Trusted Kernel-Based Coalition Formation

Bastian Blankenburg[1], Rajdeep K. Dash[2], Sarvapali D. Ramchurn[2],
Matthias Klusch[1], Nicholas R. Jennings[2]
[1]German Research Centre for Artificial Intelligence, Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
[2]School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, UK
[1]{blankenb,klusch}@dfki.de, [2]{rkd02r,sdr,nrj}@ecs.soton.ac.uk

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems, Coherence and coordination.*

General Terms

Algorithms, Design, Economics.

Keywords

Coalition formation, rational agents, task allocation, trust, payment protocol.

ABSTRACT

We define **Trusted Kernel-based Coalition Formation** as a novel extension to the traditional kernel-based coalition formation process which ensures agents choose the most reliable coalition partners and are guaranteed to obtain the payment they deserve. To this end, we develop an encryption-based communication protocol and a payment scheme which ensure that agents cannot manipulate the mechanism to their own benefit. Moreover, we integrate a generic trust model in the coalition formation process that permits the selection of the most reliable agents over repeated coalition games. We empirically evaluate our mechanism when iterated and show that, in the long run, it *always* chooses the coalition structure that has the maximum expected value and determines the payoffs that match their level of reliability.

1. INTRODUCTION

Coalition formation (CF) is the coming together of a number of distinct, autonomous agents in order to act as a coherent grouping in which they increase their individual gains by collaborating. As such, it is an important form of interaction in multi-agent systems (MAS) in general and in particular it has recently been advocated for task allocation scenarios where agents derive a certain value (and cost) from tasks being performed in the coalition in return for payments from other agents in the coalition [11]. Examples of such scenarios include the Grid (where virtual organisations are formed to perform services not achievable by a single agent [4]) and sensor networks (where groups of sensors collaborate to track a target [6]). Now, in this context, cooperative game theory (CGT) provides a well developed and mathematically founded framework to determine which coalitions should be formed and how the respective coalition values should be distributed in an individually rational and stable manner [7] (i.e. no agents finds an incentive to break away from the coalition structure formed).

CF approaches that rely on CGT to determine stable coalition structures and payoffs must ensure that all relevant information is known by all agents. This, however, is an unrealistic assumption in real world MAS scenarios. In previous work on CF, this problem has been approached in several ways, incorporating learning, heuristics, or a communication phase where agents inform about each other's private information (see e.g. [2, 5, 11], respectively). While the latter one enables the agents to find an exact CGT solution, in [1] it was shown for at least one specific Kernel based CF protocol that agents are generally able to deceive each other about their valuations and costs to unjustifiably obtain a higher payoff. In fact, to date, there is no existing CF mechanism that allows the sharing of private formation while preventing agents from deceiving each other. More specifically, existing work does not provide any protocols which allow agents to safely share private information about costs and valuations in a non-manipulable way that permits the calculation of the agents' exact and justified payoffs in the coalition. Moreover, extant work in CGT based CF neglects the fact that, in most complex environments, there exists some uncertainty about the level of reliability of the participating agents. Thus, agents may not perform tasks perfectly or even at all and this may lead to agents overestimating coalition values and consequently choosing inefficient coalitions and selecting the wrong payoffs. Furthermore, existing work does not provide a protocol that ensures that agents actually obtain (or effect) the payments that they should. The problem here is that while a CGT solution is stable once side-payments have been executed, agents might still have an incentive to break away from their coalition at some time *during* this execution.

Against this background, we develop the area of trusted coalition formation through our *Trusted Kernel-based Coalition Formation* (TKCF) mechanism which caters for the above shortcomings (we discuss our choice of the Kernel as a solution concept in section 3). Specifically, this work advances the state of the art in the following ways. First, we develop a novel communication protocol that relies on cryptographic techniques to ensure agents can safely exchange their private information in order to calculate expected coalition values and the agents' payoffs. The protocol significantly reduces the possibility that an agent can exploit any information asymmetry (i.e. it is aware of another agents' private information without hav-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.
Copyright 2005 ACM 1-59593-094-9/05/0007 ...$5.00.

ing itself provided its private information) to gain higher payoffs. Second, we capture the uncertainty about the reliability, or Probability Of Success (POS), of agents through the concept of trust and integrate it in a kernel-based payoff calculation mechanism. Here, trust is defined as the expectation that agents will perform reliably when defecting would prove more profitable. Specifically, trust incorporates an agent's private observation of its counterpart's POS, as well as other agents' reports about it (i.e. the agent's reputation). Thus, through the use of trust we ensure that the most reliable agents are selected in the mechanism as the trust values of agents are refined over repeated coalition games. Third, we develop the first payment protocol of its kind to ensure agents in the coalition actually obtain their due payoff. This is achieved by specifying both the sequence and amount of payments that should occur. Moreover, our payment protocol occurs before tasks are actually performed as doing otherwise may entice the agent not to implement the payment to maximise its gains (conversely the trust model incentivises agents not to be unreliable as the agents are considered to be non-myopic).

The remainder of the paper is organised as follows. Section 2 discusses related work in the area of CF. In section 3 we provide the definitions from CGT that we use in rest of the paper. In section 4, we then discuss how trust is incorporated into the standard CGT framework and we describe TKCF. Given this, in section 5 we prove the properties of the mechanism. Section 6 empirically evaluates TKCF, while section 7 concludes the paper and gives a brief overview of future work in this area.

2. RELATED WORK

Of the three main components we design in TKCF, namely the communication protocol, the computation of payoffs given uncertainty about the agents' reliability, and the payment protocol, only the last two have been considered in the CF literature in MAS (albeit in a very limited fashion).

Thus, with regards to uncertainty in CF, we note the work of [2], where agents learn about each others types (that are uncertain). This model implies that agents can have different expectations of coalition values. To account for this, a *Bayesian Core* was introduced. While covering a broad range of uncertainties, the Bayesian Core is shown to be *not always non-empty*, and the already exponential complexity for computing core stable solutions is further increased. In contrast, our incorporation of a trust model in a kernel-based payoff computation mechanism does not require a specific extended CGT solution concept. Thus, it allows for applying low-complexity variants like the polynomial kernel (see e.g. [11]). We also differentiate our work from that of Vassileva et al. [12] which uses trust to compute coalitions since they do not consider stable payoffs which ensure agents will not disengage from the chosen coalition. We also note the heuristic approach to general CF under uncertainty of Kraus et al. which, contrary to ours, avoids computing stable payoffs in the sense of CF (see e.g. [5]).

