

Towards Models of Incomplete and Uncertain Knowledge of Collaborators' Internal Resources

Christian Guttman and Ingrid Zukerman

School of Computer Science and Software Engineering
Monash University
VICTORIA, 3800, Australia
{xtg,ingrid}@csse.monash.edu.au

Abstract. Collaboration plays a critical role when a group is striving for goals which are difficult or impossible to achieve by an individual. Knowledge about collaborators' contributions to a task is an important factor when establishing collaboration, in particular when a decision determines the assignment of activities to members of the group. Although there are several systems that implement collaboration, one important problem has not yet received much attention – determining the effect of incomplete and uncertain knowledge of collaborators' internal resources (i.e. capabilities and knowledge) on the outcomes of the collaboration. We approach this problem by building models of internal resources of individuals and groups of collaborators. These models enable a system to estimate collaborators' contributions to the task. We then assess the effect of model accuracy on task performance. An empirical evaluation is performed in order to validate this approach.

1 Introduction

Humans tend to form collaborative groups to achieve goals that are difficult or impossible to attain by individuals. Most people are naturally endowed with capabilities in order to approach the solution of a problem collectively. However, the incorporation of collaboration into complex computational systems (hereafter called agents) is a challenging problem (corroborated by Cohen & Levesque [1991], Kinny *et al.* [1992], Grosz & Kraus [1996], Panzarasa *et al.* [2002]). The interest in the solution of this problem has increased due to technologies such as peer-to-peer and wireless mobile computing (e.g. Ad-Hoc networking, Digital Home, computational grid) and large-scale traffic simulations (e.g. car, train and air traffic).

One important aspect of collaboration in networks of agents or Multi-Agent Systems (MAS) is knowledge about collaborators when activities are planned or performed together. For example, different features of collaborators are modelled in order to achieve flexible team behaviour in military domains [Tambe 1997], establish domain-dependent conventions among agents [Alterman & Garland 2001], determine behaviour based on collaborators' utility-functions [Gmytrasiewicz & Durfee 2001], or predict behaviour of "soccer-agents" under commonly-known domain dynamics [Stone *et al.* 2000] or commonly-known action policies [Kok & Vlassis 2001]. Denzinger &

Kordt [2000] describe how modelled strategies of agents facilitate on-line learning processes.

An implicit assumption made by these systems is that each agent possesses (or can easily derive) complete and accurate models of its collaborators' capabilities, thus simplifying collaboration scenarios. However, this assumption does not always hold, e.g. when agents in established groups are coping with idiosyncratic capabilities of new agents. In this paper, we consider how the relaxation of this assumption affects task performance. To this effect we build models of collaborators' capabilities (employed by each agent) and perform empirical studies that assess the impact of model accuracy. Our approach is particularly applicable for agents which have to use these models in a decentralized fashion, viz. when there is no overarching central administration that can supervise the overall collaboration due to technical or computational restrictions.

The central issue addressed in this paper is how the accuracy of these models influences the task performance of decentralized agents (where the evaluation of the task performance is based on the extent of the goal achievement, Section 2.1.2). A major consideration in our approach is the reduction of overhead costs (i.e. transaction and resource costs) resulting from the introduction of these models. For our empirical study, we focus on operating parameters such as the memory boundedness of individual agents and the model-updating and group-decision-making processes.

To illustrate the motivation of this approach let us consider an example where several parties have decided to collaborate and merge their system resources in order to achieve a joint goal, e.g. when several distinct groups of mobile devices (i.e. the devices in each group are made by the same company and the devices in different groups are made by different companies) intend to establish a decentralized telecommunication infrastructure. Assume that all agents have a common language, common interaction protocols, a joint intention and a shared ontology of the activities in this domain, but that the capabilities of each mobile device in each group differ significantly and are not known to the other groups of mobile devices prior to the collaboration. Thus, an agent that collaborates well with mobile devices of its own kind may encounter problems when collaborating with agents in other groups. In order to support this collaboration, a structure is needed which enables the distinct groups of agents from different companies to collaborate with each other. Hence, we propose to build models of internal resources of other devices, and integrate these models into each agent in a team.

The remainder of this paper is structured as follows. In Section 2, we present our notation for collaborative scenarios, which includes the tasks, agents and algorithms, and in particular the agents' internal resources and the models of collaborators' resources. Additionally, we define the overhead costs for these scenarios. In Section 3, we describe our empirical evaluation, including the description of a Surf Rescue (SR) scenario, the procedure for simulating agent collaboration, the definition of various simulation parameters, and the results and analysis of several simulations. In Section 4, we present our conclusions and outline future work.

2 Notation

We first describe terms and specifications which provide the basis for an implementation and formal analysis of collaborative agents which perform tasks together. Second, we

provide the definitions for the overhead costs that can arise in collaboration scenarios due to the use of models of collaborators.

Groups of agents are referred to in uppercase letters (e.g. A), individual agents are referred to in uppercase letters with an index (e.g. A_i), and subgroups of agents in uppercase letters with tilde and an index (e.g. \tilde{A}_j). Activities are referred to in lowercase letters with an index (e.g. a_i) and parameters of activities are referred to in Greek letters with an index (e.g. α_i). Cardinalities of sets are referred to in lowercase letters (e.g. m, n, q, r, s).

