subtask being a high-level task at the same time - [5] uses the notion of one layer of high-level tasks only, and we will stick to that.

The group of agents, $A_1, A_2, ..., A_l$ is characterized by their abilities: Each agent can work on task solely if it is met by its abilities. That includes the decomposition of a shared task into subtasks and contracting these out to

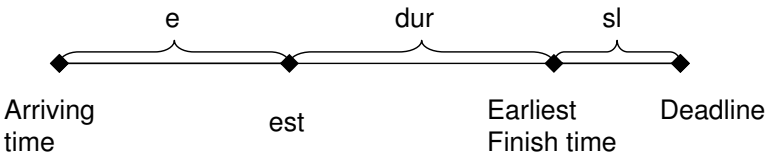

Figure 2: The relationship of different parameters of a task

other agents.

A task T_i with $i \in \{1, 2, ..., n\}$ is determined by a set of parameters. These can be probabilistically distributed for each task and are

- r_i: task T_i arrives at an agent at time t with a probability of $1/r_i$.
- e_i: the difference between the arrival time of a task T_i and its earliest start time est_i.
- dur_i: the duration of the task T_i.
- sl_i: the difference between the earliest possible finish time of a task T_i and the deadline dl_i.
- R_i: the reward of a task T_i if it's finished.

The relationship of e_i, est_i, dur_i, sl_i and dl_i is illustrated in Figure 2.

The high-level tasks can be decomposed into a set of subtasks, $T_{i1}, T_{i2}, ..., T_{im_i}$, for which we will use an analogous notation. There are some relations between the subtasks among each other and with their corresponding high-level task.

- r_{ij}: $r_{ij} = r_i$.
- e_{ij}: the difference between the arrival time of a subtask T_{ij} and its earliest start time est_{ij}.
- dur_{ij}: the duration of the subtask T_{ij}.
- sl_{ij}: the difference between the earliest possible finish time of the subtask T_{ij} and its deadline dl_{ij}.
- R_{ij}: the reward of the subtask T_{ij} if it is finished. $\sum_j R_{ij} + R_{i0} = R_i$, where R_{i0} is the reward A_s gets after handing out the rewards to each subtask if all of the subtasks are completed.

Note that, using this framework, we can force sequentiality of subtasks.

2.2 Integrative Negotiation

When contracting out subtasks, in addition to the parameters above the agent sends its own utility increase along. This piece of meta-information is called relational reward and can be used by contractee agents to evaluate how important the whole task is for the contractor. This notion is called *integrative negotiation*. According to [7], an agent who accounts for the utility of the other agent in the decision process as if it were its own is considered fully externally-directed, whereas one that does not care about the other agent's gains is called self-directed. Note, however, that the agent never gets paid the relational reward. It is only used for reasoning.

In contrast to the notion of cooperative and self-interested agents, which describes their objectives, self-directed and externally directed are mechanisms in the negotiation and decision process. Although these terms are related, they are far from being equal, as shown in [7].

2.3 The Example Multi-Agent System

The system we use for our statistical analysis consists of three agents, A_1, A_2, and A_3. All of them have local tasks coming in: T_1, T_2, and T_3, respectively. T_2 and T_3 are low-level tasks and T_1 is a shared task with the subtasks T_{12} and T_{13} (or sub_2 and sub_3) which can only be executed by A_2 and A_3. This system is shown in Figure 3.
It has multi-linked negotiation, because the decision of A_2 on committing to T_{12} affects A_3's utility gains as well as potentially its optimal decisions. If A_2 committed to T_{12} initially, but then a task T_2 arrives and seems more promising to A_2, it decides to decommit from T_{12} in order to do the other task.

Decommitment can happen in one of two ways: implicitly or explicitly. If it happens implicitly, the agent who decommits simply does not report that it is done with the task at any point in time. Only this way is considered in [5]. If an agent decommits explicitly by sending a decommitment message, the other agents can potentially empty the scheduled time slots and take on other incoming tasks, due to the gap in the schedule. However, this is not always the case, since the decommitment message might arrive after the agent already discarded a task that could have fit into the part of the schedule now freed up. In either case, there is some uncertainty in whether the reward of a shared task will be paid, and this uncertainty is introduced by agents

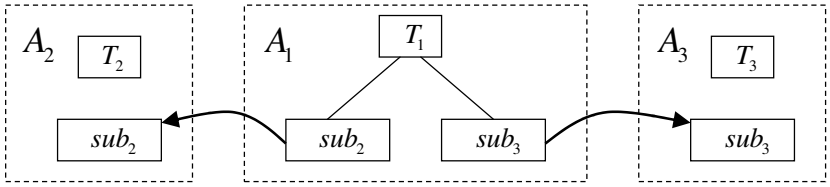

Figure 3: The simplest organization structure with the necessary inter-agent interactions. Repeated from Figure 1.

without a direct connection. In Section 6, we broaden our view to allow for explicit decommitment.

2.4 A Statistical Model for the Three-Agent System

In this section the model developed in [5] is described. It captures all the dynamics in the three-agent organization of interest. We analyze the modeled function in Section 3 and extend it in later sections.

2.4.1 Parameter Settings

As mentioned in Subsection 2.1, the external parameters like duration, earliest start-time, or similar can be randomly distributed. Here, we assume unform distribution within the interval $(a_i^y, b_i^y]$ for some parameter y_i. E.g. for the duration, dur_i, of task T_i, the probability distribution is given by:

$$P(dur_i = x) = \begin{cases} \frac{1}{b_i^d - a_i^d}, & a_i^d < x \leqslant b_i^d \\ 0, & \text{otherwise} \end{cases}$$

In this sense, for any instance of task T_i, the parameters are drawn from the uniform distributions as follows: $e_i \in (a_i^e, b_i^e], dur_i \in (a_i^d, b_i^d]$, and $sl_i \in (a_i^s, b_i^s]$.

Note, that T_i can also be one of the subtasks. However, the values for them have to fit the ones of the higher-level task, T_1. E.g. $est_{1j} \geqslant est_1$, $dl_{1_j} \leqslant dl_1$, and more.

Also, R_2 and R_3, the rewards for T_2 and T_3 respectively, are uniformly distributed over $(a_i^r, b_i^r]$. R_1 is kept fixed, since this is crucial for the feasibility of the statistical model. It is not that much of a loss in generality, since

the utility is a somewhat abstract value and thus rather the ratio between rewards is interesting. Any useful ratio can still be established, only R_1 is not probabilistic.

The reward splitting for the shared task has two constraints with it: $R_1 = R_{11} + R_{12} + R_{13}$ and $\forall j : R_{1j} \geqslant 0$. Besides that, it is a choice of A_1. For the time window of interest, we assume an arbitrary but fixed splitting that satisfies the conditions mentioned above.

The integrative negotiation can be done in several ways, as pointed out in Subsection 5.1. In [5], the relational reward is R_{11} and integrated using the local attitude parameter, k_i, which expresses the degree of local cooperation. Thus, an agent A_i uses $Rn_i = R_{1i} + k_i R_{11}$ as utility estimate for subtask T_{1i} in its decision process. That is, if $Rn_i > R_i$, it commits to T_{1i}. When we broaden the view on the calculation of Rn_i later on, the formulae in the next section persist.

The reward for the shared task, R_1, has to be split among the agents. [5] did not deal with this issue and a fixed splitting was used. That is, both A_2 and A_3 get paid approximately $\frac{1}{8}$ and A_1 keeps $\frac{3}{4}$. In Section 4, we will address the effects of choosing the splitting more carefully.

The following passages are mostly quoted from [5].

2.4.2 Probability of Conflict

An agent needs to choose between tasks to execute only when there is a conflict between tasks. A task of type i is in conflict with a task of type j (whether it comes 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:

$$\begin{aligned} dl_i - est_j &\leqslant dur_i + dur_j, \\ dl_j - est_i &\leqslant dur_i + dur_j. \end{aligned} \tag{1}$$

Intuitively, (1) means that the tasks i and j cannot be both fit onto the schedule and satisfy the earliest start time and deadline restrictions. Rewriting (1) in terms of *est*, *dur* and *sl*, we get

$$sl_i - dur_j \leqslant est_j - est_i \leqslant dur_i - sl_j \tag{2}$$

For a task of type i that arrives at a given time, we define $Pc_{i,j}$ as the probability of there being a task of type j that has conflict with it. Notice

that for task i, we only know of its arriving time, not its other relevant parameters. In addition, we do not know any parameter of task j.

$$\begin{aligned} Pc_{i,j} &= P(sl_i - dur_j \leqslant est_j - est_i \leqslant dur_i - sl_j) \\ &= \sum_{z=-\infty}^{+\infty} \sum_{y=z}^{+\infty} \sum_{x=z}^{y} P_{est_j - est_i}(x) P_{dur_i - sl_j}(y) P_{sl_i - dur_j}(z) \end{aligned} \tag{3}$$

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 $P(est = t)$:

$$\begin{aligned} P(est = t) &= \sum_{x=-\infty}^{+\infty} Pa(t - x) P_e(x) = \sum_{x=a^e+1}^{b^e} \frac{1}{r} \cdot \frac{1}{b^e - a^e} \\ &= \frac{1}{r} \end{aligned} \tag{4}$$

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

$$\begin{aligned} P_{est_j - est_i}(x) &= \sum_{y=-\infty}^{+\infty} P_{e_i}(y) P(est_j = y + x) = \sum_{y=a_i^e+1}^{b_i^e} \frac{1}{b_i^e - a_i^e} \cdot \frac{1}{r_j} \\ &= \frac{1}{r_j} \end{aligned} \tag{5}$$