Regarding the payment protocol, the closest work to ours is that by Sandholm and Lesser in their model of levelled-commitment contracts [9]. Though their model is not specifically developed for CF, it is the only existing protocol that specifies how payments should be made for task allocation schemes. In particular, it allows agents to decommit from contracts by paying some penalty. It is however not clear how this penalty payment could be enforced without the existence of enforcing third parties, which our mechanism does not require.

3. COALITION GAMES

In this section we briefly recall some basic definitions of CGT using the task allocation scenario we will consider in this paper. Thus, we define agents as elements of the set $\mathcal{I}$ noted as $1...I$ where $I = \mid \mathcal{I} \mid$. We will use identifiers i, j, k to refer to an agent in the set. Any subset $C \subseteq \mathcal{I}$ represents a coalition of agents. A *coalition game* in characteristic function form is a pair $(\mathcal{I}, \boldsymbol{v})$ with the set of agents $\mathcal{I}$ and the *characteristic function* $\boldsymbol{v} : 2^{\mathcal{I}} \mapsto \mathbb{R}$. $\boldsymbol{v}(C)$ is called the value of the coalition C and intuitively it can be viewed as a measure of the total payoff achievable by C if all its members cooperate effectively ($\boldsymbol{v}(\emptyset) = 0$). In the task allocation context, agents can perform and/or request tasks $\tau \in \mathcal{T}$ to be performed. An agent i requesting a task has a valuation $v_i : \mathcal{T} \mapsto \mathbb{R}$ a cost $c_i : \mathcal{T} \mapsto \mathbb{R}$ for tasks it executes. For a given coalition C, we denote the set of all possible mappings from tasks to agents as A_C. For any $\alpha_C \in A_C$, $\tau_i^j \in \alpha$ denotes that task τ requested by i is to be executed by j. Note here that the same task might be requested by many agents which will all derive a positive value when the task is performed even if they do not pay for it specifically. Let $v_{\alpha_C} := \sum_{\tau_i^j \in \alpha_C} v_i(\tau) - c_j(\tau)$ be the total payoff for C given α_C. Then α_C^* with $\forall \alpha_C \in A_C : v_{\alpha^*} \geq v_\alpha$ denotes a task allocation which maximizes the achievable total payoff for C. We call the value created for agent i if α_C^* is executed, disregarding any side-payments, the *local worth* of an agent i:

$$w_i(C) := \sum_{\tau_i^j \in \alpha_C^*} v_i(\tau) - \sum_{\tau_j^i \in \alpha_C^*} c_i(\tau) \tag{1}$$

Hence the overall coalition value is the sum of all local worths of agents in the coalition, i.e.:

$$\boldsymbol{v}(C) := \sum_{i \in C} w_i(C) \tag{2}$$

A *configuration* $(\mathcal{S}, \boldsymbol{u})$ for a game $(\mathcal{I}, \boldsymbol{v})$ specifies a *payoff distribution* $\boldsymbol{u} = \langle u_i, ..., u_{\mathcal{I}} \rangle$ for a *coalition structure*, which is a partition of $\mathcal{I}$. Formally, $\mathcal{S} \subset 2^{\mathcal{I}}$ with $\forall C, C' \in \mathcal{S} : C \bigcap C' = \emptyset$ and $\bigcup_{C \in \mathcal{S}} C = \mathcal{I}$. Let $u_i, i \in \mathcal{I}$ denote the *payoff* for agent i. Then $\boldsymbol{u}$ is called *individually rational* iff $\forall i \in \mathcal{I} : u_i \geq \boldsymbol{v}(\{i\})$ (i.e. i is better off in $(\mathcal{S}, u)$ as it would be by itself) and *efficient* iff $\forall C \in \mathcal{S} : \sum_{i \in C} u_i = \boldsymbol{v}(C)$ (i.e the overall value of the coalition is distributed completely amongst the agents forming the coalition).

Individual rationality and efficiency provide minimal constraints allowing rational agents to agree with a configuration. To also ensure agents do not have an incentive to be in another coalition, a configuration must also satisfy a chosen *stability concept* which defines this incentive.

In this paper we choose the *Kernel* because of a number of favorable properties as compared with other stability concepts; contrary to the Core or the Shapley value, it is *always non-empty*, i.e a solution always exists (even for non-superadditive games) [7]. In particular, for every coalition structure for which there exists at least one individually rational payoff distribution, there also exists a payoff distribution such that the resulting configuration is Kernel-stable. It is however to be noted that multiple such payoff distributions might exist.

Also, polynomial variants of this concept have been proposed in the literature (see e.g. [11]) which avoid the otherwise exponential complexity of computing solutions and thus is computationally tractable. This is achieved by limiting the maximum allowed coalition size. We however omit to apply this variant, because in this paper, the focus lies on other aspects of the coalition formation process. We thus use the traditional definition of the Kernel.

The *Kernel* of a cooperative game $(\mathcal{I}, \boldsymbol{v})$ with respect to a given

coalition structure $\mathcal{S}$ is a set of configurations $(\mathcal{S}, u)$ wherein each pair of agents i, k in each coalition $C \in \mathcal{S}$ is in equilibrium. That is the case if the agents cannot outweigh each other in $(\mathcal{S}, u)$ by having the option to get a better payoff in coalition(s) *not* in $\mathcal{S}$ excluding the opponent agent (agent i outweighs k, if $s_{ik} > s_{ki}$ and $u_k > w_i(C)$). The *surplus* of agent i with respect to the opponent k in a given configuration $(\mathcal{S}, \boldsymbol{u})$ is $s_{ik} = \max_{C \in 2^{\mathcal{I}}}\{\epsilon(C, u) : i \in C \text{ and } k \in \mathcal{I}/C\}$, where $\epsilon(C, u) = \boldsymbol{v}(C) - \sum_{i \in C} u_i$ denotes the *excess* of alternative coalitions C.