2.1 Collaborative Scenario S

Definition 1. A collaborative scenario S is denoted by a tuple with four elements, $S = \langle E, T_E, A, P_A \rangle$, where E corresponds to an environmental state space under which a task specification T_E is defined, A denotes a group of agents that is assigned to a task T_E , and P_A is a policy which assists the agents in A to make decisions together.

A collaborative scenario assumes that agents share a joint intention, use the same language and ontology to communicate, and are selfless¹. We define the elements of a collaborative scenario S throughout the remainder of this section.

2.1.1 Environmental State Space E

Definition 2. An environment E is a state space described by first-order logic facts and predicates (i.e. properties of facts and relations among them). A state $e \in E$ describes the current status of these facts and predicates at a particular step in the collaboration.

For example, E consists of a set of locations $L = \{loc_1, \dots, loc_k\}$ and a set of objects $O = \{obj_1, \dots, obj_j\}$. Each location is described by static coordinates, e.g. $loc_i = (x_i, y_i, z_i)$. Each object is described by features such as location ($loc(obj_j) = loc_i$), weight ($weight(obj_j)$) and size ($size(obj_j)$). Consider a “table environment” which consists of two locations: $house_a$ and $house_b$ and two objects: a car and a table. An example for a state e is $e = (at(car, house_a) \wedge in(table, car))$, viz. the car is located at $house_a$ and the table is located in the car.

2.1.2 Task T_E

Definition 3. A task T_E is a tuple with two elements $T_E = \langle EC_T, MS_T \rangle$, where

1. EC_T denotes a set of Evaluation Criteria of a Task $EC_T = \{ec_1, \dots, ec_n\}$, where $n = |EC_T|$. Each element in EC_T is a function $ec_i : e \mapsto [0..1]$ which maps an environmental state e to a real number between 0 and 1, where 0 corresponds to the worst possible performance and 1 corresponds to the optimal performance.
2. MS_T denotes a set of MileStones of a Task $MS_T = \{ms_0, \dots, ms_m\}$, where $m = |MS_T|$, ms_0 represents the initial state (and is satisfied by default) and ms_m represents the goal state of the task. Each milestone has to hold once in order to reach the goal state and thus accomplish the task. The elements in MS_T are partially ordered.

¹ Selflessness is understood as an antonym for selfishness, viz. an agent that is selfless prioritizes the utility of the group rather than its own utility.

For instance, in the “table scenario” the task is to move the *table* from *house_a* to *house_b* by using the *car*. To evaluate how well the task is performed we denote two evaluation criteria. First, the goal of this task should be achieved quickly, i.e. the time to the completion of the milestones (including the initial and goal state) ought to be minimal (expressed by a function min_{time}). Second, the *table* should not be damaged during the performance of the task, i.e. the quality of the completed milestones and the completed goal of the task ought to be maximal (expressed by a function max_{qual}). These two elements are represented as follows:

$$EC_T = \{ec_1, ec_2\} = \{min_{time}, max_{qual}\}$$

The elements of EC_T provide the units which may be needed in further calculations, e.g. min_{time} includes the unit *seconds*.

A plan to achieve milestones is generated by a planner. The table scenario has four milestones, which are described as follows:

$$MS_T = \{ms_0, ms_1, ms_2, ms_3\}, \text{ where}$$

- $ms_0 = (in(table, house_a) \wedge at(car, house_a)),$
- $ms_1 = (in(table, car) \wedge at(car, house_a)),$
- $ms_2 = (in(table, car) \wedge at(car, house_b)),$
- $ms_3 = (in(table, house_b)).$

The task specification T_E (EC_T and MS_T respectively) is known and accessible to all agents in the group during the collaboration.

2.1.3 Agents and Groups of Agents A

Definition 4. A selfless group (or team) of agents A is denoted by a set $A = \{A_1, \dots, A_q\}$ with $q = |A|$, where $A_i \in A : i \in \{1, \dots, q\}$ is an individual agent and $\tilde{A}_j \subseteq A : j \in \{1, \dots, r\}$ denotes a subset of A (with $r = |\wp(A)|$ being the cardinality of the powerset of A).

A group of agents A is assigned to a task T_E , if each agent in the group acts in order to optimize functions $ec_i \in EC_T$, and if the agents perform activities in order to achieve $ms_j \in MS_T$.

We consider two main factors that determine the performance of an activity: Internal Resources (IR) of agents and the Context Parameter (CP). CP depends on the activity. In the table scenario, CP is determined by features of the objects that are manipulated (e.g. the weight and size of the *table*) and other contextual features (e.g. the *table* is lifted for a certain distance). IR are not known to other agents in many collaboration scenarios and thus can be estimated by Models (M).

Definition 5. Let A_i be an agent in set A , so that $A_i = \langle IR_{A_i}, M_{A_i}, RA_{A_i} \rangle$, where

1. IR_{A_i} denotes a measure of the influence of agent A_i 's Internal Resources (e.g. capabilities, such as skills or knowledge) on the performance of activities. Formally, we say that

- $IR_{A_i} : A_i \rightarrow \{act_k : k \in \{1, \dots, s\}\}$ is a set of activity tuples act_k , where s denotes the number of activities that agents can perform.
- $act_k = (a_k, \alpha_1(a_k), \dots, \alpha_n(a_k))$ is a tuple with $n + 1$ elements for each activity a_k , where each α_j is based on $ec_j \in EC_T$ for $j = \{1, \dots, n\}$ (with $n = |EC_T|$).