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

$$\begin{aligned} P_{dur_i - sl_j}(y) &= \sum_{x=-\infty}^{+\infty} P_{dur_i}(x) \cdot P_{sl_j}(x - y) \\ &= \begin{cases} \frac{b_i^d - a_j^s - y}{(b_i^d - a_i^d)(b_j^s - a_j^s)}, & \max(a_i^d - a_j^s, b_i^d - b_j^s) < y < b_i^d - a_j^s; \\ \frac{1}{b_i^d - a_i^d}, & a_i^d - a_j^s \leqslant y \leqslant b_i^d - b_j^s; \\ \frac{1}{b_j^s - a_j^s}, & b_i^d - b_j^s \leqslant y \leqslant a_i^d - a_j^s; \\ \frac{b_j^s + y - a_i^d}{(b_i^d - a_i^d)(b_j^s - a_j^s)}, & a_i^d - b_j^s < y < \min(a_i^d - a_j^s, b_i^d - b_j^s); \\ 0, & otherwise. \end{cases} \end{aligned} \tag{6}$$

Similarly, we get

$$P_{sl_i-dur_j}(z) = \begin{cases} \frac{b_i^s - a_j^d - z}{(b_i^s - a_i^s)(b_j^d - a_j^d)}, & \max(a_i^s - a_j^d, b_i^s - b_j^d) < z < b_i^s - a_j^d; \\ \frac{1}{b_i^s - a_i^s}, & a_i^s - a_j^d \leqslant z \leqslant b_i^s - b_j^d; \\ \frac{1}{b_j^d - a_j^d}, & b_i^s - b_j^d \leqslant z \leqslant a_i^s - a_j^d; \\ \frac{b_j^d + z - a_i^s}{(b_i^s - a_i^s)(b_j^d - a_j^d)}, & a_i^s - b_j^d < z < \min(a_i^s - a_j^d, b_i^s - b_j^d); \\ 0, & otherwise. \end{cases} \tag{7}$$

Now, we can put (5), (6) and (7) back to (3) 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 probably 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.

2.4.3 Expected Reward

What we are really concerned about is the expected reward that the system may receive at any given time. Multiplying it by the time that the system has run yields the expected reward of the system. This is an approximation, since we are not accounting for setup times in the beginning or half-finished tasks in the end. As the time horizon gets larger, these boundary conditions get negligible. Thus, this model is most applicable for long-lived systems.

For A_2 and A_3, there may be two types of tasks coming in at any moment: the local task T_i with a probability of $1/r_i$ and the non-local task sub_i with a probability of $1/r_1$. 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 sub_i 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., $R_i > Rn_i$. Therefore,

$$E(R_i | R_i > Rn_i) = \sum_{x=\lfloor Rn_i \rfloor + 1}^{b_i^r} P_{R_i}(x) \cdot x$$

$$= \begin{cases} \frac{a_i^r+b_i^r+1}{2}, & \lfloor Rn_i \rfloor < a_i^r; \\ \frac{(b_i^r-\lfloor Rn_i \rfloor)(b_i^r+\lfloor Rn_i \rfloor+1)}{2(b_i^r-a_i^r)}, & a_i^r \leqslant \lfloor Rn_i \rfloor < b_i^r; \\ 0, & \lfloor Rn_i \rfloor \geqslant b_i^r. \end{cases} \quad (8)$$

The part of expected reward gained by executing the new task in this case is then:

$$ER_i^{(1)} = Pc_{1i,i} \cdot (1 - Pc_{i,i}) \cdot E(R_i | R_i > Rn_i) \quad (9)$$

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:

$$ER_i^{(2)} = (1 - Pc_{1i,i}) \cdot Pc_{i,i} \cdot [E(R_i | R_i > R_i') + \frac{1}{2} E(R_i | R_i = R_i')] \quad (10)$$

where

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

and

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

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.

$$\begin{aligned} ER_i^{(3)} &= Pc_{1i,i} \cdot Pc_{i,i} \cdot [E(R_i | R_i > Rn_i \& R_i > R_i') \\ &\quad + \frac{1}{2} E(R_i | R_i > R_{ni} \& R_i = R_i')] \end{aligned} \quad (11)$$

where

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

and

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

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

$$ER_i^{(4)} = (1 - Pc_{1i,i})(1 - Pc_{i,i}) \cdot \frac{a_i^r + b_i^r}{2} \tag{12}$$

Similarly, when a subtask sub_i arrives at A_i, A_i 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. Therefore the expected reward will be:

$$ER_i^{(5)} = R_{1i} \cdot Pcommit_2 \cdot Pcommit_3 \tag{13}$$

where

$$\begin{aligned}Pcommit_i &= Pc_{1i,i}(1 - Pc_{1i,1i})P(Rn_i \geqslant R_i) + \frac{1}{2}Pc_{1i,i} \cdot Pc_{1i,1i}P(Rn_i \geqslant R_i) \\ &\quad + \frac{1}{2}(1 - Pc_{1i,i})Pc_{1i,1i} + (1 - Pc_{1i,i})(1 - Pc_{1i,1i}) \end{aligned} \tag{14}$$

and

$$P(Rn_i \geqslant R_i) = \sum_{x=a_i^r+1}^{\lfloor Rn_i\rfloor} P_{R_i}(x) = \begin{cases} 1, & \lfloor Rn_i\rfloor \geqslant b_i^r \\ \frac{\lfloor Rn_i\rfloor-a_i^r}{b_i^r-a_i^r}, & a_i^r \leqslant \lfloor Rn_i\rfloor \leqslant b_i^r \\ 0, & \lfloor Rn_i\rfloor \leqslant a_i^r \end{cases} \tag{15}$$

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

$$ER_i = \frac{1}{r_i}(ER_i^{(1)} + ER_i^{(2)} + ER_i^{(3)} + ER_i^{(4)}) + \frac{1}{r_1}ER_i^{(5)} \tag{16}$$

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,

$$ER_1 = \frac{1}{r_1} \cdot R_{11} \cdot Pcommit_2 \cdot Pcommit_3 \tag{17}$$

Now that we have the expected reward for each of the agents, we can calculate the k_i that will maximize the social utility given the set of the parameters. More formally, we set k_2 and k_3 to be:

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

Please notice that when we calculate the expected reward collected by each agent from executing T_1 we assume perfect knowledge of the other agent's model. This is useful from a system designer's perspective. Having a global view of the system, the designer can set the attitude parameter k of each agent such that the global utility is maximized.

3 Analysis of the Statistical Model

When a certain 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 from the preceding section, is determined by a set of environmental parameters: These are the limits on the uniform distributions of the task arrival time, earliest start time, duration, deadline, and reward. The local parameters that can be controlled are 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 utility in each agent. In this section, we will examine the function that describes this mapping. It is a step function, that is not convex and also discontinuous, thus forbidding to use of usual optimization algorithms. Agents in general have a goal that is related to the reward. It can be to maximize the reward for all of the agents, in which case they are cooperative. There are other goals, e.g. the optimization in terms of the local reward. However, 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 will analyze the structure of the

function and develop the math that is necessary for a seamless sampling grid, therefore enabling the numerical optimization process.

3.1 Objective Functions

The goal function of an agent depends on its objectives. If it is interested in maximizing the social utility, then the overall expected value of the joint gains is to be maximized. If this is the case for each of the agents, i.e. they are all fully cooperative, then the expected social utility is the only function of interest.

However, agents can be self-interested to a certain degree. For this case, an agent A_i can have an objective function like the following one:

$$O_i = w_i E_i + (1 - w_i)(\sum_{j \in Agents} (E_j) - E_i)$$

That is, it weights its own expected utility by w_i and the rest of the social utility by $1 - w_i$. For a fully cooperative system one would use $w_i = 0.5$, meaning that it is interested in its own utility to the same degree as in anybody else's. A completely self-interested agent would set $w_i = 1$, which means that is not interested in the local utility of other agents at all. An agent with $w_i = 0$ is altruistic and considers the gains of the other agents only. In every case where one of the agents has a $w_i \neq 0.5$, the optimization of the full system requires a maximization of three intertwined functions at the same time. In general, there is no trivial solution to that, but using game theory's Nash equilibria one should be able to obtain a maximal solution.

3.2 Influences of Local Parameters

Given a set of environmental parameters, there are only a few points where the local control parameters influence the expected utility of the agents. Rn_i and R_{1i} are the only factors and summands where they come into play. The following locations relate to them, although in most cases only within certain bounds.

- $ER_i^{(1)} \sim \lfloor Rn_i \rfloor^2$
- $ER_i^{(3)} \sim \lfloor Rn_i \rfloor^2$
- $ER_i^{(5)} \sim R_{1i} \lfloor Rn_2 \rfloor \lfloor Rn_3 \rfloor$

- $ER_1 \sim R_{11}\lfloor Rn_2 \rfloor \lfloor Rn_3 \rfloor$

Due to 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.

In contrast to [5] we look at different ways to set Rn_i (see Section 5). The shape of the function depends on that along each dimension. However, each of the variations has the form

$$Rn_i = R_{1i} + k_i x \ .$$

For both pairs, (k_2, k_3) and (R_{12}, R_{13}) the analysis is analogous and we will outline it only with respect to one of the agents, A_i, where we will refer to A_j as being the other agent. In other words: $i, j \in \{2, 3\}, i \neq j$.

3.2.1 Structure of Varying the Local Attitude Parameter k

When we look at the expected utilities as a function of k_i, we have a step function with a step size of $\frac{1}{x}$, which is the case due to the floor function in $\lfloor Rn_i \rfloor$. All R_{1i} are assumed to be fixed, so this is the only factor describing the structure. Furthermore, k_i's and k_j's steps are independent of each other. That means that the two-dimensional function over both k_i and k_j has rectangular tiles of equal values.