Now, the above model is sufficient to compute values of coalitions if we assume the agents have perfect information about the valuations and costs of tasks, as well as the matching of requested and offered tasks. Moreover, it is assumed, in computing Kernel-stable payoffs, that the agents executing tasks will perform reliably and the side-payments will actually be implemented. However, as already argued, many of these assumptions are inappropriate for realistic MAS and, therefore, in what follows we describe our TKCF mechanism in which these assumptions are relaxed.

4. TRUSTED KERNEL-BASED COALITION FORMATION

In this section we describe our TKCF mechanism which consists of four main stages: 1) a communication stage 2) a CF stage 3) a payment execution stage, and 4) a task execution and evaluation stage. We will first detail the generic trust model (adapted from [3]) used by the agents in order to choose the most reliable coalition partners.

4.1 Properties of the Trust Model

Many computational trust models have been developed to allow agents to choose their most trustworthy interaction partners. However, at their most fundamental level, these models can be viewed as alternative approaches for achieving the following properties:

1. The trust measure of an agent i in an agent j depends both on i's perception of j's POS and on the perception of other agents about j's POS. This latter point encapsulates the concept of *reputation* whereby the society of agents generally attributes some characteristic to one of its members by aggregating some/all the opinions of its other members about that member. Thus, each agent can consider this societal view on other members when building up its own measure of trust in its counterparts [8]. Specifically, the trust of agent i in its counterpart j, $t_i^j \in [0, 1]$, is given by a function, $g : [0, 1]^{|\mathcal{I}|} \rightarrow [0, 1]$, (which, in the simplest case, is a weighted sum) of all POS measures sent by other agents to agent i about agent j as shown below:

$$t_i^j = g(\{\eta_1^j, \ldots, \eta_i^j, \ldots, \eta_N^j\}) \quad (3)$$

where $\eta_i^j \in [0, 1]$ is the POS of agent j as perceived by agent i and g is the function that combines both personal measures of POS and other agents' measures. In general, trust models compute the POS measures over multiple interactions. Thus, the level of success recorded in each interaction is normally averaged to give a representative value (see [8] for a general discussion on trust metrics).

2. Trust results from an analysis of an agent's POS in performing a given task. The more successful, the more trustworthy it is. Thus, the models assume that trust monotonically increases with POS. Therefore, the relationship between trust and POS is expressed as: $\frac{\partial t_i^j}{\partial \eta_i^j} > 0$, where t_i^j is the trust of i in agent j and η_i^j is the actual POS of agent j as perceived by i.

Given the above, agents can update their trust rating for another agent each time they perceive the execution of a task (both by recording their view of the success of their counterpart and by gathering new reports from other agents about it). Thus, if an agent's POS does not change, the trust measure in it should become more precise as more observations are made and received from other agents.

4.2 The Coalition Formation Mechanism

Given the trust model, we now introduce a CF protocol that uses trust to compute Kernel stable payoffs and enforces incentive compatibility in all aspects of the process. In the following sections, we describe each TKCF stage in detail.

4.2.1 Communication

This covers the protocol agents use to exchange valuations, costs, and trust values with one another so that no information asymmetry can exist among them such that one agent can find exploit another (which would make the mechanism unattractive to potential participants). Perhaps the easiest way of achieving this is to ensure that all agents get information about these variables at the same time. Otherwise, agents can simply wait for messages about other agents' valuations and costs, analyse these and, in turn, transmit their own valuations and costs such that the latter exploit the agents that have already transmitted their private information. To achieve such simultaneous information revelation, we adapt the common Data Encryption Standard (DES) cryptographic technique [10] to build our communication protocol. Specifically, we assume that each agent has a unique key e_i (randomly chosen) that allows it to encrypt a message (e.g. containing information about valuations and costs) using a commonly known function enc. The message can only be decrypted using that key and inverting the function enc^{-1}. The protocol is as follows:

1. All agents transmit $enc(\langle \boldsymbol{\tau}, v_i, c_i, \boldsymbol{\eta}_i, g_i(.)\rangle, e_i)$. This means that they encrypt their private information with their key e_i. Then, this encrypted message is sent to all agents directly (it is reasonable to assume here that all agents are directly connected to each other).

2. All agents confirm to all other agents that they have received all encrypted messages from all the other agents. This means that for I agents, each one needs to receive $I - 1$ encrypted messages and send a confirmation of this to all others.

3. When $I - 1$ confirmations (of the reception of $I - 1$ messages) have been received by each agent, all agents send their key e_i to all agents in the population. Then all agents can use this key to decrypt received messages simultaneously using $enc^{-1}(enc(\langle \boldsymbol{\tau}, v_i, c_i, \boldsymbol{\eta}_i, g_i(.)\rangle), e_i)) = \langle \boldsymbol{\tau}, v_i, c_i, \boldsymbol{\eta}_i, g_i(.)\rangle$.

The above protocol guarantees that there is no information asymmetry between any pair of agents in the population. Note that the agents need to obtain $I - 1$ confirmations before sending their keys since, doing otherwise, results in an information asymmetry that could lead to agents being exploited. For example, let agent A send its (encrypted) private information to agents B and C, while B sends its private information to A and C, and C only sends its private information to B. Then, let A send its key to B, and B responds by sending its key to A. C then sends its key to B and gets B's key in return. Now, C can analyse its own information and B's information in order to select valuations, costs, and trust vectors that could

allow it to exploit unfairly both A and B. This happens because C can calculate what it can profitably reveal (i.e. its valuations and costs) to A since A does not already possess C's encrypted private information while C already has A's private information which it can no more change (i.e. there exists an information asymmetry). To avoid this, our protocol forces agents to wait for $I-1$ confirmations each time private information is shared, and ensures that all agents have the same informations.

4.2.2 Kernel Stable Solution Computation

We now provide a protocol that lets the agents achieve a Kernel-stable configuration given the information they obtained by executing the communication protocol of section 4.2.1. As has been stated in section 3, there generally exist multiple coalition structures for which Kernel-stable solutions can be found. In the proposed protocol, a coalition structure which maximizes the sum of the values (sometimes also called *social welfare*) of the formed coalitions is chosen. We consider this a favorable approach with respect to the experimental evaluation (see section 6), because it enables us to compare the quality of the generated coalition structures to a theoretical optimum. But there exist also other, more individual agent or coalition centric coalition structure generation approaches (as e.g. proposed in [11]).

