A Trust-based Decision-making Approach Applied to Agents in Collaborative Environments

Lucile Callebert, Domitile Lourdeaux and Jean-Paul Barthès
Sorbonne Universités, Université de Technologie de Compiègne, CNRS,
Heudiasyc UMR 7253, CS 60 319, 60 203 Compiègne, France

Keywords: Cognitive Agents, Trust-based Decision-making, Activity Description, Multi-agent Systems.

Abstract: In Virtual Environments for Training, the agents playing the trainee's teammates must display human-like behaviors. We propose in this paper a preliminary approach to a new trust-based decision-making system that allow agents to reason on collective activities. The agents' integrity, benevolence and abilities dimensions and their trust beliefs in their teammates' integrity, benevolence and abilities allow them to reason on the importance the give to their goals and then to select the task that best serves their goals.

1 INTRODUCTION

In Collaborative Virtual Environments for Training (CVET), virtual characters and the trainee have to work together on a collective activity to achieve team goals. Examples of such CVET include the *SecuReVi* application for firefighters training (Querrec et al., 2003), or the 3D Virtual Operating Room for medical staff training (Sanselone et al., 2014). In those CVET, agents must display a human-like behavior and people must monitor their teammate activities to make the best decisions.

To train people to work in teams, we must confront them with all types of teammates: 'good teammates' (i.e. working hard to achieve the team goals and doing their best to help other team members, etc.), or on the opposite, 'bad teammates' (i.e. favoring to their own goals, not helping team members, etc.) Team members will then have to address questions like: *Is my teammate able to do that or should I do it myself?*, *Is my coworker committed enough to the team to help?* or *Would my teammate be kind enough with me to help me?*. Such questions rely on the concept of trust (i.e. *Do I trust my coworker's commitment to the team?* , *Do I trust my teammate's benevolence toward me?*, *Do I trust my teammate's capacities?*).

We propose in this paper a preliminary approach to a trust-based decision-making mechanism enabling agents in CVET to reason on collective activities. We first introduce some major works on activity description models and on trust models in Section 2. We present in Section 3 a general overview of our system. In Section 4 we present the activity model used to describe the team activity, and in Section 5 we present the activity instances on which agents reason. In Section 6 we introduce the agent model. We detail in Section 7 the decision-making mechanisms a=that we then illustrate by an example in Section 8, before concluding in Section 9. The coupling with a virtual environment and the integration of the trainee will not be discussed in this paper.

2 RELATED WORK

2.1 Activity Models in CVET

In the *SecuriVi* project (Chevaillier et al., 2012) and the 3D Virtual Operating Room project (Sanselone et al., 2014) the activities that the agent and the trainee can perform are described respectively thanks to the HAVE activity meta-model and Business Process Model Notation diagrams. In both projects, agents are assigned to a role and have specific tasks to do for which they must be synchronized. Yet we want the trainee to adapt her behavior to her teammates on activities where no roles are pre-attributed to team members. The scenario language LORA (Language for Object-Relation Application) (Gerbaud et al., 2007) supports the description of collective activity: roles can be associated with actions and domain experts can specify collective actions that several agents have to

Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

do simultaneously. However in LORA only the prescribed procedure is represented, which does not allow agents to deviate from this procedure. The activity meta-model ACTIVITY-DL (Barot et al., 2013) is used both for monitoring the trainee's behavior and for generating virtual-characters' behavior. Task description in ACTIVITY-DL corresponds to a cognitive representation of the task, which is well suited to our human-like behavior-generation objective: agents will be able to reason on the task description in a way that imitates human cognitive processes. ACTIVITY-DL supports the description of procedure deviations: it is representative of the activity observed in the field.

2.2 Dyadic Trust Models

A dyadic trust model will allow agents to take into account individual characteristics of their teammates to make a decision about the collective activity. In the following, we call 'trustor' the subject of the trust-relationship (i.e. the person who trusts). The trustee is the object of the trust-relationship (i.e. the person who is trusted).

(Mayer et al., 1995) propose a model of organizational trust including factors often cited in the trust literature. They identify three dimensions to trust: integrity, benevolence and ability. One trusts someone else's integrity if one believes the other will stick to its word and fulfill her promises. The trustor trusts the trustee's benevolence toward her if she ascribes good intentions to the trustee toward her. The trustor's trust in the trustee's abilities depends on how she evaluates the trustee's capacity to deal with an identified task.