- $\alpha_l(a_k) : (l, a_k) \rightarrow [0..1]$ represents a “real” internal factor which influences the performance of an activity a_k performed by A_i , where 0 represents the lowest possible performance and 1 represents the highest possible performance.
2. M_{A_i} denotes agent A_i 's Models measuring the influence of subgroups \tilde{A}_j on the performance of activities. Formally, we say that
 - M_{A_i} is a family of models $\{M_{A_i}(\tilde{A}_j) : j = \{1, \dots, r\}\}^2$.
 - Each model $M_{A_i}(\tilde{A}_j) : \{A_i, \tilde{A}_j\} \rightarrow \{\widehat{act}_k : k \in \{1, \dots, s\}\}$ is a set of activity tuples \widehat{act}_k , where s denotes the number of activities that agents can perform.
 - $\widehat{act}_k = (a_k, \alpha_1(\widehat{a}_k), \dots, \alpha_n(\widehat{a}_k))$ is a tuple with $n + 1$ elements for each activity a_j , where each α_j is based on $ec_j \in EC_T$ for $j = \{1, \dots, n\}$ (with $n = |EC_T|$).
 - $\alpha_l(\widehat{a}_k) : (l, a_k) \rightarrow [0..1]$ represents an “estimated” internal factor, which agent A_i believes influences the performance of activity a_k performed by the subgroup \tilde{A}_j , where 0 represents the lowest possible performance and 1 represents the highest possible performance.
 - If a model of \tilde{A}_j is not known to A_i then $M_{A_i}(\tilde{A}_j) = \emptyset$.
 3. RA_{A_i} is the Reasoning Apparatus of A_i which consists of a set of algorithms that enable an agent to act in an environment and to interact with collaborators. The main algorithm is illustrated in Figure 1. Its inputs are: an environment E , a task T , models M_{A_i} , and a policy P_A . The output is a set of updated models UM in the same form as M_{A_i} . The side effects of this algorithm are the actual actions performed by the agents chosen to perform a task.

We first illustrate internal resources IR_{A_i} and models of internal resources M_{A_i} , and towards the end of this section we explain the algorithms in RA_{A_i} .

Let us return to the table scenario, where time and quality are defined as the evaluation criteria. According to Definition 5, IR_{A_i} and M_{A_i} are based on these criteria ($n = 2$). Assume that two agents A_1 and A_2 perform the “table task” and both agents are able to perform two activities: *lift* (which moves a “liftable” object from one location to another location) and *drive* (which moves a “driveable” object from one location to another location). The internal resources for A_1 are as follows:

$$IR_{A_1} = \{(a_1, \alpha_{time}(a_1), \alpha_{qual}(a_1)), (a_2, \alpha_{time}(a_2), \alpha_{qual}(a_2))\}, \text{ where}$$

$$\cdot a_1 = \textit{lift} \quad , \alpha_{time}(a_1) = 0.5, \alpha_{qual}(a_1) = 0.5,$$

$$\cdot a_2 = \textit{drive}, \alpha_{time}(a_2) = 0.4, \alpha_{qual}(a_2) = 0.7.$$

and for A_2 :

$$IR_{A_2} = \{(a_1, \alpha_{time}(a_1), \alpha_{qual}(a_1)), (a_2, \alpha_{time}(a_2), \alpha_{qual}(a_2))\}, \text{ where}$$

$$\cdot a_1 = \textit{lift} \quad , \alpha_{time}(a_1) = 0.3, \alpha_{qual}(a_1) = 0.4,$$

$$\cdot a_2 = \textit{drive}, \alpha_{time}(a_2) = 0.6, \alpha_{qual}(a_2) = 0.5.$$

For instance, the activity tuple act_1 of agent A_1 has two real internal factors which influence the performance of the activity *lift*, i.e. 0.5 is the internal factor for the performance in relation to time, and 0.4 is the internal factor for the performance in relation to quality. According to IR_{A_1} and IR_{A_2} , A_2 lifts an *obj_i* faster than A_1 (i.e. $0.5 > 0.3$), and the

² If agents have unlimited memory, they could maintain $2^q - 1$ models (excluding a model about “no agent”), where q is the number of agents. However, in most collaboration scenarios, agents have limited memory and collaborate without maintaining an exhaustive number of models.

outcome of A_1 's lift is of better quality than that of A_2 's lift (i.e. $0.5 > 0.4$), while CP stays constant.