Now, since there might exist multiple optimal coalition structures, task assignments in individual coalitions, and kernel stable payoff distributions for a given coalition structure we introduce a function to allow the agents to jointly make unambiguous choices. We therein assume that each agent possesses a strictly ordered list $L^{\mathcal{I}}$ of all agents in $\mathcal{I}$. This list could, for example, be obtained by the agents' joining order in the system, but since the exact method is not important here, we simply consider it as given.

Let p be a task assignment, coalition structure, or payoff distribution, let p_i denote that p was computed by agent i and let $P := \langle p_1, \ldots, p_{|\mathcal{I}|} \rangle$. Then let $Choose(P)$ return p which was computed by the greatest number of agents. If there are more than one such elements, among them choose the one which was computed by an agent which is considered lowest by $L^{\mathcal{I}}$. To achieve a kernel stable configuration which maximizes the sum of the coalition values, each agent $i \in \mathcal{I}$ performs:

1. Compute expected coalition values; for each $C \subseteq \mathcal{I}$ do:
   (a) Compute an optimal task allocation α_C^i for C and send it to each other agent $j \in C$; receive all α_{C_j}. $P^{\alpha} := \langle \alpha_{C_1}, \ldots, \alpha_{C_{|\mathcal{I}|}} \rangle$; determine $\alpha_C^* := Choose(P^{\alpha})$.
   (b) Given the trust values about agents, for each coalition $C \in \mathcal{S}$ assess the *expected local worths* $\underline{w}_k(C)$ for all agents in $k \in C$:
   $$\underline{w}_k(C) := \sum_{\tau_k^j \in \alpha_C^*} t_k^j \times v_k(\tau) - \sum_{\tau_j^k \in \alpha_C^*} c_k(\tau) \quad (4)$$
   (c) Compute the overall *expected coalition value*:
   $$\underline{\boldsymbol{v}}(C) = \sum_{k \in C} \underline{w}_k(C) \quad (5)$$
2. Find a coalition structure $\mathcal{S}_i$ such that $\sum_{C \in \mathcal{S}_i} \underline{\boldsymbol{v}}(C)$ is maximised. Send $\mathcal{S}_i$ to each other agent $j \in \mathcal{I}$ and receive all other $\mathcal{S}_j$. Let $P^{\mathcal{S}} := \langle \mathcal{S}_1, \ldots, \mathcal{S}_{|\mathcal{I}|} \rangle$; determine $\mathcal{S} := Choose(P^{\mathcal{S}})$
3. Compute a kernel stable payoff distribution $\boldsymbol{u}_i$ for $\mathcal{S}$.
4. Send $\boldsymbol{u}_i$ to all other agents j and receive all $\boldsymbol{u}_j$.
5. Determine $\boldsymbol{u} := Choose(P^{\boldsymbol{u}})$

After completing the execution of this protocol, each agent is assigned to a coalition and a payoff, which completes the traditional coalition formation process. However, these coalitions and payoffs still have to be implemented in order to actually realize the solution. While the task execution performace of the agents is measured via the trust values, it is still unclear how to enforce the execution of the side-payments resulting from the solution. This is covered in the following section.

4.2.3 Payment Execution

We now develop a payment protocol which provides the incentives to the agents to faithfully implement it so as to ensure that each agent $i \in \mathcal{I}$ derives the payment m_i. As explained in section 1, our protocol specifies *what* monetary transfers are made between agents in a coalition and *how* these occur, rather than computing the actual payoffs (which is dealt with in section 4.2.2).

Our protocol initially involves the creation of $|\mathcal{S}|$ strictly ordered lists for each coalition in the stable configuration computed in section 4.2.2. With $PL^C = \{1, \ldots, k, \ldots, K\}$ we denote the list of all agents in a coalition C (hence $K = |C|$) with agents sorted in descending order of the difference, $u_i(C) - w_i(C)$. Ties are broken such that an agent in PL^C gets a higher index if it has a higher index in the list $L^{\mathcal{I}}$. Thus agent 1 in PL^C corresponds to the agent which has the maximum $u_i(C) - w_i(C)$. Since all information required to form this list has already been transmitted in the communication stage (described in section 4.2.1), PL^C is commonly known to all agents in $|C|$. Now, our protocol intuitively works by cascading payments between agents, with an agent providing a payment before it receives one. The sorted list allows us to condition payments such that agents always make positive transfers to each other. The transfer m_{k+1}^k each agent $k+1$ makes to agent k is computed as:

$$m_{k+1}^k = u_k - w_k(C) + m_k^{k-1} \quad (6)$$

The following specification of the payment protocol is designed for the case when $|C| \geq 3$ (figure 1 graphically depicts the protocol when all agents implement it faithfully with each step below corresponding to the labelled steps in the figure). Note that the payment protocols for the cases when $|C| \leq 2$ are trivial. When $|C| = 1$, no transfers occur and when $|C| = 2$ a single transfer occurs between the two agents.

I. The protocol is initiated by agent K sending an encrypted but verifiable payment, $enc(m_K^{K-1}, e'_K)$, to agent $K-1$. That is, agent $K-1$ can check the amount but cannot access it. This is what secure digital cash achieves and can be intuitively seen as an unbreakable glass safe [10]. Agent $K-1$ then broadcasts the message $\langle start\_payment \rangle$ to all agents in the list if the value of the encrypted transfer from agent K to $K-1$ is according to equation 6. Otherwise, agent $K-1$ transmits $\langle \widehat{m}_K^{K-1}\_rec \rangle$ (which means payment $\widehat{m}_K^{K-1}$ has been received) and the coalition dissolves and a new coalition structure is computed without agent K.

II. Each agent $k+1$ ($k \in PL^C \setminus K$) then pays agent k according to equation 6 if it receives the message $\langle m_k^{k-1}\_rec \rangle$ from agent $k-1$. Otherwise, if it receives message $\langle \widehat{m}_k^{k-1}\_rec \rangle$ where $\widehat{m}_k^{k-1} \neq m_k^{k-1}$, it then decides according to whether it has also received a message $\langle \widehat{m}_{k-1}^{k-2}\_rec \rangle$ from agent $k-2$. If it has received such a message and $m_{k-1}^{k-2} - \widehat{m}_{k-1}^{k-2} + \delta = m_k^{k-1} - \widehat{m}_k^{k-1}$, it then implements the transfer according to

equation 6. Otherwise, it then implements the following transfer:

$$m_{k+1}^k = u_k - w_k(C) + \widehat{m}_k^{k-1} - \delta \quad (7)$$

where $\delta \in \Re^+$ is a penalty applied for wrong payment (which may happen if the agent is irrational). The transfer $m_2 1$ is initialised to be $u_1 - \underline{w}_1(C)$.