To evaluate the function completely, it is sufficient to sample its values with a frequency of at least $\frac{1}{x}$. If we want to find a sampling frequency that is sufficient for all possible splittings of R_1, we know that x is bounded by R_1 for all the cases of interest and can use $\frac{1}{R_1}$ as the frequency.

3.2.2 Structure of Varying the Reward Splitting for the Shared Task

In the case of reward splitting, the functions' structure is a little bit more tricky. Since there is still the floor function, there are steps. But the R_{1i} part is no longer fixed, so the steps from the different agents' expected utility functions are tilted and do not have the same value. That is, the expected reward of agent A_i has the linear factor R_{1i} in it.

However, in the social utility function all of the agents' expected utilities get summed up. Since $R_1 = R_{11} + R_{12} + R_{13}$ and the factor of the R_{1i} is $\frac{1}{r_1}$ for all agents, these steps have the same value again and are *not* tilted for any given setting of external parameters.

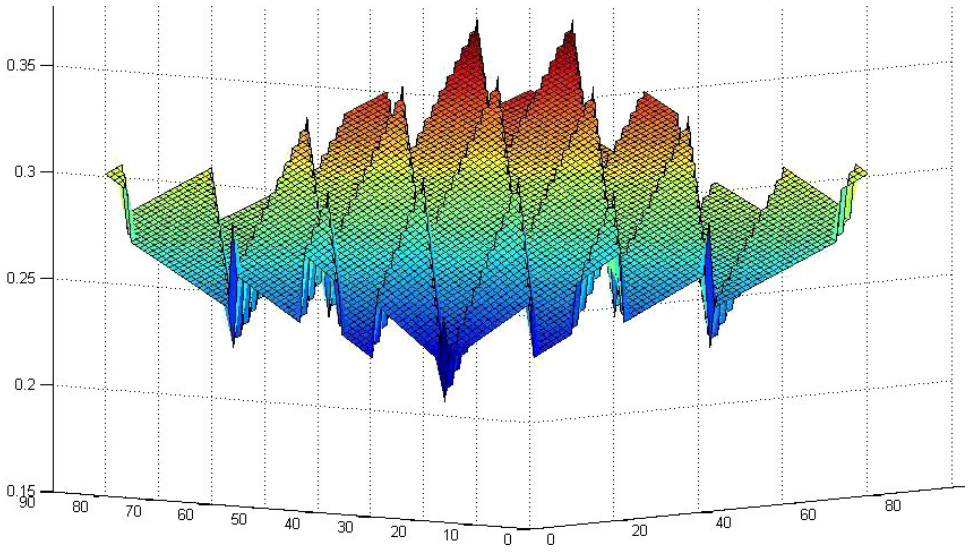

Figure 4: A closeup of what ER_1 along the variations in R_{12} and R_{13} looks like in a certain setting. $k_2 = k_3 = 0.4$

Knowing this, we can conclude that there is the potential for optimization based on the reward splitting when the agents have objective functions that differ from the social utility. That is, the optimal value for one agent will be on an edge or a corner of some tile, not in an arbitrary point of it. The variation within a tile can be as much as 36% of the optimal value from what we have seen in our experiments, but could even be more in other situation. Thus, one should be very careful about the reward splitting if not all of the agents are fully cooperative. Figure 4 shows an example for that.

In Figure 5, the step functions are illustrated. Both Rn_2 and Rn_3 introduce steps into the graph. The steps are like a staircase in top view, parallel and equidistant. When summed up, they form tiles of diamond shape.

Unlike the tiles when varying the local attitude parameter, the tiles here are not necessarily rectangular. This results from the relationship between R_{12} and R_{13}, which is: $R_{1i} = R_1 - R_{11} - R_{1j}$. Here, we look at one of the ways to compute Rn_i only: $Rn_i = R_{1i} + k_i R_{11}$. In 5.1, several different ways are presented.

To show how the relationship translates into the graph, we express the

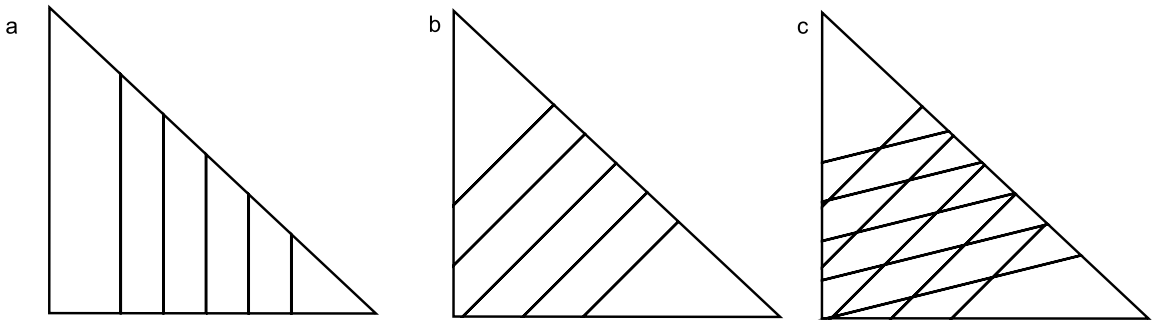

Figure 5: Shape of the steps. All the steps occur in certain bounds only. For one Rn_i they are equidistant and parallel. Graph a shows the steps for $k_2 = 0$. In b they are twisted by 45, referring to $k_2 = 0.5$. Finally, c shows the result of overlaying both functions with $k_i = 0.4$.

ratio of changes in R_{1i} and R_{1j}.

$$\begin{aligned} Rn_i &= R_{1i} + k_i R_{11} \\ &= k_i R_1 + (1 - k_i) R_{1i} - k_i R_{1j} \\ &= k_i (R_1 + (\frac{1}{k_i} - 1) R_{1i} - R_{1j}) \end{aligned}$$

Finding the steps means looking for the direction of lines of equal values, that is where the differences sum to zero. We will denote the differences in a reward R as $\triangle R$. In the following, we assume k_i, k_j and R_1 as constant.

$$\begin{aligned} 0 &= k_i (\triangle R_1 + (\frac{1}{k_i} - 1) \triangle R_{1i} - \triangle R_{1j}) \\ &= (\frac{1}{k_i} - 1) \triangle R_{1i} - \triangle R_{1j} \\ \frac{\triangle R_{1j}}{\triangle R_{1i}} &= \frac{1}{k_i} - 1 \end{aligned} \tag{18}$$

Due to $k_i \in [0, 1]$, this fraction varies between ∞ and 0. That is, the line along which the steps occur can take any slope between $0°$ and $90°$. Besides knowing the slope of the steps, we need to know the distance between the steps and the distance from the origin to the first step to describe the function completely.

Since $Rn_i = k_i R_1 + (1 - k_i)R_{1i} - k_i R_{1j}$ and, in the direction of R_{1i}, R_1 and R_{1j} are fixed, the floor function introduces steps with distances of $\frac{1}{1-k_i}$ in this direction.

$$\triangle R_{1_i} = \frac{1}{1 - k_i} \tag{19}$$

The initial offset is the R_{1i} where the first step happens, which we will denote as $\overline{R_{1i}}$. In the origin, both R_{1i} and R_{1j} equal 0. Thus, the initial value is $\lfloor k_i R_1 \rfloor$. In order to make the next step, Rn_i has to be at least 1 bigger. The first step occurs when

$$\begin{aligned} Rn_i &= \lfloor k_i R_1 \rfloor + 1 \\ &= k_i R_1 + (1 - k_i)\overline{R_{1i}} \\ \overline{R_{1i}} &= \frac{1}{1 - k_i}(\lfloor k_i R_1 \rfloor + 1 - k_i R_1) \end{aligned} \tag{20}$$

Using equations (18), (19), and (20), one can calculate the position of the maximum point of each tile and sample in these locations. Doing so assures that the global maximum is not missed due to coarse sampling. As mentioned above, this is only necessary, when not all of the agents are fully cooperative. However, when they all yield to maximize the social utility, we can overlay a grid making sure that each tile is sampled at least once, achieving the same guarantees with easier means.

3.3 Optimization Over the Resulting Space

In contrast to sampling the function, we thought about applying a known optimization algorithm to the problem. In the case of a continuous function a simple gradient ascent could work. The above analysis clearly shows that, in general, the function has steps and is not continuous.

There is an algorithm for the optimization of non-continuous functions that works similar and is called the subgradient method [1, 4]. Yet, it cannot be applied since the function is required to be convex. Again, this cannot be guaranteed in our case.

Due to the discreet nature of the 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.

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 $\lfloor Rn_i \rfloor$ 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.

In this section, we analyzed the structure of the mapping from local control parameters to the utilities of the agents. We looked at how different objective functions can be expressed in the context of the statistical model from Section 2. When all agents intend to maximize the social utility, the mapping is a four-dimensional step function, where the position of the steps in one dimension is independent of the others, except for the ones in the (R_{12}, R_{13})-space. We provided the frequencies of the steps as well as the position and angles of the diamonds that underly the (R_{12}, R_{13})-space. With all of this information it is possible to sample the four-dimensional space in discreet locations and be certain that the maxima will not be missed.

4 Varying the Reward Splitting and its Effects

The reward for a shared task is paid to the agent who gets the task from the environment. It is this agent's decision how much to offer the other agents for accomplishing their subtasks. Of course, whether the potential contractee agents find the subtask worthwhile doing is based on this reward. We call this issue the *reward splitting*.

In our example three-agent system from Figure 3, the shared task T_1 is imposed on A_1, who then negotiates with A_2 and A_3 over the subtasks T_{12} and T_{13}. Agent A_1 has to figure out how much of the overall rewards 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.