However, IR_{A_1} and IR_{A_2} are not known or accessible to A_1 and A_2 . Therefore, the internal resources of A_1 and A_2 are estimated through models M . Set M_{A_1} for A_1 is as follows:

$$M_{A_1} = \{M_{A_1}(\widetilde{A}_1 = A_1), M_{A_1}(\widetilde{A}_2 = A_2), M_{A_1}(\widetilde{A}_3 = (A_1, A_2))\}, \text{ where}$$

- $M_{A_1}(A_1) = \{(lift, 0.3, 0.2), (drive, 0.3, 0.5)\},$
- $M_{A_1}(A_2) = \{(lift, 0.5, 0.7), (drive, 0.6, 0.1)\},$
- $M_{A_1}((A_1, A_2)) = \{(lift, 0.8, 0.9), (drive, \lambda, \lambda)\}.$

and the model set M_{A_2} for A_2 is as follows:

$$M_{A_2} = \{M_{A_2}(\widetilde{A}_1 = A_1), M_{A_2}(\widetilde{A}_2 = A_2), M_{A_2}(\widetilde{A}_3 = (A_1, A_2))\}, \text{ where}$$

- $M_{A_2}(A_2) = \{(lift, 0.2, 0.4), (drive, 0.1, 0.1)\},$
- $M_{A_2}(A_1) = \{(lift, 0.3, 0.2), (drive, 0.3, 0.4)\},$
- $M_{A_2}((A_1, A_2)) = \{(lift, 0.5, 0.6), (drive, \lambda, \lambda)\}.$

λ corresponds to a parameter of an activity which cannot be performed by two agents. Activity parameters for models with $|\widetilde{A}_j| > 1$ are determined by combining models of individual agents by using specific functions for different activities, e.g. $M_{A_2}(A_1, A_2)$ is a model where the parameters of the *lift* activity of A_1 and A_2 respectively are added.

An agent can only access its own models (it can not access the models maintained by other agents), e.g. M_{A_1} is accessible only by A_1 . These models allow each agent to estimate the potential contribution of individuals or subgroups of the team. The estimated time and quality for each activity is calculated by using the parameters $\widehat{\alpha}$ and the Context Parameter (CP). For instance, A_1 calculates the time that it would take A_2 to *lift* the *table* based on $\widehat{\alpha}_{time}(lift) \in M_{A_1}(A_2)$ and the weight and size of the *table*. In this example, we multiply these factors: $time(lift, \widehat{\alpha}_{time}, table) = \widehat{\alpha}_{time}(lift) * CP = 0.5seconds * 500 = 250seconds$.

Let us continue with the illustration of the algorithms in RA_{A_i} as described in Definition 5. The main algorithm (Figure 1) uses the following variables:

- PROPOSAL is a tuple that specifies an action a_i that shall be carried out by agent(s) \widetilde{A}_j to reach ms_i according to EC_T . In the table scenario, EC_T comprises time and quality, and proposals have the form (milestone ms_i , action a_i , agent(s) \widetilde{A}_j , time, quality),
- PROPOSAL_LIST is a list of proposals communicated by agents,
- UM is an updated model of the same form as a model in $M_{A_i}(\widetilde{A}_j)$.

In line 5 of Figure 1, the subroutine *generate_best_proposal*(m_i, M_{A_i}) provides a proposal which matches one or more agents with an activity, e.g. $(A_1, lift(table))$. A_i first calculates the time and quality of the performance for each activity by using the estimated internal parameters $\widehat{\alpha} \in M_{A_i}$ and CP. A_i orders possible proposals according to the functions $ec_i \in EC_T$ and then selects the best proposal.

Once an individual agent has calculated a proposal, it is communicated to other agents (line 6 in Figure 1). This is done by broadcasting it to every collaborator. The proposal from collaborators ($\forall A_j : j \neq i$) and the proposal made by A_i are stored in the PROPOSAL_LIST.

A joint decision is reached by choosing one of the proposals made by the agents. This is done by means of an election mechanism or policy P_A (line 8 in Figure 1). Agents

commit to a joint policy before they join a collaboration with other agents (similar to voters in a democracy). The concept of a policy P_A is discussed below.

Other routines not fully described in this paper deal with the task performance and the model updating process (which is also understood as a learning process). Briefly, the performance of the activities are in accordance with the entries in PROPOSAL_LIST. For instance, if A_i is in the PROPOSAL_LIST, it may be chosen to reach ms_i . The calculation of the time and quality of the real performance uses IR of collaborators (instead of using the models M for the proposals). The real performance is then compared to the proposed performance and is used as a basis for updating the existing models. Currently, we employ a simple updating process, but we are considering other learning algorithms for the future [Garrido *et al.* 1998].

2.1.4 Policy P_A

Each member of the group A can have a different proposal, because each agent has different models. However, agents should follow only one proposal at a time in order to reach coordinated behaviour. Hence, agents have to collectively decide on one proposal. A sensible way to solve this problem is to introduce a joint policy P_A (or election mechanism) which is provided to each agent that joins a collaborative scenario S . Formally, a joint policy P_A is denoted as:

Definition 6. *A joint policy P_A describes an election mechanism which incorporates the proposals made by several agents in a democratic fashion (by incorporating the opinion of each individual). The outcome of an election is a collective choice for one proposal.*

Like a human voter in a democratic election, an agent is expected to accept and commit to the outcome of this policy, which implies that an agent may relinquish the exertion of its own proposal in order to follow the proposal resulting from such an election.