III. Upon receipt of payment m_{k+1}^k, each agent k ($k \in PL^C \setminus K$) transmits message $\langle m_k^{k-1}\_\rangle$ to agent $k+2$. However, if the payment received is $\widehat{m}_{k+1}^k$ where $\widehat{m}_{k+1}^k \neq m_k^{k-1}$, agent k then transmits $\langle \widehat{m}_{k+1}^k\_rec\rangle$ to both agents $k+2$ and $k+3$.

IV. The protocol is different for the last three agents since these agents control the message which will start the task execution stage. If agent K receives the message $\langle m_{K-1}^{K-2}\_rec\rangle$ (or if it receives message $\langle \widehat{m}_k^{k-1}\_rec\rangle$ from $K-2$ and it also receives $\langle \widehat{m}_{K-2}^{K-3}\_rec\rangle$ from agent $K-3$ and $m_{K-1}^{K-2} - \widehat{m}_{K-1}^{K-2} - \delta = m_{K-2}^{K-3} - \widehat{m}_{K-2}^{K-3}$) it then transmits the key and broadcasts the message $\langle key\_sent\rangle$. If $w_{K-1} \geq 0$, agent K transmits the key to agent $K-1$ who then broadcasts the message $\langle start\_tasks\rangle$. Otherwise, it then transmits the key to the agent n such that $w_n \geq 0$ and has the highest index in PL^C. This agent then transmits the key to agent $K-1$ and broadcasts the message $\langle start\_tasks\rangle$. If ever agent K receives $\langle \widehat{m}_k^{k-1}\_rec\rangle$ and it detects a deviation by $K-1$, agent K then broadcasts the message $\langle no\_key\_sent\rangle$.

We now show that a rational agent would not find it in its best interest to deviate from the payment protocol, i.e. it will implement the payments specified by equation 6 and would not send erroneous messages once it has received the payments.

THEOREM 1. *Every rational agent in the game follows the payment scheme faithfully.*

PROOF. (Sketch) We prove the above theorem by comparing the utility that an agent derives when following the protocol faithfully to that when it deviates.

The net utility an agent derives when following the protocol faithfully is its payoff which can be rewritten from equation 6 as :

$$u_k(C) = w_k + \begin{cases} m_2 1, & k = 1 \\ m_{k+1}^k - m_k^{k-1}, & \forall k \in PL^C \setminus \{K, 1\} \\ -m_K^{K-1}, & k = K \end{cases} \quad (8)$$

Now consider each agent's opportunity to defect as the protocol proceeds (assuming all other agents have followed it till that point). At the beginning, agent K can deviate by not sending the correct value in the encrypted payment. Then, this is detected by agent $K-1$ and thus the coalition does not start. Agent K then derives a utility of $u_K(K)$ ($u_K(K) < u_K(C)$ by the definition of kernel stability) and thus will not deviate.

On the second step, agent $K-1$ may deviate by not acknowledging the payment and not sending the $\langle start\_payment\rangle$ message. Again, since $u_{K-1}(K-1) < u_{K-1}(C)$, agent $K-1$ does not deviate.

On the third step, agent 2 can deviate by sending an incorrect payment, $\widehat{m}_2^1$, to agent 1. In this case, agent 1 sends to agent 3 and agent 4 the message $\langle \widehat{m}_2^1\_rec\rangle$ and agent 3 then pays agent 2 the amount $u_k(C) - w_k(C) + \widehat{m}_2^1 - \delta$. As a result, the net transfer to agent 2 is $u_k(C) - w_k(C) - \delta$ which is strictly less than in equation 8. Thus, agent 2 cannot benefit by providing a payment $\widehat{m}_2^1 \neq m1_2$. Notice also that by the protocol, agent 3 derives a benefit of δ when applying the correct penalty and will not get charged by agent 4 who has been informed of agent 2's deviation. However, if agent 3 deviates and does not apply the correct penalty, then agent 4 will also penalise it. Notice also that if agent 1 receives the correct payment, it can still deviate by misreporting this payment. Furthermore, the agent is indifferent between all the messages it can send (in a scenario where the coalition game is run only once) once it has received its correct payment. However, in a repeated coalition game (which is the case we consider here), this would amount to penalising a good payer or not penalising a bad payer, which is clearly not what an agent would like to do here. The same argument as used for agents 1 and 2 can now be used for all other payments between agents until agent $K-1$.

Now if agent $K-1$ deviates when paying, then agent $K-2$ will report this deviation to agent K who will withhold the key. Then agent $K-1$ will derive a net payment of $-\widehat{m}_{K-1}^{K-2}$ which is less than the amount it derives in equation 8. Agent K can also deviate by sending the wrong key. In this case agent K does not derive any higher utility by so doing. Finally the agent who has to send the message $\langle start\_tasks\rangle$ can deviate by not sending it. However, the agent sending it (either agent $K-1$ or some other agent) would not find any utility in doing so since it gains a positive w_k when the coalition tasks are performed. □

4.2.4 Task Execution

Once the payment execution phase is completed (i.e. after the agents have received the two broadcasted messages $\langle key\_sent\rangle$ and $\langle start\_tasks\rangle$ or the single message $\langle no\_key\_sent\rangle$), the agents start performing their tasks. All agents deriving value from a task $\tau \in \mathcal{T}$ then measure the POS values of the respective executing agents, and the next round of CF starts.