In [5], the reward splitting was fixed to the following ratios: $R_{11} = \frac{3}{4}R_1$ and $R_{12} = R_{13} = \frac{1}{8}R_1$. This section will deal with the impacts of choosing an optimal reward splitting depending on the environmental parameters. Here we will use the original way to calculate the relational reward, Rn_i, which is $Rn_i = R_{1i} + k_i R_{11}$. Section 5 introduces other such ways and evaluates them. Furthermore, we assume that the common goal of all agents is the maximization of the social utility.

During the course of this section, we will see that careful and systematic reward splitting in fact is beneficial and can lead to higher expected utility. We will also present a canonical solution to the reward splitting problem that is guaranteed to yield the maximal social utility.

4.1 Optimality Graphs

Throughout the section, any reward splitting of interest will be expressed in terms of R_{12} and R_{13}. This is sufficient because R_{11} can be calculated from them as $R_{11} = R_1 - R_{12} - R_{13}$, where R_1 is fixed in a given environment. Thus, when we illustrate a statement in the space of reward splittings, the respective graphs are two-dimensional, with R_{12} and R_{13} as the two dimensions; they are discreet, since we are sampling in a regular grid, as justified in Section 3; and the possible values are in a triangular shape due to the existence of an upper bound on the sum: $R_{12} + R_{13} \leqslant R_1$.

Within a specific environment, there is a maximal expected utility for all settings of k_2, k_3, R_{12}, and R_{13}. We will denote any setting of these four variables that results in this value as optimal. In order to show the optimality of a certain setting of R_{12} and R_{13} only, we keep these two variables fixed and check for the highest social utility that can be reached by varying the two variables that are still free, k_2 and k_3. If this maximum value in the space restricted by the assignment of R_{12} and R_{13} is the same as the overall optimum, this assignment of R_{12} and R_{13} is seen as optimal. In the graph, the corresponding point is drawn in dark gray, suboptimal points in medium gray and unreachable points in light gray. Figure 6 shows a typical example of such a graph, produced on the basis of Scenario 1 in Appendix A. The white cross in this graph shows the setting that would have been used in [5], according to their fixed reward splitting strategy.

In other words, the dark gray areas in an optimality graph mark reward splittings, which allow for an optimal social utility. That is, when a setting of R_{11}, R_{12}, and R_{13} is chosen that lies within a dark gray area, it is possible

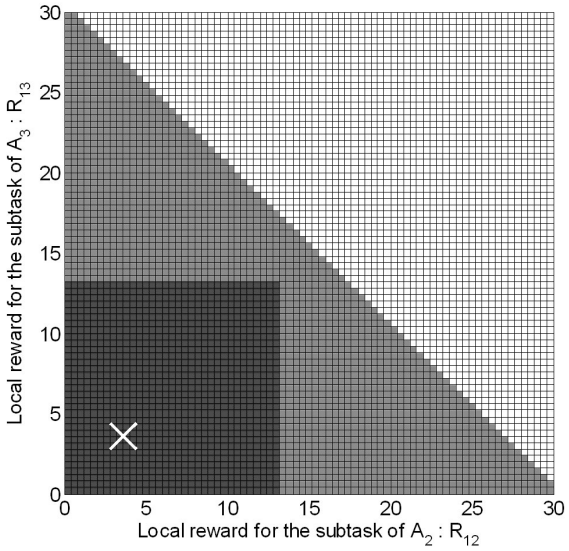

Figure 6: A typical optimality graph in the (R_{12}, R_{13})-space. The dark gray area in the lower left corner denotes reward splittings that allow for optimal social utility, the surrounding medium gray area yields only suboptimal results, and the light gray area above the diagonal is unreachable. The setting that would have been selected by the fixed strategy from [5] is marked with a white cross and lies within the dark gray area. Therefore, it would have been optimal here.

to set k_2 and k_3 in a way that the whole multi-agent system achieves the maximal social utility of 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} always 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 $R_1, R_{11}, R_{12}, R_{13} \geqslant 0$.

Of course, there are also optimality graphs for the other set of variables, the local attitude parameters. The dimensions spanning these graphs are k_2 and k_3, and the color of each point denotes whether for the respective setting of the variables there exists a reward splitting that leads to the maximal social utility possible in the current environment.

4.2 The Importance of Reward Splitting

4.2.1 Relevance of the Reward Splitting to Arbitrary Objective Functions

As pointed out in Subsection 3.2.2, for any goal function that differs from the social utility reward splitting is very important. The (R_{12}, R_{13})-space is divided into tiles which are tilted for the reward functions of different agents. Figure 4 illustrates this. Within a single tile, the difference of the lowest and the highest value can be up to 36% of the optimum and potentially more.

Thus, in the case of arbitrary goal functions in the agents, R_{12} and R_{13} should definitely be free variables in the optimization. The sampling should account for all the maximum points at least. Equations (18), (19), and (20) contain the necessary information to reconstruct their position.

4.2.2 Relevance of the Reward Splitting to the Social Utility

If all the agents are fully cooperative, the reward splitting still is important, albeit not as critical as if they were partially altruistic or self-interested. Figure 6 shows a typical, symmetric optimality graph.

For a lot of the environments we tested, the splitting from [5] turned out to be within the optimal region. However, in certain environments this is not the case and the fixed strategy forbids to achieve the optimal social utility. For one of these environments, Figure 7 shows the optimality graph. The underlying environment is Scenario 2 from Appendix A.

As we can see in the graphs, the density of optimal solutions is fairly high. This is the case, because for most values of $\lfloor Rn_i \rfloor$ there is an infinite number of ways to calculate it. As you may recall, $Rn_i = R_{1i} + k_i R_{11}$, and, from a global perspective, the optimization has control over all the variables on the right hand side. As stated in Subsection 3.2, $\lfloor Rn_i \rfloor$ is the only factor influencing the expected social utility. Therefore, in most environments a lot of reward splittings may lead to the optimal behavior - given the local attitude parameters are set accordingly. We could exploit this fact in one of the following ways:

- Introduction of 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.

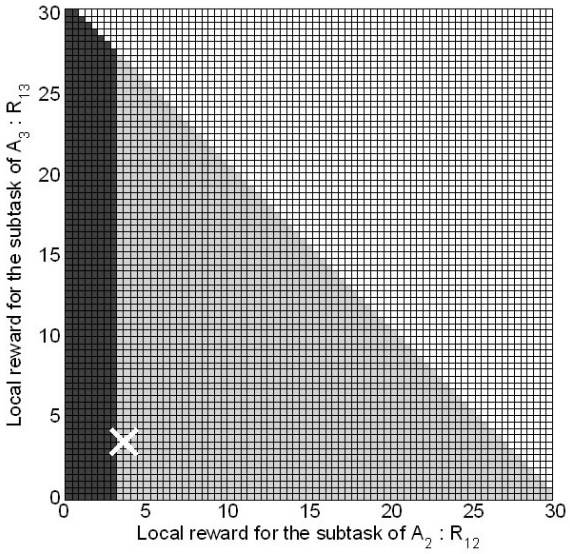

Figure 7: An optimality graph in the (R_{12}, R_{13})-space with a slim optimal area, again shown in dark gray. The white cross denoting the setting from [5] clearly lies in the suboptimal area, marked in medium gray. Thus, systematic reward splitting can lead to higher gains.

- Easiness and forgiveness are beneficial for approximate, distributed algorithms with limited communication bandwidth. Especially when the environment changes, not all agents can always gain complete knowledge.

From a global, system designers point of view, there exists some redundancy insofar that he has control over all of 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.

4.2.3 Canonical Optimal Solution for Cooperative Systems

Another observation from all optimality graphs in the (R_{12}, R_{13})-space 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 - although possibly not the most favorable one - in the case when all agents intend to maximize the social utility. This solution is optimal in terms of social utility, but very inflexible when it comes to secondary goals. In a lot of cases it will result in a highly skewed distribution of the reward among the agents, totally neglecting fairness.

To see why the social utility 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 social utility. Although it can be used, we would encourage other, more sophisticated approaches.

This section addresses the issue of how to divide up the reward for a shared task among the participating agents. Using the notion of optimality graphs, that allow to see at first glance which settings of variables are preferable, we showed that the fixed reward splitting strategy from [5] does not achieve

the maximal utility in certain environments. Furthermore we proved that keeping the complete reward for the shared task T_1 at agent A_1 is always optimal. It is a canonical solution, though not the most favorable one, to the reward splitting problem in fully cooperative systems that is guaranteed to enable optimal performance.

5 Different Ways to Calculate the Relational Reward

By taking into account the relational reward instead of the local reward, an agent that is requested to do a subtask can consider the benefits for other agents of it completing the subtask. That is, the agent bases its decision whether to do the task not solely on the local reward, but includes the other agents' local rewards to a certain degree. We came up with different ways to address the question of how this inclusion should take place. These can be separated into two categories: basic ways to calculate the relational reward and ideas of how to increase the flexibility of the basic formulae.

In the first category, three different ways have been proposed, all based on a real-world justification. In Subsection 5.1, we prove a certain order between them. That is, the expressiveness of the formulae differ and we state and prove a subset relation between them. Potentially there are results of one relational reward formula that lead to rewards which cannot be replicated by the other formulae, while it can never be the other way around. Due to the hypotheses we made, we can give clear and certain advice which notion to use in which situation.

For the second category of issues, we evaluate two notions that increase the flexibility. There is an example where one of the extensions yields gains which exceed the achievable maximum of the basic formulae.

We will show, that the traditional, static ways of choosing the relational reward are not optimal. This issue refers to setting the relational reward to either the local reward or the social utility.