(Marsh and Briggs, 2009) propose a computational model of trust for agent collaboration. To decide over cooperation, the trustor compares the situational trust, which represents how much she trusts the trustee in an identified situation with a cooperation threshold, which represents how much she needs to trust the trustee in this situation to cooperate with her. In this model, there is no notion of team, yet we believe that the team concept itself plays a role in the team member behavior.

(Castelfranchi and Falcone, 2010) propose a computational model of social trust. In this model the trustor's beliefs about the trustee's motivation, capacity and opportunity to do a task, are used to decide over delegation. In this model the motivation belief is very difficult to compute without a context: for example it is deduced from the trustee's profession (e.g. a doctor is believed to be motivated to help her patients), or from the trustee's relationship with the trustor (e.g. friends are supposed to be willing to help one another). Without this context, motivation is difficult to explain, yet agent behavior needs to be explainable to the trainee. The integrity and benevolence dimensions in the model of (Mayer et al., 1995) provide such an explanation.

The model of organizational trust of (Mayer et al., 1995) is the most appropriate in our context: it takes into account a dimension specific to the team, and the benevolence relationships provide a basis for helping behaviors without having to add specific models of interpersonal relationships.

3 GENERAL FUNCTIONING

The general functioning of the trust-based task-selection system for agent x in the team of agents $\mathcal{A} = \{x, y, z\}$ is presented in Figure 1. We propose augmenting the activity meta-model ACTIVITY-DL to support collective-activity description. The collective activity model is described by ergonomists as a tree of tasks, in which leaf tasks correspond to actions. At the beginning of the simulation, the activity-treatment module uses the activity model to generate an activity instance representative of agents' progress on the activity and on which agents will reason during the whole simulation. This corresponds to Step ① of Figure 1. Agent x reasons on the activity instances that correspond to its goals for selecting a task that corresponds to an action. Its personal integrity, benevolence and ability dimensions influence its choices as well as the other agents that it takes into account thanks to its trust beliefs about them. This corresponds to Step ② of Figure 1. In Step ③ of Figure 1 x's action is treated by the virtual-environment module, which then informs the activity-treatment module in Step ④ of Figure 1. Those two stages are not further discussed in this paper since not directly related to the agents decision-making system for collective activities. Finally the activity-treatment module updates the activity instances in Step ⑤ of Figure 1.

4 ACTIVITY MODEL

We first briefly present ACTIVITY-DL before proposing our augmentation of ACTIVITY-DL for collective activities.

4.1 ACTIVITY-DL

ACTIVITY-DL is a meta-model for describing human activities inspired from studies in ergonomics: the activity is represented as a set of tasks hierarchically decomposed into subtasks in a way that reflects human

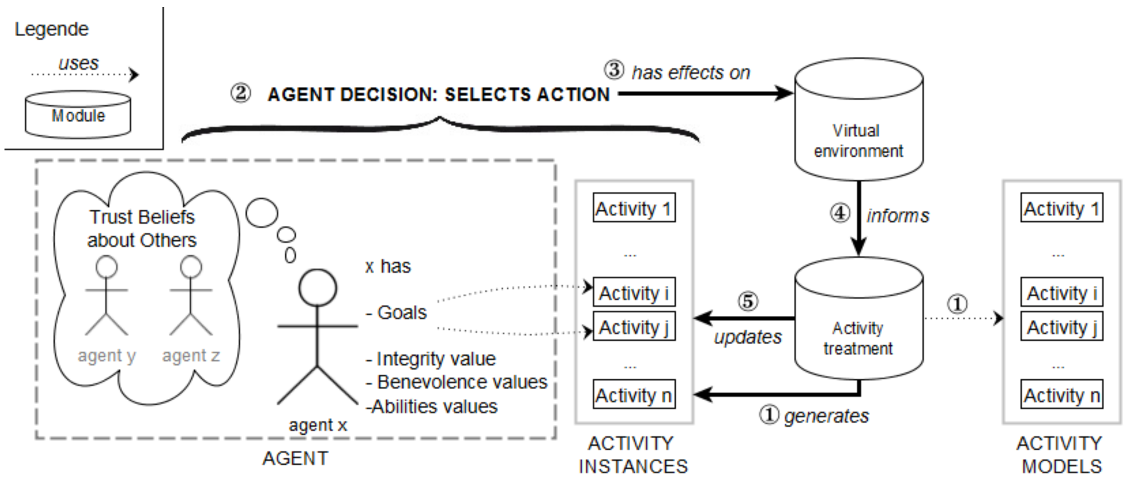

Figure 1: General functioning of the agent decision-making system and its effects on the simulation.

cognitive representations. Task hierarchical decomposition takes the form of tree of tasks, with leaf tasks τ_l representing concrete actions to be executed in the virtual environment. Non-leaf tasks τ_k are abstract tasks and are decomposed into subtasks: let T_k be the set of τ_k's subtasks. Those substasks are logically organized thanks to satisfaction conditions and temporally organized through ordering constraints. For more details, see (Barot et al., 2013).