In this paper, we consider an *optimistic policy* to determine a collectively agreed proposal, i.e. agents choose the most optimistic proposal promising the best time and

-
1. INPUT: environment E , task T_E , models M_{A_i} , policy P_A
 2. OUTPUT: updated models UM
 3. PROPOSAL, PROPOSAL_LIST, $UM \leftarrow \emptyset$
 4. **for** each $m_i \in MS$ **do**
 5. PROPOSAL \leftarrow *generate_best_proposal* (m_i, M_{A_i})
 6. *communicate_proposal* (PROPOSAL)
 7. PROPOSAL_LIST \leftarrow COMMUNICATED_PROPOSALS
 8. BEST_PROPOSAL \leftarrow *choose_best_proposal* (PROPOSAL_LIST, P_A)
 9. **end for**
 10. $UM \leftarrow$ *observe_real_performance*(PROPOSAL_LIST)
 11. return UM
-

Fig. 1. Pseudocode of find_best_proposal algorithm.

best quality of performance. For example, if agent A_1 proposes that the time to lift the *table* from *house_a* to the *car* takes A_2 100 seconds, and agent A_2 proposes that the time to perform this activity takes A_1 80 seconds, then the optimistic policy determines that A_2 should perform the activity. Due to the prior agreement to comply with the most optimistic proposal, agent A_2 will lift the *table*, and A_1 will stay idle. A contravention of this policy would result in uncoordinated collective behaviour.

In the future, we propose to investigate other forms of joint policies. For instance, policies based on *hierarchy* (i.e. each proposal is weighed according to a dominance structure), *candidate voting* (i.e. agents choose the agent that is preferred by most agents), or *experience* (i.e. each proposal is weighed according to the experience level of each agent). These mechanisms enable agents to determine an appropriate agent(s)-action match based on proposals of members of the group, and facilitates the coordination of agents while incurring limited overhead costs. However, it has yet to be shown which mechanisms are the most effective.

It is worth mentioning that an aggregation policy may be less important if agents negotiate with each other. Such a negotiation among agents may reveal a better agent for an activity. However, it generates additional communication among agents (increasing overhead costs). In addition, agents must possess capabilities to negotiate, which is a research field in its own right [Chu-Carroll & Carberry 2000, Li *et al.* 2003, Rahwan *et al.* to appear].

2.2 Overhead Costs

We consider two types of overhead costs: resource and transaction. *Resource costs* are the costs in regards to the available memory for each agent. *Transaction costs* are incurred by agents when communicating to arrive at a joint proposal. We consider the following transaction costs:

- inference costs are the costs for the agents to generate a proposal.
- communication costs are the costs for the agents to communicate proposals.
- aggregation costs correspond to the effort to form a consensus.
- update costs are the costs of updating models about each other.

One could interpret those costs in terms of computational resources (for inference, aggregation and update costs), storage resources (for memory costs), and network infrastructure and traffic (for communication costs).

In the empirical study performed in the next section, the cost of communicating is the only factor that contributes to the transaction (and overhead) costs.

3 Empirical Study

In order to evaluate agent collaboration, we implement a Surf Rescue (SR) scenario where agents are selfless, have a joint intention, and share resources to perform a rescue task together. We run several simulations with varying parameters where agents rescue a person in distress and assign a lifesaver based on models of the capabilities of lifesavers.

3.1 Surf Rescue Scenario

Imagine an SR environment where lifesavers rescue a person in distress. All lifesavers (agents $A = \{A_1, A_2, A_3, A_4, A_5\}$) are located on the shore (*shore*) and the Distressed

Person (DP) is located in the ocean (*ocean*). The task is to rescue the person in the shortest time possible. The problem is to assign the best lifesaver to perform this task based on models that each lifesaver has about the other lifesavers. The SR scenario is shown in Figure 2.

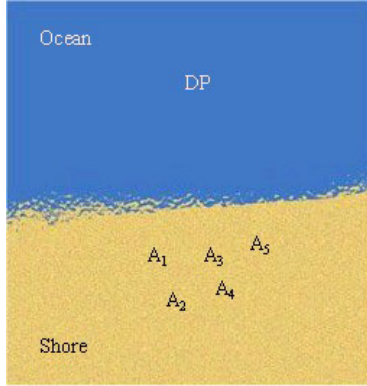


Fig. 2. A typical Surf Rescue scenario.

Each agent generates a proposal which specifies the agent that should swim and rescue the DP . The group decision procedure is based on an optimistic policy, which means that the proposed lifesaver with the most promising performance will be chosen.

EC_T consists of criteria for measuring the task performance of agents. In this scenario, EC_T consists only of minimizing the time required to complete a task (i.e. min_{time}). EC_T is given by

$$EC_T = \{min_{time}\}$$

The start state and goal state in this scenario is given by

$$MS_T = \{ms_0, ms_1\}, \text{ where}$$

- $ms_0 = (at(loc(A), beach)) \wedge (at(loc(DP), ocean))$,
- $ms_1 = (at(loc(DP), beach)) \wedge (at(loc(A), beach))$.

In other words, ms_0 represents the initial state, where the group of agents is located on the beach and the distressed person is in the ocean, and ms_1 represents the goal state, where the DP is brought back to the beach.

EC_T and MS_T are accessible to the agents which perform the task. Although all members in A independently possess the capabilities to perform an activity that will lead from ms_0 to ms_1 , the purpose of this collaboration is to find the best agent(s)-activity match based on each agent's models of other agents.