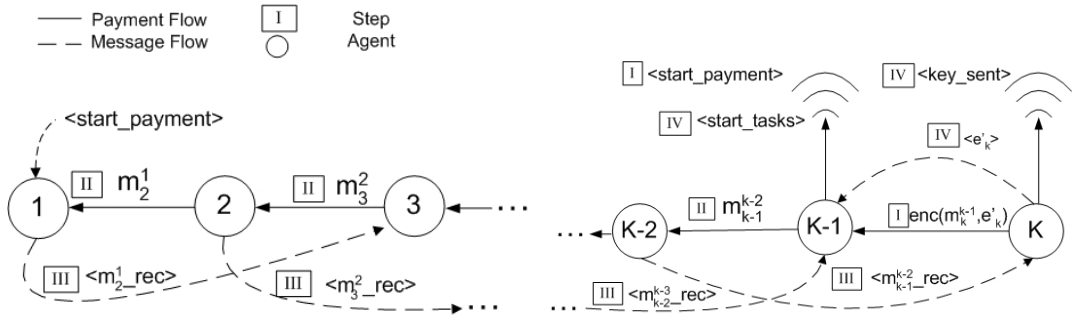
5. RATIONALITY OF TRUTH-TELLING

In our model, the coalition values are defined as the sum of the local worths of the agents in a coalition. Each agent's local worth is determined by its reported valuation of requested tasks, the costs of its offered tasks and its trust values. Now according to the Myerson-Sattherthwaite impossibility result, no incentive-compatible, budget-balanced (i.e. all payments between agents sum to zero, which translates to efficiency in terms of CGT) and individually rational mechanism can exist [7]. Hence, here we instead prove that our mechanism achieves *near incentive-compatibility*, with which we define as the fact that agents cannot determine *how* to lie profitably, even if the possibility to do so theoretically exists.

Let $\hat{x}$ denote the *reported value* for a value x, which could represent the valuation, cost, or POS. Now suppose that for an execution of the TKCF in order to achieve a solution for a game $(\mathcal{I}, \boldsymbol{v})$ there exists at least one x with $\hat{x} \neq x$. Then the game that is actually solved is different from the original game $(\mathcal{I}, \boldsymbol{v})$. We call this new game $(\mathcal{I}, \boldsymbol{v}')$.

THEOREM 2. *In the TKCF, no agent a is able to determine values $\hat{x} \neq x$ to report that will unjustifiably increase a's payoff wrt the original game $(\mathcal{I}, \boldsymbol{v})$.*

PROOF. (*Sketch*) We first look at reporting false task valuations. Suppose an agent i reports $\hat{v}_i(\tau) = v_i(\tau) + r, r \in \mathbb{R}$, as its valuation of task τ. Let α_C^* and $\alpha_C^{*'}$ be the task allocations established by the TKCF for C in $(\mathcal{I}, v)$ and $(\mathcal{I}, \boldsymbol{v}')$, respectively, and $w^{e'}$ be the expected local worths in $(\mathcal{I}, \boldsymbol{v}')$. Now, if $r \geq 0$, $\forall C \tau_i \in \alpha_C^*$, also $\tau_i \in \alpha_C^{*'}$ with $\underline{w}_i'(C) = \underline{w}_i(C) + r$ holds. $\forall C \tau_i \notin \alpha_C^*$, either $\tau_i \notin \alpha_C^{*'}$ with $\underline{w}_i'(C) = \underline{w}_i(C)$ or $\tau_i \in \alpha_C^{*'}$ with $\underline{w}_i'(C) = \underline{w}_i(C) + r_2,\ r_2 \leq r$. The latter can happen if



(a) For agents $1, \ldots, K-3$ (b) For agents $K-2, K-1$ and K

Figure 1: Payment Protocol

there are other agents whose valuation of τ is between a's real and reported valuations. Because the expected coalition values are the sum of the local worths of the respective coalition members, we get $\forall C, i \in C, \boldsymbol{v}'(C) = \boldsymbol{v}(C) + r_C$ with $r_C \in [0, r]$.

However, if $r < 0$, it can analogously be shown that $\forall C, i \in C$, $\boldsymbol{v}'(C) = \boldsymbol{v}(C) + r_C$ with $r_C \in [r, 0)$. But, because of our encryption-based communication, no agent knows any coalition values before the keys are sent around. Thus, there is no way for i to know which coalition values will increase by which amount when reporting any $\hat{v}_i(\tau) \neq v_i(\tau)$. Obviously, reported costs $\hat{c}_i(\tau) \neq c_i(\tau)$ have a symmetric impact on the coalition values, thus we will not show this in detail.

As mentioned earlier, the agents are expected to actually bring their local worths into their coalition. So, let $(\mathcal{S}, u)$ and $(\mathcal{S}', u')$ be kernel stable configurations for $(\mathcal{I}, v)$ and $(\mathcal{I}, \boldsymbol{v}')$, respectively. Suppose agent $i \in C \in \mathcal{S}$, and $i \in C' \in \mathcal{S}'$. Thus, the *expected profit* $p_i^{u',u}$ of i in u' as opposed to u is given by $p_i^{u',u} = u'_i - u_i - (\underline{w}'_i(C') - \underline{w}_i(C'))$. To obtain a positive profit $p_i^{u',u} > 0$, i must therefore ensure that:

$$u'_i - u_i > \underline{w}'_i(C') - \underline{w}_i(C') \tag{9}$$

But since we assume that the agents have no a priori knowledge about each others' valuations of tasks and costs, i's local worths, and, in particular, the differences between them and possible kernel stable solutions are unknown to i.

For the case where $C' = C$, we will now show (9) can only be true if the surplus of i also changes in a specific way:

LEMMA 1. *In the case where the coalition structure does not change due to the reporting of false values by i, generally $p_i^{u',u} \geq 0$ only if*

$$\frac{1}{|C|-1} \sum_{k \in C, k \neq i} s'_{i,k} - s_{i,k} \geq w_i^{e'}(C) - \underline{w}_i(C) = r_C$$

where s' denotes the surplus for the game $(\mathcal{I}, \boldsymbol{v}')$.