5.1 Basic Relational Reward Formula

In order to account for the benefits towards the social utility, an agent A_i bases its decision on whether or not to do a subtask on the relational reward. The relational reward is the sum of the local reward and some non-local

reward times a factor, which we call the local attitude parameter or degree of local cooperation. Within this notion, we came up with the the three following basic ways to calculate the relational reward.

- $Rn_i^{(1)} = R_{1i} + \frac{1}{2}k_i R_{11}$
- $Rn_i^{(2)} = R_{1i} + k_i R_{11}$
- $Rn_i^{(3)} = R_{1i} + k_i(R_{11} + R_{1j})$

As before, there are certain constraints: $i, j \in \{2, 3\}, \quad i \neq j, \quad k_i \in [0, 1], \quad R_1 = R_{11} + R_{12} + R_{13}$. From a real-world viewpoint, $Rn_i^{(1)}$ is based on the idea that it might be wrong to account for one reward twice, that is agents A_2 and A_3 both take at most $\frac{1}{2}R_{11}$ into consideration. In the case of $Rn_i^{(2)}$, the contractee agent decides on a subtask request partially based on the reward for its contractor. The idea behind $Rn_i^{(3)}$ is to give each agent the chance to account for the reward of the whole task, R_1. Since this is the amount by which the social utility is increased in case of successful completion, this approach might be especially beneficial to agents whose goal is to maximize the social utility.

Our intuition is that, for a lot of cases, varying the local attitude parameter and the reward splitting highly influences the functions' outcomes, and that thus the situations (i.e. the settings of external parameters) in which one way of computing the relational reward permits for a higher gain than the others are rather rare. But there are statements which we can make and prove about the relationships of these formulae among each other.

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

$$\begin{aligned} &\forall R_1, R_{11}, R_{1i}, R_{1j} \geqslant 0 \; s.t. \; R_1 = R_{11} + R_{1i} + R_{1j} \quad with \quad k_i \in [0, 1] : \\ &\qquad I(Rn_i^{(1)}(k_i)) \subset I(Rn_i^{(2)}(k_i)) \subset I(Rn_i^{(3)}(k_i)) \end{aligned} \tag{21}$$

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:

$$\begin{aligned}
\forall R_1 \geqslant 0, \quad & with \quad k_2, k_3 \in [0,1] \quad and \quad R_{11}, R_{12}, R_{13} \geqslant 0 \; s.t. \; R_1 = R_{11} + R_{12} + R_{13} : \\
& I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13})) \\
\subset \quad & I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13})) \\
\equiv \quad & I(Rn_2^{(3)}(k_2, R_{11}, R_{12}, R_{13}) \times Rn_3^{(3)}(k_3, R_{11}, R_{12}, R_{13})) \qquad (22)
\end{aligned}$$

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

Proof

We will first prove hypothesis (21), 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 (22), 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 $k_i^{(2)}$.

$$Rn_i^{(1)}(k_i) \quad = \quad Rn_i^{(2)}(k_i)$$

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

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: $R_{11} > 0$

$$\frac{1}{2}k_i^{(1)} = k_i^{(2)} \tag{23}$$

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

$$k_i^{(3)} = k_i^{(2)}\frac{R_{11}}{R_{11} + R_{1j}} \tag{24}$$

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 $\frac{R_{11}}{R_{11}+R_{1j}} > 0$[1] and $\frac{R_{11}}{R_{11}+R_{1j}} < \frac{R_{11}}{R_{11}} = 1$.
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 $I(Rn_i^{(2)}(k_i))$.

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

Proof of $I(Rn_i^{(1)}(k_i)) \not\supseteq I(Rn_i^{(3)}(k_i))$:
Above, we have shown that $I(Rn_i^{(1)}(k_i)) \not\supseteq I(Rn_i^{(2)}(k_i))$ 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 (23) and 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 $I(Rn_2^{(1)}(k_2, R_{11}, R_{12}) \times Rn_3^{(1)}(k_3, R_{11}, R_{13}))$
$\not\supseteq I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$:

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 (24) 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 $I(Rn_2^{(2)}(k_2, R_{11}, R_{12}) \times Rn_3^{(2)}(k_3, R_{11}, R_{13}))$.

□

Note, that we do not make a statement about the direct effects of these hypothesis. In fact, in all of the scenarios we looked at, the basic way of calculating the relational reward did not have an impact on the expected social utility. However, the statement we do make, is that using $Rn_i^{(3)}$ when having control over the local attitude parameters, k_i, only, or using $Rn_i^{(2)}$

or $Rn_i^{(3)}$ when exhibiting control over the reward splitting as well, can take on the same and more values than the respective other ways. As stated in Subsection 3.2.2, the Rn_i are the only point of influence - given any fixed environment and the maximization as the goal of all agents. Therefore, the suggested formulae will lead to at least as high rewards as the others - and potentially more.

In the case where we have control over the reward splitting and the local attitude parameters, $Rn_i^{(3)}$ allows for more flexible solutions than $Rn_i^{(2)}$. Say, both A_2 and A_3 need to account for the whole social utility. The only solution to this constraint with $Rn_i^{(2)}$ is to set $R_{11} = R_1, k_2 = k_3 = 1$, and $R_{12} = R_{13} = 0$. With $Rn_i^{(3)}$ an arbitrary reward splitting can be realized, as long as $k_2 = k_3 = 1$. Also, choosing $R_{11} = R_1$ and $R_{12} = R_{13} = 0$ always allows for the maximum in $Rn_i^{(2)}$, since with this choice and k_2 and k_3 any element of $[0, R_1] \times [0, R_1]$ can be selected easily. With $Rn_i^{(3)}$, Rn_2 is not increasingly limited as R_{13} is raised - or vice versa. Although $Rn_i^{(2)}$ allows for as good contributions to the social utility as $Rn_i^{(3)}$, the latter formula can incorporate more sophisticated parameter settings.

5.2 The Need for more Flexibility in the Relational Reward calculation

After evaluating the basic ways of introducing the other agents' rewards into the local decision process from the preceding section, we now take a look at how to increase the flexibility in the relational reward calculation. The motivation behind more flexibility is to account for other metrics that cannot be dealt with by the basic ways to calculate relational reward. Such metrics may be high uncertainty on whether a subtask's reward will be paid, or the duration of a task, which does not influence the reward for the complete task but the reward per time step. The first issue could arise when the real reward for a subtask should be decent, but the probability of the payment taking place is low. One example for the latter case is Scenario 2 from Appendix A: The mean reward for agent A_2's local task is 3, A_3's local task pays 40 on average, and the shared task, T_1, only 30. Knowing only that, the best would be to always accept the two local tasks when possible. But the mean duration of T_2 is approximately three times as much as T_1's duration. Thus, A_3's relational reward, Rn_3, should be higher than R_3, which is in particular higher than R_1. If this is the case, A_3 accepts the subtask and sticks with it,

even when a request for the local task, which pays more, comes in.

If we summarize all three approaches for the basic relational reward calculation as $Rn_i = R_{1i} + k_i x$, the two ideas to deal with these cases are:

- $Rn_i = R_{1i} + k_i x$ with $k_i \in [a^{k_i}, b^{k_i}]$, $a^{k_i} \leqslant 0$, $b^{k_i} \geqslant 1$
- $Rn_i = u_i(R_{1i} + k_i x)$ where $u_i \in [0, 1]$ is the uncertainty associated with the shared task

5.2.1 More Flexibility Through Wider Bounds on the Degree of Local Cooperation

While the first two basic formulae, $Rn_i^{(2)}$ and $Rn_i^{(3)}$, restrict the relational reward to be in $[0, R_1]$, the first extension above, $Rn_i = R_{1i} + k_i x$ with $k_i \in [a^{k_i}, b^{k_i}]$, $a^{k_i} \leqslant 0$, $b^{k_i} \geqslant 1$, broadens the scope of Rn_i to values between $[-a^{k_i} R_1, b^{k_i} R_1]$. Obviously there is no gain if $a^{k_i} = 0$ and $b^{k_i} = 1$ at the same time, which is just the basic case.

But for less constraining boundaries, the additional opportunities lead to a higher social utility, as shown in Figure 8. This optimality graph in the (k_2, k_3)-space has been produced based on Scenario 2 from Appendix A with $k_i \in [-1, 2]$. For more explanation on optimality graphs, please refer to 4.1. There, the optimal social utility can be achieved only with a $k_i > 1.3$. Apparently, this optimum could not be realized with $k_i \in [0, 1]$.

Nevertheless, by loosening k_i's bounds we give up part of its real-world meaning: A $k_i = 0$ belongs to a self-directed agent, so is an agent with $k_i = -1$ spiteful? How does that interfere with its goal of maximizing the social utility?

5.2.2 Explicitly Accounting for Uncertainty

The second approach, $Rn_i = u_i(R_{1i} + k_i x)$ with u_i as explicit uncertainty, does not broaden the scope of Rn_i, but introduces the flexibility to set it to any value in $[0, Rn_i^{(l)}]$. In other words, we can choose the reward splitting and local attitude parameters following other criteria, like a minimal real reward for a subtask. Meanwhile, it is still possible to account for an arbitrary uncertainty in the payment of the subtask - at the cost of some redundancy: this notion introduces two more variables, u_2 and u_3, thus adding two dimensions to the optimization problem and leading to a minor search space explosion. For the latter reasons, this idea might have little applicability.