3.2 Methodology

The parameters α (of the internal resources IR) and $\hat{\alpha}$ (of the model sets M) are initialized with random real numbers between 0 and 1 for each lifesaver (IR is initialized once for the whole series of simulations, and M is initialized for each simulation). The

activity that is considered in the rescue scenario is to *swim and rescue*. Models are about individual lifesavers (i.e. $|\tilde{A}_j| = 1$) rather than several lifesavers together. We investigate several conditions of the collaboration scenario, which are described by the following parameters (expressed in percentages):

- Memory Boundedness (MB) determines the memory that is available to all agents, where 0% corresponds to no memory, and 100% to unlimited memory to accommodate all the models for the agents. For example, MB =80% means that on average 20% of the models are randomly discarded by each agent.
- Update Discrimination (UD) determines the number of agents that update (or learn) their models about other agents, where 0% corresponds to no agents updating their models, and 100% corresponds to all agents updating their models.
- Decision Discrimination (DD) determines the number of agents that make a proposal, where 0% indicates that no agents make a proposal, and 100% indicates that all agents make a proposal.
- M-IR Relationship (MIRR) determines the degree to which agents’ real capabilities are underestimated at the initialization of M, where 0% corresponds to a full underestimation of agents’ capabilities, and 100% corresponds to an estimation of agents’ capabilities which matches real capabilities³. MIRR provides an upper bound for model initialization, e.g. if MIRR is 50% the models are randomized between 0% and 50% at the real capabilities.

The agents which are affected by changing these parameters are chosen randomly.

Restrictions of an agent’s decision-making apparatus have been considered by Walker [1996] in collaborative task scenarios where the communicative choice of agents is influenced by inferential or attentional resource limitations. Rubinstein [1998] has investigated such restrictions on decision-making in a game-theoretic context. For example, he argues that agents have limited computational and memory resources, and therefore are not able to reason about all the alternatives in a decision problem.

We construct several collaboration scenarios from different settings of these parameters. We first define two benchmark scenarios:

- a “Lower Bound (LB)” scenario defines agents which do not maintain a model of their collaborators’ resources, and thus do not communicate proposals and do not update models.
- an “Upper Bound (UB)” scenario consists of agents which have an accurate model of each lifesavers’ capabilities in relation to EC_T , i.e. $IR_{A_i} = M_{A_i}(A_i)$, and an accurate $M_{A_i}(\tilde{A}_j)$.

In an LB scenario, the rescue is conducted by an agent which has been chosen randomly from the group. The UB scenario is consistent with the traditional assumption of MAS where agents have accurate knowledge about the internal resources of other agents prior to the collaboration, which results in an optimal assignment of agents to an activity. In the UB scenario, agents do not update their models or communicate proposals, because the same models are provided to all agents.

In a “Default” scenario, all parameters are set to 100%, i.e. all agents have enough memory to maintain all models (MB =100%), communicate proposals (DD =100%),

³ In the future, we will also consider overestimation of agents’ capabilities.

update their models (UD =100%), and initialize models according to the real capabilities (MIRR =100%). The main difference between the default scenario and the UB scenario is that models are initialized with random values as opposed to values that reflect accurate and complete knowledge of collaborators. Additionally, we define the scenarios $Scen(SimParam)$, where $SimParam \in \{MB, UD, DD, MIRR\}$ and it varies from 0% to 100%, while the other parameters stay at a constant 100%.

In order to investigate how the combination of simulation parameters affects task performance, we construct two additional scenarios from two pairs of simulation parameters: MB and DD, and UD and DD. For the pair MB and DD we consider the scenario $Scen(MB,DD)$ where the MB parameter varies from 0% to 100%, while the DD parameter varies from 0%,25%,50%,75% and 100%, and the scenario $Scen(DD,MB)$ where this relationship is reversed. The same scenario definition applies for the pair UD and DD resulting in two scenarios: $Scen(UD,DD)$ and $Scen(DD,UD)$.

To illustrate the effect of model accuracy on task performance, we average the result of a series of simulations for a particular $SimParam$ with different models, which are depicted in the figures below. We obtained the most representative results by running 1000 simulations for each value of $SimParam$ ⁴.

A better performance corresponds to a faster rescue, i.e. the measure of performance (depicted in the figures below) is an inverted function of time for a rescue. The transaction cost is measured by the total number of communicated proposals until convergence is reached. Initially, different lifesavers may be chosen for a rescue task due to the discrepancy between the models maintained by the agents and the internal resources. After each rescue, the agents update their models. The same rescue task is repeated until the agents choose the same lifesaver twice consecutively. The number of rescue tasks that is required to converge to one lifesaver determines the number of proposals that are communicated.

3.3 Results

Figure 3 plots task performance as a function of the different parameters in the scenarios, and Figure 4 plots the transaction costs against these parameters. Overall, the task performance is worse in simulations where agents have no models of agents in the group compared to simulations where agents maintain such models. The influence of model accuracy on the task performance differs in relation to each simulation parameter.

As expected, the results of the LB and UB scenarios correspond to the worst and best performance respectively, which are used as a benchmark for comparison with other scenarios. For instance, the agents in the Default scenario exhibit a constant task performance which is close to the performance of the UB scenario. We found that the task performance of agents appears to increase logarithmically as the available memory increases for each agent (MB scenario). In the UD scenario, the performance appears to increase polynomially as the agents' ability to update models increases. There is a quasi-linear increase in performance when more agents make proposals (DD scenario),

⁴ We varied the number of simulations until the data about performance and transaction costs showed stable and continuous patterns (Section 3.3).