PROOF. We first consider the case for $r_C > 0$. For any kernel stable configuration $(\mathcal{S}, u)$ with $C \in \mathcal{S}$ for $(\mathcal{I}, v)$ with $\forall k \in C : u_k > \boldsymbol{v}(\{k\})$ it was shown in [1] that if C is the only coalition with $u'_i = u_i + r_C$ and $\forall k \in \mathcal{I}, k \neq i, u'_k = u_k$, $(\mathcal{S}, u')$ is not kernel stable for $(\mathcal{I}, \boldsymbol{v}')$. That is, the additional amount brought into C by i as opposed to the game $(\mathcal{I}, v)$ is not completely paid back to i in a kernel stable configuration $(\mathcal{S}, u^+)$ for $(\mathcal{I}, \boldsymbol{v}')$. More specifically, it was shown that $s'_{i,k} = s_{i,k} - r_C < s_{i,k}$ in $(\mathcal{S}, u')$. Let $C^{s_{\cdot,\cdot}}$ denote a coalition with $e(C, u) = s_{\cdot,\cdot}$ for a given surplus. Now consider the case where for one agent $k \in C \setminus \{i\} : r_{C^{s_{i,k}}} < r_C$ and for all other agents $j \in C \setminus \{i, k\} : r_{C^{s_{i,k}}} = r_C$. Then for u' with $u'_i = u_i + r_C$, we have $s'_{i,j} = s_{i,j} + r_C - r_C = s_{i,j}$, but $s'_{i,k} = s_{i,k} + r_{C^{s_{i,k}}} - r_C < s_{i,k} + r_C - r_C = s_{i,k}$. Thus, even if the other agents' surpluses do not increase, i (besides other agents) would have to make some transfer payment to k in order to obtain a kernel stable configuration. In the case where surpluses of other agents in C do increase, i's kernel stable payoff obviously further decreases.

For $r_C < 0$, we just have to consider the reverse step from the modified to the original game. Then, because of the above, $p_i^{u,u'} \geq 0$ only if:

$$\frac{1}{|C|-1} \sum_{k \in C, k \neq i} s_{i,k} - s'_{i,k} \geq -r_C$$

$$\Leftrightarrow \frac{1}{|C|-1} \sum_{k \in C, k \neq i} s'_{i,k} - s_{i,k} \leq r_C$$

Now, since $p_i^{u,u'} = -p_i^{u',u}$, the lemma then follows. □

But i cannot infer whether the average of its surplusses will change by a greater or smaller amount than the value of the coalition that i will join. Thus, i cannot determine if the condition for it to obtain an unjustified profit is met for any reported $\hat{v}_i$ or $\hat{c}_i$.

In the case where $C' \neq C$, we have $\boldsymbol{v}'(C') - \boldsymbol{v}(C') > \boldsymbol{v}'(C) - \boldsymbol{v}(C)$ because otherwise the optimal coalition structure would not have changed. But because of lemma 1, this does not allow any conclusion about the size of u'_i. In fact, since C' now belongs to the optimal coalition structure, it is quite unlikely that i's average surplus in the game $(\mathcal{I}, \boldsymbol{v}')$ is increased by a greater or decreased by a smaller value than $\boldsymbol{v}'(C') - \boldsymbol{v}(C')$ with respect to a kernel stable solution $(\mathcal{S}', u^+)$ for $(\mathcal{I}, v)$. Thus, $p_i^{u',u} \geq 0$ is only likely if $u'_i - u_i > \boldsymbol{v}'(C') - \boldsymbol{v}(C')$. But since i does not a priori know anything about the game, it is not able to decide if such a coalition C' exists. Moreover, since i cannot manipulate its local worths independently of each other, it has no facility to enforce the formation of C'.

However, there is also the possibility for reporting false POS values $\hat{\eta}_i^k = \eta_i^k + r$ about another agent k. This changes other agents' valuation of tasks executed by k and thus will increase or decrease their local worths in coalitions containing k. Note that this also includes i. Thus, this case is equivalent to each agent in $\mathcal{I} \setminus \{k\}$ reporting false task valuations for tasks executed by k. But then, as shown above, each agent, and, in particular i, has likewise chances of obtaining a profit or a loss.

We have thus shown in this section that no agent can determine

the actual way in which to lie so as to make an unjustified positive improvement over what it could achieve when reporting truthfully.

6. EXPERIMENTAL EVALUATION

Having ensured that TKCF incentivises agents to reveal their true costs and valuations and that they execute the payments that are due, we now turn to evaluating the effectiveness of TKCF in choosing the coalitions where the most reliable agents are selected to execute certain tasks. To this end, we aim to see whether TKCF can use the trust model defined in section 4.1 in order to evaluate the reliability of agents over multiple interactions. Here we consider a super-additive game, but restrict the maximum coalition size (which remains non-trivial) in order to analyze the TKCF's behaviour when finding an optimal coalition structure. This size is fixed to half of the number of agents in our case. The agents' valuations and costs are taken from a uniform distribution between 0 and 1. The agents' POS are determined *a priori* and their actual success after each coalition executes tasks is taken from a uniform distribution whose mean is equal to their POS. Then, according to our trust model, the agents' reported POS in each other are summed using a weight vector to give the actual trust values. Given this, a number of agents, six in this case, are allowed to form coalitions of a maximum of 3 agents. To simplify the analysis, each agent is allowed to execute more than one task and asks for only one task to be completed. However, agents might request different tasks and vary valuations and costs in each game. Thus, in each iteration, a solution to a possibly different game is to be found. Although this might increase the number of iterations until the correct POS are determined, and thus the correct solutions are found, we consider this a more realistic situation than the case of repeating just one game all the time.

Given that the payoffs described in section 4 are calculated according to the expected value (resulting from the trustworthiness of agents) of coalitions (see equation 4), we postulate the following hypothesis:

H1:*The agents' payoffs converge to those reflecting their actual POS in the long run. Given this, the coalition structure $\mathcal{S}$ chosen converges to $\mathcal{S}^*$ which maximises the overall value $\boldsymbol{v}(\mathcal{S}) = \sum_{C\in\mathcal{S}} \boldsymbol{v}(C)$.* To test this hypothesis we performed an experiment given the above settings and recorded each agent's payoff and determined the ratio $\frac{|u_i^* - u_i|}{u_i}$ which indicates the distance of the calculated payoff u_i from the exact payoff u_i^*. We also recorded the ratio $\frac{\boldsymbol{v}(\mathcal{S})}{\boldsymbol{v}(\mathcal{S}^*)}$ to check whether we actually chose the most valuable coalition structures. We repeated the CF game 200 times over which trust measures were refined each time the tasks were executed. The results are shown on figure 2. As can be seen, the difference between the payoffs converge to 0 indicating that the exact payoffs are chosen in the long run. Moreover, an optimal coalition structure ($\mathcal{S}^*$) is chosen well before the payoffs stabilise (when the trust is exactly determined after 200 interactions). This means that even though an optimal coalition structure has been chosen (after around 167 interactions in this case), the payoffs are still affected by slight deviations of the trust perceived by agents.We used ANOVA (Analysis Of VAriance) to determine whether there were any significant differences between means of $\frac{|u^* - u|}{u^*}$ of the agents. Thus, it was found that for 10 samples of 200 games that $p = 0.5534$ for $\alpha = 0.5$ such that $p > \alpha$ and the null hypothesis is validated. Also, for the value of $\frac{\boldsymbol{v}(\mathcal{S})}{\boldsymbol{v}(\mathcal{S}^*)}$, it was found that $p = 0.182$ for $\alpha = 0.1$. This validates the null hypothesis in this case since $p > \alpha$, which tells us there is no significant difference between the means of the samples.