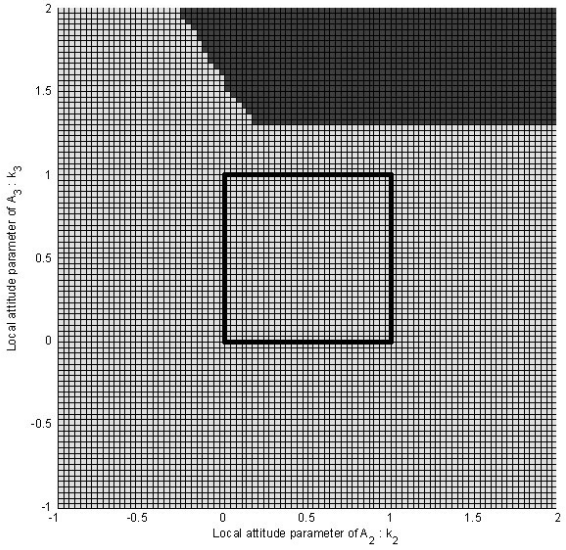

Figure 8: An optimality graph in the (k_2, k_3)-space showing the need for dealing with greater uncertainty: The dark gray area refers to settings with which an optimal reward can be achieved, the medium gray area denotes suboptimal settings, the black square shows the bounds within which original settings may take place.

5.3 Sub-Optimality of Traditional Solutions

From a high-level point of view, the relational reward determines the value of doing a subtask to an agent and its goal. While noting it as relational reward is a very recent idea, the issue of quantifying the value of a task has, of course, been addressed before. Former solutions were usually static and simple, in contrast to our adaptive optimization approach, that bases its decisions on the current environment. By introducing relational reward, we now can distinguish between the goals and the means we use to meet them in the best possible way.

In this manner, there are two traditional ways to set the relational reward, which are related to the goal of the respective agents. Self-interested agents, whose goal is to maximize the local reward, typically only cared about the real reward in their decisions. This is equal to self-directed agents in our model, i.e. setting their local attitude parameters to $k_i = 0$, which results in $Rn_i = R_{1i}$. Completely cooperative agents, on the other hand, want to optimize for the social utility. They based their decisions usually on the joint reward without caring about the distribution among the agents. This behavior corresponds to externally directed agents with $Rn_i = R_1$. It can be achieved via computing Rn_i as $Rn_i^{(3)} = R_{1i} + k_i(R_{11} + R_{1j})$ (as introduced in Subsection 5.1), with $k_i = 1$.

When the goal of all agents is to maximize the social utility, both of these simple, static ways are suboptimal. Figure 9 shows the optimality graph in the (k_2, k_3)-space for Scenario 3 in Appendix A. The lower left corner, marked with a black cross, corresponds to $Rn_i = R_{1i}$ and is far from the optimal, dark gray area.

This is the case, because in the corresponding scenario the durations of the local task T_2 and T_3 are far higher than the durations of the subtasks T_{12} and T_{13}. At the same time, the rewards of the local tasks are a little higher than half of the reward for the shared task, T_1. In order to achieve the maximal social utility, both A_2 and A_3 must therefore have a relational reward of more than half of R_1, which cannot be the case with $Rn_i = R_{1i}$.

Figure 10 also visualizes the (k_2, k_3)-space, but for Scenario 1 from Appendix A. The relational reward here has been calculated as $Rn_i^{(3)} = R_{1i} + k_i(R_{11} + R_{1j})$. The black cross in the upper right corner, where $(k_2, k_3) = (1, 1)$, denotes the position of $Rn_2 = Rn_3 = R_1$, which corresponds to the second traditional way of setting the relational reward. In this environment, it lies in the suboptimal area.

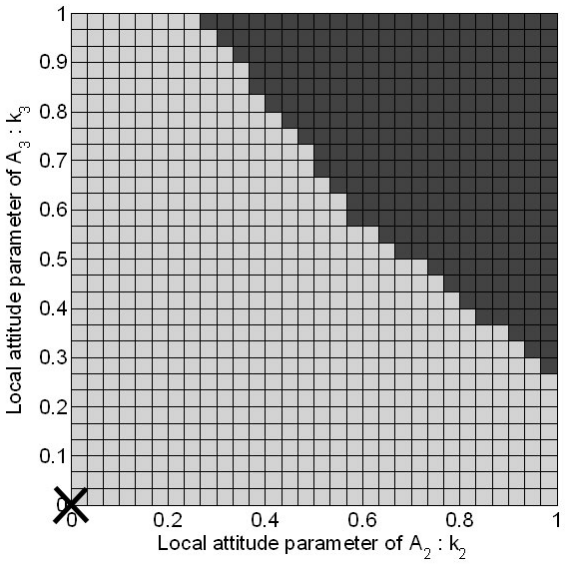

Figure 9: An optimality graph in the (k_2, k_3)-space, gathered from Scenario 3 in Appendix A and using $Rn_i^{(2)}$. Dark gray areas allow for optimal solutions, light gray areas denote sub-optimality. The black cross marks where $(k_2, k_3) = (0, 0)$, thus $Rn_i = R_{1i}$, which is not an optimal solution.

This result is intuitively explainable by the duration/reward-relation again: The subtasks of the shared task take twice the time of the local tasks, T_2 and T_3, whereas the mean summed reward of them is only a little less. When the agents take the total reward of the shared task into account in their decisions, that is $Rn_i = R_1$, they will always choose to commit to it. However, the number of local tasks they can fit into their schedules is higher, leading to more total reward.

In both cases, the difference between the social utilities in the optimal area and at the crosses' locations is around 10%. Other scenarios presumably yield even higher differences.

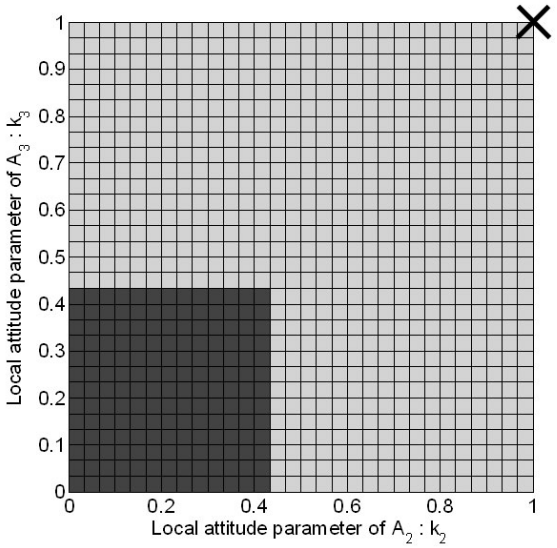

Figure 10: An optimality graph in the (k_2, k_3)-space, gathered from Scenario 1 in Appendix A and using $Rn_i^{(3)}$. Dark gray areas allow for optimal solutions, light gray areas denote sub-optimality. The black cross marks where $(k_2, k_3) = (1, 1)$, thus $Rn_i = R_1$, which, again, is a suboptimal solution.

Clearly, the maximization of the social utility differs from the original goal of self-interested agents. The traditional and most intuitive way to calculate the relational reward, i.e. setting $Rn_i = R_1$, in cooperative systems in general is suboptimal. The other case shows that borrowing the traditional way to set relational rewards in fully self-interested systems, $Rn_i = R_{1i}$,

for a cooperative system cannot overcome the shortfall either. Thus, our statements here are: It is beneficial to choose the relational reward carefully and with respect to each environment, when having a cooperative goal. More generally speaking, distinguishing between the goal of a group of agents and the obvious means to achieve it can be advantageous.

In this section, we examined different basic ways to set the relational reward and discussed how to extend them towards more flexibility. We stated a hypothesis describing the relationship between the three basic ways and proved it. Therefore, we can be certain that computing Rn_i as $Rn_i^{(3)} = R_{1i} + k_i(R_{11} + R_{1j})$ is suited best among the introduced options. Still, if k_i is required to be in $[0, 1]$, the relational reward cannot be more than R_1. However, for some environments, that is required in order to achieve the maximal social utility. We showed that Scenario 2 from Appendix A is among them. The needed flexibility can be established by loosening the bounds on k_i. In the end, we examined the two traditional and most intuitive ways to set the relational reward: to account for the local or the social utility only. It was shown, that there are environments for both where they keep the system from reaching the maximal social utility.

6 The Effects of Explicit Decommitment

When agent A_2 or A_3 receive a task that causes a scheduling conflict with a subtask it is committed to, the agent has to decide among the two. If the newly arrived task seems more beneficial than the scheduled subtask, the agent decides on doing the other task and not completing the subtask. The agent *decommits* from the subtask. Since the subtask will not be completed, the other agents who work on the shared task will not receive any reward for it. If the agent who decommitted from the subtask notifies the other agents of its decision, it *decommits explicitly*. When no message is sent, *implicit decommitment* takes place. That is, since the other agents do not receive any reward, they learn about the decommitment - albeit with hindsight. Thus far, we assumed implicit decommitment, as it was the case in [5].

Note, that introducing explicit decommitment is an alteration of the communication protocol. This section contains possible extensions to the statistical model, enabling it to capture the effects of explicit decommitment. The

results shown do not necessarily reflect the real world in that respect.

In general, decommitment could be associated with a cost (or fine), which has to be accounted for in the respective decisions. In this work, we will only consider cost-free decommitment.

Explicit decommitment, however, can be beneficial. An agent that learns about the decommitment knows that its work on a shared task is useless, and could take it off the schedule under certain circumstances. Therefore, some other task might potentially be accepted, leading to higher rewards. We will see, that using our intermediate extensions to introduce explicit decommitment suggests advantages in certain environments.

6.1 Necessary Assumptions

In order to keep a feasible model, we make the following assumptions:

- Tasks can be removed from the schedule only before the work on them began.
- Once started, a task can no longer be stopped or paused.
- There is a delay of some time $\delta > 0$ between the arrival of a task at agent A_i that causes it to decommit, and the arrival of the decommitment message at A_j.