We give an example of activity description in ACTIVITY-DL that we will use to illustrate agent reasoning. If we consider the utterance: 'For the lab to be set up, the floors have to be cleaned and the desk has to be assembled. To clean the floors, vacuuming should be done first and then mopping. Assembling the desk and cleaning the floors can be done at the same time.' This example can be represented in ACTIVITY-DL through the tree of tasks:

- τ_{111} = *vacuum floors* (corresponds to an action).
- τ_{112} = *mop floors*. (corresponds to an action).
- τ_{11} = *clean floors*. τ_{111} and τ_{112} are τ_{11}'s subtasks: $T_{11} = \{\tau_{111}, \tau_{112}\}$. τ_{111} and τ_{112} have to both be done sequentially so a SEQ-ORD ordering-constraint and an AND satisfaction condition are attached to τ_{11}.
- τ_{12} = *assemble desk*. (corresponds to an action).
- τ_1 = *set up lab*. τ_{11} and τ_{12} are τ_1's subtasks: $T_1 = \{\tau_{11}, \tau_{12}\}$. A PAR ordering-constraint and an AND satisfaction condition are attached to τ_1 since τ_{11} and τ_{12} can be done at the same time.

We provide two graphical representations of a task tree described in ACTIVITY-DL: as a tree in Figure 2(a), and as a work flow in Figure 2(b). In this latter representation, dark-gray bars indicate the beginning of the task, and light-gray bars indicate the end. Rectangular boxes represent concrete actions.

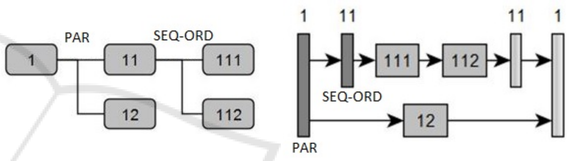

Figure 2: Set-up-lab example of an ACTIVITY-DL tasks-tree represented as a task tree on the left(a) and a work-flow on the right (b).

4.2 Augmenting ACTIVITY-DL to Support Collective-activity

Until now, ACTIVITY-DL was used for describing individual activity. In order to describe collective activities, we added some requirements that have to be specified by ergonomists only on leaf tasks τ_l (i.e. tasks corresponding to actions). Such requirements are the followings:

Number-of-agent Requirement: We consider collective activities with collective actions on which several agents can or have to do at the same time. For a leaf task τ_l representing such a collective action, it is necessary to specify $n_{\min}(\tau_l)$ a minimum of persons that are needed to do τ_l and $n_{\max}(\tau_l)$ a maximum of persons that can work together on τ_l. For example ergonomists can specify that two to four persons can work on τ_{12} = *assemble desk*: $n_{\min}(\tau_{12}) = 2$ and $n_{\max}(\tau_{12}) = 4$.

Skill Requirement: An action may require some particular skills that are attached to the action. We define $\Sigma = \{\sigma_1, \sigma_2, ...\}$ the set of skills and $\Sigma_l \subseteq \Sigma$ the set of skills attached to τ_l. For example ergonomists can specify that skills σ_1 = *know how* *to use a screwdriver* and σ_2 = *know haw to read instructions* are necessary to do τ_{12}.

5 ACTIVITY INSTANCE

To enable agents to reason on the abstract tasks of the activity tree, the leaf task requirements have to be propagated to all abstract tasks, which is done by the activity-treatment module when generating an activity instance (i.e. Step ① of Figure 1). An activity instance is different from an activity model in that it directly supports agent reasoning thanks to abstract task constraints that are representative of the agent progress in the activity tree. Also because those constraints are representative of the agent progress in the activity, the activity instances are updated by the activity-treatment module each time progress is made (i.e. Step ⑤ of Figure 1), namely each time agents do an action.

For an abstract task τ_k, two types of conditions are generated by the activity-treatment:

The Feasibility Condition is static and must be verified so that τ_k can be done. This condition relies on the satisfaction condition attached to τ_k: there must be enough agents and all together agents must have all the required abilities to achieve either all of τ_k's subtasks (AND satisfaction-condition) or one of τ_k's subtasks (OR satisfaction-condition).