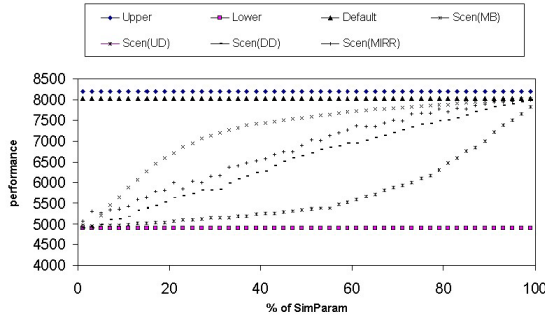


Fig. 3. Average task performance for 1000 simulations plotted against the simulation parameters from several scenarios.

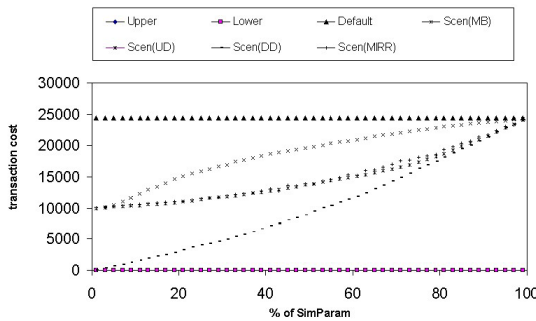


Fig. 4. Average transaction cost for 1000 simulations plotted against the simulation parameters from several scenarios.

which is similar to the results obtained when agents’ estimation of other agents become more accurate (MIRR scenario).

As indicated above, we consider only communication cost as the transaction cost for different scenarios (depicted in Figure 4). The transaction costs are the highest for the Default scenario. In the DD and the MIRR scenarios, the transaction costs increase proportionally to the increase of the simulation parameters. The transaction cost in the UD scenario increases slowly compared to the transaction costs in scenario MB. There is no transaction cost in the benchmark scenarios LB and UB, because no models are communicated (the chart lines of the LB and UB scenario overlap in Figure 4).

Figures 5 and 6 demonstrate the impact of the varying DD parameter on five different UD parameters and vice versa for *Scen(UD,DD)*. Figure 5 shows that restricting the updating of models to 0% and 25% cancels the effect of the increasing percentage of agents that make proposals. In the reverse scenario (Figure 6), the four values of the DD parameter between 25% and 100% have little impact on the performance until approximately 75% of agents are able to update their models.

Figures 7 and 8 demonstrate the impact of the varying DD parameter on five different MB parameters and vice versa for *Scen(MB,DD)*. Figure 7 shows that varying the memory of agents between 50%, 75% and 100% has little impact as the percent-

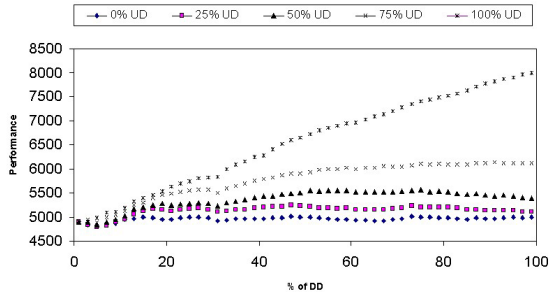


Fig. 5. The impact on the performance of the interaction between the increasing DD parameter and the five sets of UD parameters.

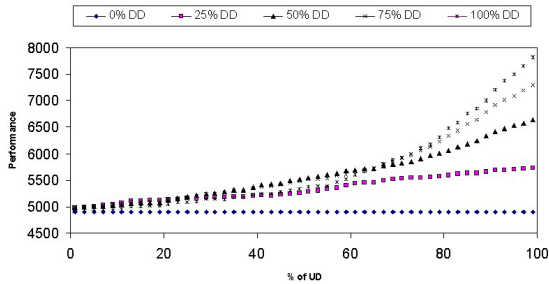


Fig. 6. The impact on the performance of the interaction between the increasing UD parameter and the five sets of DD parameters.

age of agents that are making proposals increases. In the reverse scenario (Figure 8), the performance for each value of DD increases proportionally to the percentage of the memory being used to store models.

4 Conclusion and Future Work

We have offered a notation for collaborative scenarios, which includes models of inaccurate and incomplete capabilities of collaborators (for each agent). This notation provided the basis for an empirical study. We demonstrated the effect of model accuracy on task performance of agents with varying operating parameters in regards to memory and decision-making processes. The findings of the empirical study support our thesis that agents improve their overall performance when employing these models, even under constrained scenario settings (e.g. limited memory and restricted decision-making). The evaluation of our study also included the cost of communication (in terms of the communicated proposals) resulting from the introduction of these models. Such costs may be considered in large-scale MAS, for example, when trying to avoid network congestion.

While our current framework offers a method to improve task performance, we are planning to extend this framework as follows. We propose to consider additional overhead costs in our empirical studies, such as memory costs, and inference and updating

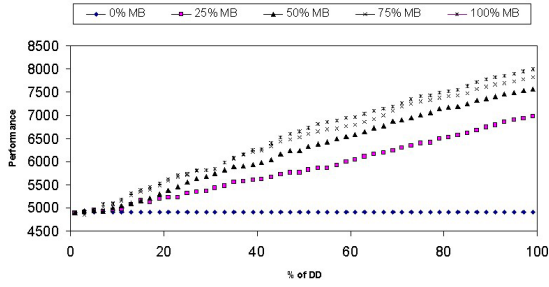


Fig. 7. The impact on the performance of the interaction between the increasing DD parameter and the five sets of MB parameters.