These assumptions can be true in real-world applications, because before the execution was started, the task did not change anything. But once the work on it started, the cost of reaching a consistent state after an interruption can be equal to simply completing the task - or higher. Non-zero delays are the common case, since reasoning about a decision and communicating it typically take some time, even if it is not much.

When talking about message delay, δ, we mean the delay between the arrival of a task that causes agent A_i to decommit from subtask T_{1i}, and the time when A_j makes note of the decommitment (with $i, j \in \{2, 3\}$ and $i \neq j$). For our notion of delay it is irrelevant whether the decommitment message is sent to A_1 only and then forwarded to agent A_j, or sent directly from A_i to both A_1 and A_j.

The case of instantaneous communication and computation, i.e. the delay is 0, can be approximated by our model in the following way: setting $\delta = 1$ while scaling up the other time-related parameters, such as e_i, dur_i, and sl_i.

This way, the error introduced by the delay can be made arbitrarily small, or, in practice, as small as machine precision allows. As a result, this technique can make δ negligibly small.

6.2 Extension of the Statistical Model

Due to the message delay, we have to introduce the notion of time into the statistical model. We only describe, where the model from Section 2 has to be altered in order to cope with explicit decommitment.

If agent A_2 decommits from subtask T_{12} and the decommitment message reaches A_3 before it starts to work on T_{13}, it can take its subtask off the schedule and free the assigned time. Thus, the probability of conflict with this subtask decreases. It is the probability of this event, that we want to model here. Affected are only $Pc_{12,2}, Pc_{13,3}, Pc_{12,12}$ and $Pc_{13,13}$. Since time has to be taken into account, we rename them to $Pc^t_{1i,i}$ and $Pc^t_{1i,1i}$, respectively, where t denotes the according time step.

Let δ be the message delay time. Then, the probability of a decommitment message arriving at time t is the probability of the other agent sending such a message at time $t - \delta$. The latter is referred to as $Pdec^{t-\delta}_{1i}$, being the probability that agent A_i sends a decommitment message about subtask T_{1i} at time $t - \delta$. Thus, the probability of conflict changes as follows:

$$Pc^t_{1i,i} = Pc^0_{1i,i} - Pdec^{t-\delta}_{1j} \quad (25)$$

$$Pc^t_{1i,1i} = Pc^0_{1i,1i} - Pdec^{t-\delta}_{1j} \quad (26)$$

where $Pc^0_{1i,i}$ and $Pc^0_{1i,1i}$ are the original probabilities of conflict, $Pc_{1i,i}$ and $Pc_{1i,1i}$ respectively.

But let us first construct the model for explicit decommitment in a bottom up fashion. A_i decommits from a subtask T_{1i} when a new local task arrives and one of the following two scenarios is true. The new local task causes a conflict with T_{1i} only and its reward is higher. Or the new local task is in conflict with the subtask and another local task and the reward for both of the local tasks are higher. A conflict with another shared task does not cause decommitment, because their reward is equal, leaving no benefit from replacing one shared task with another. Thus, A_i's decommitment probability can be described as follows.

$$Pdec_{1i}^{t} = \frac{1}{r_i}[Pc_{1i,i}^{t}(1 - Pc_{i,i}^{t})P(R_i > Rn_i) + Pc_{1i,i}^{t} Pc_{i,i}^{t} P(R_i > Rn_i)^2] \quad (27)$$

with

$$P(R_i > Rn_i) = \begin{cases} 1, & \lfloor Rn_i \rfloor < a_i^r; \\ \frac{b_i^r - \lfloor Rn_i \rfloor}{b_i^r - a_i^r}, & a_i^r \leqslant \lfloor Rn_i \rfloor < b_i^r; \\ 0, & \lfloor Rn_i \rfloor \geqslant b_i^r. \end{cases}$$

Note, however, that the above extensions of the statistical model reflect our current, preliminary status of work on the issue. They are no final or proven results and may only capture the real world procedures partially.

The way the probability of conflict, Pc, is set up does not allow for conflicts with tasks that have already been started, since the arrival time of the new task is assumed to be before or equal to the earliest start time of the task it is in conflict with. Thus, we need not bother with whether an agent has already started to work on a subtask which it would be beneficial to decommit from now.

Omitting the subscript indices, Pc^t depends on $Pdec^{t-\delta}$. In turn, $Pdec^{t-\delta}$ depends on $Pc^{t-\delta}$, which then depends on $Pdec^{t-2\delta}$. This dependency chain is shown in Figure 11, and ends at the start time of the whole run, $t = 0$, since $\forall t < 0 : Pdec^t = 0$. That is, there cannot be a task to decommit from at any time $t < 0$.

In the scenarios we tested, the probabilities of decommitment and conflict converged very quickly to a fixed value. Usually that was the case after $10 \cdot \delta$ time steps. Once, the values do not change within $\delta + 1$ time steps, they will not change anymore. That is, when $Pc_{1i,i}^{t} = Pc_{1i,i}^{t+\delta}$, it will be the same for any later time t_l, with $t_l \geqslant t$. We denote the first such time step t that satisfies this criterium as t_{conv}. For our comparisons, we only used the converged probabilities, $Pdec_{1i}^{t_{conv}}$, $Pdec_{1j}^{t_{conv}}$, $Pc_{1i,i}^{t_{conv}}$, $Pc_{1i,1i}^{t_{conv}}$, $Pc_{1j,j}^{t_{conv}}$, and $Pc_{1j,1j}^{t_{conv}}$. When we replace Pc with $Pc^{t_{conv}}$ in the statistical model from Section 2, it captures explicit decommitment.

6.3 Impacts of Explicit Decommitment

Let us compare the following three approaches:

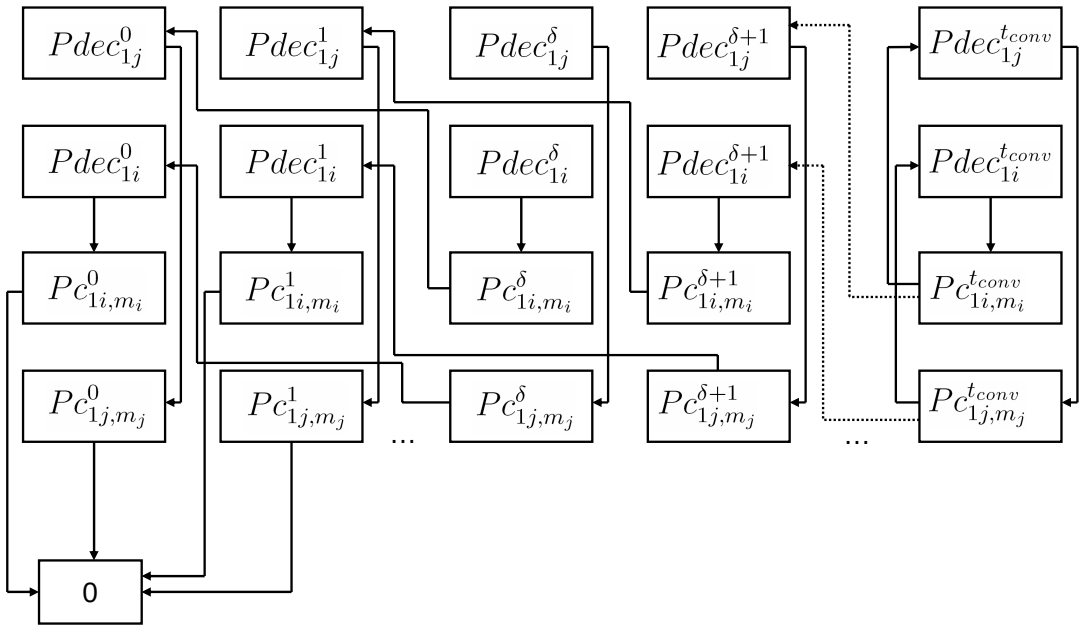

Figure 11: The dependencies between the probabilities. The decommitment probability of an agent depends on the probability of conflict with the subtask in the same time step. The probability of conflict at a certain time step in turn depends on the decommitment probability δ time steps earlier. The latter is zero for any time $t < 0$. Pc^t_{1i,m_i} stands for both $Pc^t_{1i,i}$ and $Pc^t_{1i,1i}$, the probabilities of conflict, and $Pdec^t_{1i,i}$ denotes the decommitment probability. t_{conv} is a time step, where the probabilities converged.

- Varying the local attitude parameters, but without explicit decommitment
- Keeping the local attitude parameters fixed, but using explicit decommitment
- Varying the local attitude parameters and employing explicit decommitment

The second approach's performance is similar to the traditional ways to calculate the relational reward from Subsection 5.3: any fixed strategy is suboptimal in certain cases. The optimality graphs look like Figure 9 and 10

with slightly smaller optimality areas. Therefore, we did not provide them here.

With the preliminary extensions of the statistical model from Subsection 6.2, which introduce explicit decommitment, there can be an increased maximal social utility. When messages get delayed by 3 time steps, the summed rewards per time in Scenario 1 from Appendix A increase from 0.7210 with implicit decommitment to 0.7270 with the explicit alternative, in Scenario 2 from 0.5976 to 0.6030, while in Scenario 3 there is no difference.

The differences in the social utility are not large (approximately 1%) in the environments we provide. There are two possible ways to interpret them: The difference can be seen as statistically irrelevant and will never be significant in any other scenario; Or it can be interpreted as a present difference and assumed that other environments exist, where the differences are more significant.

Since we do not know any better, we can only state, that the first and the third approach performed approximately equally well in the scenarios we provided. Both were clearly better than the second approach.