We can also note that the results show that as trust is being learnt by all agents, the agents' payoffs may, at times, significantly diverge from the optimal ones (the spikes in the graph), though the size of this occasional divergence decreases over time (due to more precises trust values). In such cases, the spikes are due to u_i^* being very low compared to the difference $u_i^* - u_i$. These, in turn, are due to the sensitivity of the kernel-based payoffs to slight changes in the trust values.

The convergence of the TKCF might seem slow, but taking into account that different games are played and thus different coalitions are formed in each iteration, we consider the result at least reasonable.

7. CONCLUSIONS

In the task allocation via coalition formation domain, we proposed a novel model to compute expected coalition values that account for agents' trust in each others' ability to execute tasks with satisfactory reliability. Instead of specifying a particular trust model, we identified necessary properties of trust models in general in order that they can be soundly applied within this context. Thus, any trust model exhibiting these properties can be used.

We further presented a protocol that allows the agents, based on the expected coalition values, to form kernel stable coalitions. The protocol accounts for every step in the coalition formation process from the communication of individual valuations and costs to the actual execution of side payments and tasks, as well as updating of the trust values. It was experimentally shown that for the realistic case of repeated games with varying task requests, valuations and costs, the computed solutions over time converge to their theoretical optimum. Moreover, it was formally shown that for all communications and payments required by this protocol, it is not rational for any agent to deviate from what we specify. To achieve this, we applied encryption-based communication techniques and developed a sequential payment protocol.

In our proposed mechanism, expected optimal coalition structures and kernel stable solutions are computed and this involves exponential complexities. This was done in order to demonstrate convergence to the theoretical optimum in the experiments. However, we believe that none of our results actually depend on this property, and that polynomial kernel based coalition formation can equally be applied. However further work is needed to confirm this conjecture.

8. REFERENCES

[1] B. Blankenburg and M. Klusch. On safe kernel stable coalition forming among agents. In *Proc. 3rd Int. Conference on Autonomous Agents and Multiagent Systems*, volume 2, pages 580–587, New York, USA, 2004. ACM Press.

[2] G. Chalkiadakis and C. Boutilier. Bayesian reinforcement learning for coalition formation under uncertainty. In *Proc. 3rd Int. Conference on Autonomous Agents and Multiagent Systems*, New York, USA, 2004. ACM Press.

[3] R. K. Dash, S. Ramchurn, and N. R. Jennings. Trust-based mechanism design. In *Proc. 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems*, pages 746–753, New York, USA, 2004.

[4] I. Foster, N. R. Jennings, and C. Kesselman. Brain meets brawn: why grid and agents need each other. In *Proc. 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems*, pages 8–15, New York, USA, 2004.

[5] S. Kraus, O. Shehory, and G. Tasse. Coalition formation with uncertain heterogeneous information. In *Proc. 2nd Int.*

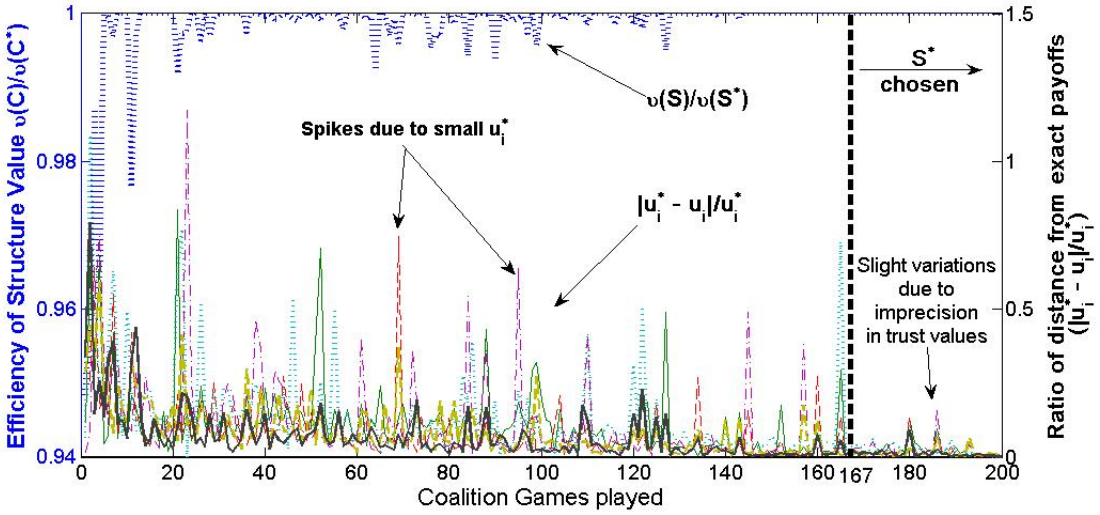


Figure 2: Computing stable payoffs as trust values converge.

Conference on Autonomous Agents and Multiagent Systems, New York, USA, 2003. ACM Press.

[6] C. Ortiz, V. Lesser, and M. Tambe, editors. *Distributed Sensor Networks: A Multiagent Perspective*. Kluwer, 2003.

[7] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, Cambridge MA, USA, 1994.

[8] S. D. Ramchurn, D. Huynh, and N. R. Jennings. Trust in multi-agent systems. *The Knowledge Engineering Review*, 19(1):1–25, 2005.

[9] T. Sandholm and V. Lesser. Leveled commitment contracts and strategic breach. *Games and Economic Behavior*, 35:212–270, 2001.

[10] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[11] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1–2):165–200, 1998.

[12] J. Vassileva, S. Breban, and M. Horsch. Agent reasoning mechanism for long-term coalitions based on decision making and trust. *Computational Intelligence*, 4(18):583–595, 2002.