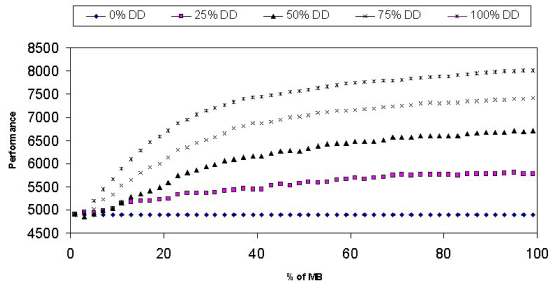


Fig. 8. The impact on the performance of the interaction between the increasing MB parameter and the five sets of DD parameters.

costs, in order to obtain more meaningful results. We also intend to consider different mechanisms for updating models from observed performance, and to evaluate the various aggregation policies mentioned in Section 2.1.4. Additionally, while we have assumed a domain with few uncertainties, more realistic scenarios exhibit probabilistic features, e.g. distorted observation of the real performance of other agents, unreliability of communication channels, or non-determinism of the outcome of activities. Probabilistic domain features may require an extension of the current framework.

Acknowledgements

This research was supported in part by Linkage Grant LP0347470 from the Australian Research Council, and by an endowment from Hewlett Packard. The authors would like to thank Michael Georgeff, Ian Peake, Yuval Marom, Bernard Burg, Iyad Rahwan, and Heinz Schmidt for helpful discussions. Additionally, we would like to thank the reviewers for valuable feedback.

References

ALTERMAN, RICHARD, & GARLAND, ANDREW. 2001. Convention in Joint Activity. *Cognitive Science*, 25(4), 611–657.

- CHU-CARROLL, JENNIFER, & CARBERRY, SANDRA. 2000. Conflict Resolution in Collaborative Planning Dialogues. *International Journal of Human Computer Studies*, **53**(6), 969–1015.
- COHEN, P. R., & LEVESQUE, H. J. 1991. *Teamwork*. Tech. rept. 504. Stanford University, Menlo Park, CA.
- DENZINGER, JOERG, & KORDT, MICHAEL. 2000. Evolutionary On-line Learning of Cooperative Behavior with Situation-Action-Pairs. In: *Proceedings of the International Conference of Multi-Agent Systems*.
- GARRIDO, LEONARDO, BRENA, R., & SYCARA, KATIA. 1998 (December). Towards Modeling Other Agents: A Simulation-Based Study. In: *Multi-Agent Systems and Agent-Based Simulation, LNAI Series*, vol. 1534.
- GYTRASIEWICZ, PIOTR J., & DURFEE, EDMUND H. 2001. Rational Communication in Multi-Agent Environments. *Autonomous Agents and Multi-Agent Systems*, **4**(3), A233–272.
- GROSZ, BARBARA J., & KRAUS, SARIT. 1996. Collaborative Plans for Complex Group Action. *Artificial Intelligence*, **86**(2), 269–357.
- KINNY, D., LJUNGBERG, M., RAO, A. S., SONENBERG, E., TIDHAR, G., & WERNER, E. 1992. Planned Team Activity. Pages 226–256 of: C. CASTELFRANCHI AND E. WERNER (ed), *Artificial Social Systems — Selected Papers from the Fourth European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW-92 (LNAI Volume 830)*. Springer-Verlag: Heidelberg, Germany.
- KOK, JELLE R., & VLASSIS, NIKOS. 2001 (August). *Mutual Modeling of Teammate Behavior*. Tech. rept. UVA-02-04. Computer Science Institute, University of Amsterdam, The Netherlands.
- LI, CUIHONG, GIAMPAPA, JOSEPH ANDREW, & SYCARA, KATIA. 2003 (November). *A Review of Research Literature on Bilateral Negotiations*. Tech. rept. CMU-RI-TR-03-41. Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.
- PANZARASA, P., JENNINGS, N., & NORMAN, T. 2002. Formalizing Collaborative Decision-Making and Practical Reasoning in Multi-Agent Systems. *Journal of Logic and Computation*, **12**(1), 55–117.
- RAHWAN, I., RAMCHURN, S. D., JENNINGS, N. R., MCBURNEY, P., PARSONS, S., & SONENBERG, L. to appear. Argumentation-Based Negotiation. *The Knowledge Engineering Review*.
- RUBINSTEIN, ARIEL. 1998. *Modeling bounded rationality*. Zeuthen lecture book series. Cambridge, Mass.: MIT Press.
- STONE, PETER, RILEY, PATRICK, & VELOSO, MANUELA M. 2000. Defining and Using Ideal Teammate and Opponent Agent Models. Pages 1040–1045 of: *In Proceedings of the Twelfth Annual Conference on Innovative Applications of Artificial Intelligence*.
- TAMBE, MILIND. 1997. Towards Flexible Teamwork. *Journal of Artificial Intelligence Research*, **7**, 83–124.
- WALKER, MARILYN A. 1996. The Effect of Resource Limits and Task Complexity on Collaborative Planning in Dialogue. *Artificial Intelligence Journal*, **1-2**(85), 181–243.