The uncertainty about the payment for a subtask of the shared task gets reduced when explicit decommitment is available. Our preliminary results suggest that, in this case, there are still significant benefits from varying the local attitude parameters and reward splitting. Therefore, there is more to the relational reward then anticipating the payment uncertainty of shared tasks when there is no explicit decommitment.

This section introduces decommitment, which can be conducted in one of two ways: implicitly or explicitly. While decommitment in [5] was strictly implicit, we here offer a possible way to extended the statistical model to enable the simulation explicit decommitment. The three scenarios we provide show only small differences of about 1% in the maximal social utility, but there might be other environments, where the difference between explicit and implicit decommitment is more significant. Due to the fact that the extensions of the model have not been validated as of now, we did not try to find such environments yet.

7 Conclusions and Future Work

7.1 Summary

In this work, we introduced an example multi-agent system with three agents as well as the corresponding statistical model. The model captures the system's behavior in terms of the individual gains of the agents as a consequence of the values of environmental and local parameters. It is based on the notions of integrative negotiation and relational rewards. Both the system and the model were borrowed from [5].

We examined the structure of the function that maps local parameters to expected utilities for a given environment. For the case of fully cooperative agents, this function is the a step function, more precisely the four-dimensional equivalent of a step function. Therefore sampling the values on all steps and choosing a setting that leads to the highest possible reward is sufficient to maximize the social utility. We developed the math that is necessary to reconstruct the shape of the multi-dimensional steps, allowing for a seamless sampling strategy.

In the preliminary work, the optimization was only conducted over the space of the two local attitude parameters, k_2 and k_3. But since the rewards for the subtasks, R_{12} and R_{13}, can be chosen by agent A_1, they are also local control parameters. The issue of how to choose them is referred to as reward splitting, because A_1 divides the reward for the shared task among the agents working on it. We showed, that the static way to split from the preceding work in general is suboptimal. However, there is a canonical, optimal solution to this issue, given the agents are fully cooperative. This canonical solution is to keep all the reward at agent A_1. Although it is guaranteed to enable maximal social reward, we encourage more sophisticated and potentially fairer approaches.

The notion of relational reward allows agents to account for the reward of other agents in their decisions. Therefore, they can reflect the importance of a shared task with respect to their goal. While working with the system, we thought about the ways, relational reward can be calculated, and extracted three basic ways as well as two approaches to increase the flexibility. We were able to proof a relationship between the three basic ideas and concluded that calculating the relational reward, Rn_i, as $Rn_i^{(3)} = R_{1i} + k_i(R_{11} + R_{1j})$ is most promising among them. Increasing the flexibility by allowing k_i to be in a wider range than $[0, 1]$ can further increase the gains.

Relational reward is a rather recent idea. Traditionally, cooperative agents based their decision on doing one task or another statically on either the real reward for the agent or the combined reward for the group of agents. This work showed that neither way to address the issue can always be as good as dynamically adapting the relational reward to the respective environment.

When a new task arrives at an agent, it may decide to drop a task it is committed to. If it drops a subtask from a shared task and notifies the other agents about that, the others can potentially take their parts of the shared task off their schedules, since those will not be rewarded anymore. This notification, which implements explicit decommitment, was so far not part of the model. Here, we introduced possible extensions that enable the statistical model to simulate explicit decommitment. Since these extensions are not validated empirically yet, they form a preliminary piece of work and we are not certain, how the results are to be interpreted.

Although the example multi-agent system seems small and simple at first glance, it turned out to be a quite rich model. It is the smallest instance of a system with multi-linked negotiation and the need for optimization. Therefore, we expect that results, which were already important in this lowest common denominator of a class of multi-agent systems, are present in larger systems as well. Still, there are many ideas how this work can be extended. In the following section, we provide a subset of them in greater and lesser details.

7.2 Future Work

7.2.1 Finalizing the Results on Explicit Decommitment

As it has been done with the statistical system with implicit decommitment in [5], the extensions of the model for dealing with explicit decommitment should be validated empirically.

Also, one of the following two options should be pursued:

- Establishing a proof, that the differences between explicit and implicit decommitment are always statistically insignificant, or
- Finding scenarios where there is a significant difference.

7.2.2 Exploiting the Density of Maximal Solutions With Local Mechanisms

In this work, we assumed the point of view of a system designer: perfect knowledge and control over all parameters at any given time. In realistic systems the agents are most likely to lack a global view, especially when the environment evolves over time. [5] introduces a way to deal with that, by learning the probability of payment for the shared task and optimizing for the simplified equations. (The probability of payment replaces $Pcommit_i$ in the statistical model, which is the probability that the other agent commits to the task, but it also captures the probability of decommitment.) Of course, in this process the issue of reward splitting was not addressed.

Let us assume that A_1 controls the reward splitting and A_2 and A_3 control their respective local attitude parameter. If the reward splitting is kept fixed while both A_2 and A_3 set their k in a way that maximizes the social utility function independently, the system converges to the maximum that can be reached with the given reward splitting. That is a proven result from [5].

Our analysis from Sections 4 and 5 includes the statement that the density of optimal solutions in the (R_{12}, R_{13})-space of cooperative systems is fairly high. As long as we can guarantee to have a reward splitting, that allows for an optimal solution, we can keep it fixed. In any fairly stable environment, it should be sufficient to establish a more or less global view at agent A_1 very infrequently. Agent A_1 could optimize the reward splitting in these moments, preferably choosing a robust splitting, and notify the other agents of its decisions. They, in turn, could constantly adapt to little changes in the environment.

7.2.3 Game-Theoretic Approach to Optimize for Multiple Objective Functions

In this work, we mostly concentrated on systems of fully cooperative agents. As soon as the objective function of one agent differs from the social utility, we have to use another optimization approach. As mentioned in Subsection 3.1, the objective function of an agent that is partially self-interested differs from the social utility function. That is, when one or more of the agents are not fully cooperative, we have to change to a completely different way of optimization. A game-theoretic approach that finds Nash equilibria lends itself for that purpose.

We suspect that such a change of gears, reduces some of the redundancy in the current optimization, thus decreasing the density of optimal solutions.

Another impact is that the diamond-shaped tiles that form the (R_{12}, R_{13})-space, are tilted for any objective function but the social utility. Therefore, the optimization cannot rely on coarse sampling anymore. While it is not hard to see where the maximal point within a tile lies, the Nash equilibrium could be at another point.

7.3 Miscellaneous

We also want to mention the following points briefly.

- Other random distributions could be used instead of the uniform distributions, e.g. multi-variant Gaussians.
- More agents and more complicated task structures can be modeled using similar methods and patterns.
- A cost for decommitment could be introduced. On the one hand, that would make agents less likely to decommit, on the other hand, they have to reason about the cost when committing to a task.
- The optimization process could be revised, when we exhibit control over the reward splitting and the local attitude parameters: First, one could search for the optimal relational rewards as they appears in the statistical model, $\lfloor Rn_i \rfloor$. This is a two-dimensional, discrete search. Then, an arbitrary way of calculating them from the local control parameters (k_2, k_3, R_{11}, R_{12} and R_{13}) can be selected, e.g. following other guidelines.

A Appendix A: The Scenarios

The following tables list the environmental parameters of the three scenarios. The notation $[a, b]$ means that the according parameter p is uniformly distributed between a and b: $a < p \leqslant b$. For a detailed description of the semantics of the parameters, please refer to Subsection 2.1. Below, we repeat the short parameter description from there.

A task T_i is determined by a set of parameters:

- r_i: task T_i arrives at an agent at time t with a probability of $1/r_i$.
- e_i: the difference between the arrival time of a task T_i and its earliest start time est_i.
- dur_i: the duration of the task T_i.
- sl_i: the difference between the earliest possible finish time of a task T_i and the deadline dl_i.
- R_i: the reward of a task T_i if it's finished.

The relationship of e_i, est_i, dur_i, sl_i and dl_i is illustrated in Figure 2.

A.1 Parameters of Scenario 1

	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]

A.2 Parameters of Scenario 2

	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]

A.3 Parameters of Scenario 3

	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]

References

[1] Boyd, S., Xiao, L., and Mutapcic, A. Subgradient Methods. Stanford University, October 1, 2003.

[2] Kim, K., Paulson, B., Petrie, C., and Lesser, V. Compensatory Negotiation for Agent-Based Project Schedule Coordination, 1999.

[3] Lesser, Victor, and Zhang, XiaoQin. Multi-Linked Negotiation in Multi-Agent System. *Proceedings of the First International Joint Conference on Autonomous Agents And MultiAgent Systems (AAMAS 2002)* (2002), 1207–1214.

[4] Nedic, A., and Bertsekas, D.P. Incremental Subgradient Methods for Nondifferentiable Optimization. *Report LIDS-P-2460, Dec. 2000, SIAM J. on Optimization, Vol. 12* (2001), 109–138.

[5] Shen, Jiaying, Zhang, XiaoQin, and Lesser, Victor. Degree of Local Cooperation and its Implication on Global Utility. In *Proceedings of Third International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2004)* (New York, New York, July 2004), vol. 2, IEEE Computer Society, pp. 546–553.

[6] Zhang, XiaoQin, Lesser, Victor, and Abdallah, Sherief. Efficient Ordering and Parameterization of Multi-Linked Negotiation. *Proceedings 2nd International Joint Conference on Autonomous Agents and Multiagent Systems. (Extended abstract) AAMAS03* (July 2003), 1170–1171.

[7] Zhang, XiaoQin, Lesser, Victor, and Wagner, Thomas. Integrative Negotiation in Complex Organizational Agent Systems. In *Proceedings of the 2003 IEEE/WIC International Conference on Intelligent Agent Technology (IAT 2003)* (Halifax, Canada, 2003), IEEE Computer Society, pp. 140–146.