The Progress-representative Condition is dynamic and is representative of agents' process in the collective activity, thus this condition is updated each time agents do an action. This condition relies on the ordering constraint attached to τ_k: if a PAR ordering constraint is attached to τ_k at time t their must be enough agents to do *one of* τ_k's subtasks *that is not done yet*. If a SEQ-ORD ordering constraint is attached to τ_k, the progress-representative condition is representative of τ_k's next subtask to be achieved.

We do not further detail the constraint propagation rules in this paper due to space limitations. Considering the activity model described in Section 4, at the activity instance generation, the constraint propagation gives:

- for τ_{11} that has a SEQ-ORD ordering constraint and an AND satisfaction condition attached:
  - The feasibility condition expresses that one agent is enough to realize both τ_{111} and τ_{112}, and no skills are required.
  - The progress-representative condition is initially representative of τ_{111}: one agent is enough and no skill is required.
- for τ_1 that has a PAR ordering constraint and an AND satisfaction-condition attached:
  - The feasibility condition expresses that two agents are necessary to execute τ_1 since τ_{12} requires two persons, and those agents must have the skills σ_1 and σ_2 required by τ_{12}.
  - The progress-representative condition is initially representative of both τ_{11} and τ_{12}. The minimum of agents is then one, since only one agent is required for τ_{11}. Agents with no skill can participate to τ_1 by doing τ_{11}.

6 AGENT MODEL

We propose an agent model that will allow agents to reason on their goals and on the activity instances that correspond to their goals while taking others into account. We first present the goals and dimensions of our agents and then the beliefs agents have about other agents.

6.1 Agent Goals and Dimensions

Agents have goals and personal dimensions based the model of organizational trust proposed by (Mayer et al., 1995) that we formally define in the following paragraphs.

Agent Goals. We make a distinction between agent personal goals and goals that the agent shares with the rest of the team. We define $\gamma_{x,self}$, the personal goal of agent x, and $\gamma_{x,team}$ the goal that x shares with the team. Each goal corresponds to a task tree which describes the tasks that should be done to achieve the goal.

Agent Dimensions. For agent x in the team of agents $\mathcal{A} = \{x, y_1, y_2, ...\}$, we describe x's personal dimensions as follows:

- Integrity. $i_x \in]0,1[$ is x's integrity value toward the team.
- Benevolence. As the benevolence is directed toward other agents, x has a set of benevolence values $\{b_{x,y_1}, b_{x,y_2}, ...\}$. For all agents $y_i \in \mathcal{A}, y_i \neq x$, x's benevolence value toward y_i is such that $b_{x,y_i} \in]0,1[$.

- Ability. Similarly to x's benevolence, x's abilities are related to specific skills: for all skills $\sigma_j \in \Sigma$, x has an ability value $a_{x,j} \in [0,1[$. Thus x's set of abilities is $\{a_{x,1}, a_{x,2}, ...\}$. A high ability value on skill σ_j indicates that x tends to master the skill, as a low ability value indicates that x is not very competent on that skill.

The integrity, benevolence and ability values define x's state of mind, and will influence x's decision.

6.2 Agent Beliefs

When making a decision, people tend to imagine how others would react to their choice. This theory-of-mind capability (Carruthers and Smith, 1996) allows people to take others into account in their decision-making. To reason on what others would like, agents use the beliefs they have about others' goals and others' state of mind.

Others' Goals. Agents have beliefs about their teammates' goals. For all agents $x, y_i \in \mathcal{A}, x \neq y_i$, we define $\gamma^x_{y_i,self}$ that is what x thinks is y_i's personal goal, and $\gamma^x_{y_i,team}$ that is what x believes is y_i's goal shared with the team.

Others' Personal Dimensions. Following the model of organizational trust if (Mayer et al., 1995), we define the trust beliefs of $x \in \mathcal{A}$ about agent $y_i \in \mathcal{A}, x \neq y_i$:

- Integrity trust-belief. $i^x_{y_i} \in]0,1[$ is what x thinks of y_i's integrity.
- Benevolence trust-belief. x's trust in y_i's benevolence is $\{b^x_{y_i,x}, b^x_{y_i,y_1}, ...\}$, where $b^x_{y_i,y_j} \in]0,1[$. For agent $y_j \neq y_i$, $b^x_{y_i,y_j}$ is what x thinks of y_i's benevolence toward y_j.
- Ability trust-belief. $\left\{a^x_{y_i,1}, a^x_{y_i,2}, ...\right\}$ is what x believes of y_i's abilities, with $a^x_{y_i,j} \in [0,1[$ what x thinks of y_i's ability on skill $\sigma_j \in \Sigma$.

7 DECISION-MAKING PROCESS

Agents' decision-making process allow them to reason on their goals to decide which one is the most important, and then to reason on the collective activity instances to select a task that serves their preferred goal. Both when reasoning on goal importance and when selecting a task, agents take others into account thanks to their trust beliefs. We develop in the following sections processes of goal importance computation and task selection.

7.1 Goal Importance

Agent x has to compute the importance it gives to its personal goal and to its team goal in order to decide which one to favor. In order to do so, x computes the initial importance value it gives to its goals and then takes others into account to compute the final importance value of its goals.

Initial Goal Importance. Let $iImp_x(\gamma) \in [0,1[$ be the initial importance value of the goal γ for agent x. Since x's integrity represents x's tendency to fulfill its promises, x's initial importance value for the team goal corresponds to x's integrity value: $iImp_x(\gamma_{x,team}) = i_x$, while x's initial importance value for its personal goal is $iImp_x(\gamma_{x,self}) = 1 - i_x$.

x then uses its theory-of-mind capability to compute how much much importance it thinks its teammates initially give to their goals. We define $iImp^x_{y_i}(\gamma)$ what x thinks of the initial importance of γ for agent y_i. For all agents $y_i \in \mathcal{A}, y_i \neq x$, x uses its theory-of-mind capability and reasons on $i^x_{y_i}$ to compute $iImp^x_{y_i}(\gamma_{y_i,team})$ and $iImp^x_{y_i}(\gamma_{y_i,self})$.

Final Goal Importance. Let $fImp_x(\gamma) \in [0,1[$ be the final importance value of the goal γ for agent x, which designates the importance x gives to γ *after* taking into account its teammates.

x first computes with Equation 1 its desirability $d_{x,y_i}(\gamma) \in]-1,1[$ to realize the goal γ *for* agent y_i. $d_{x,y_i}(\gamma)$ is proportional to both $iImp^x_{y_i}(\gamma)$ and to x's benevolence toward y_i. The formula used in Equation 1 allows $d_{x,y_i}(\gamma)$ to be positive if x is benevolent toward y_i (i.e. wants to help) and negative if x is not benevolent toward y_i (i.e. does not want to help).

$$d_{x,y_i}(\gamma) = iImp^x_{y_i}(\gamma) \times 2(b_{x,y_i} - 0.5) \tag{1}$$

Then in Equation 2 if x also has γ as a goal, x computes the mean of its own initial importance value for γ and the desirability values $d_{x,y_i}(\gamma)$ for all of the n agents y_i that also have γ as a goal. Since this value might be negative, the final importance of γ for x is the maximum between this mean and zero.

$$fImp_x(\gamma) = max\left(\frac{iImp_x(\gamma) + \sum_{i=1}^{n} d_{x,y_i}(\gamma)}{\mathrm{n}+1}, 0\right) \tag{2}$$

A very process (except that no initial importance value is taken into account) is used if x does not initially have γ as a goal. This process is defined in Equation 3.

$$fImp_x(\gamma) = max\left(\frac{\sum_{i=1}^{n} d_{x,y_i}(\gamma)}{\mathrm{n}}, 0\right) \tag{3}$$

7.2 Task Selection

Based on how much importance agent x gives to its goals and how much importance it thinks its teammates give to their goals, x has to choose the task it wants to do. In order to do so, x recursively reasons on task trees and computes task utilities. Finally x generates task distributions among agents to choose the one that best serves its interests.

7.2.1 Recursive Process

We explain in the following paragraphs the functioning of the recursive task-selection process for agent x. We provide in Figure 3 a diagram that represents the set of agents on which x reasons at steps n and $n+1$ of the recursive process, and Figure 3 illustrates those sets.

Set of Tasks on which x Reasons. We define $\mathrm{T}_{x,n} \subset \mathrm{T}$ the set of tasks on which x reasons during Step n of its recursive process of task selection. If $\mathrm{T}r \subset \mathrm{T}$ is the set of the task-tree roots, then x reasons at the first step of the recursive process on $\mathrm{T}_{x,1} = \mathrm{T}r$. If τ_k is the task that x selects at Step n of the recursive process, then at Step $n+1$, x reasons on $\mathrm{T}_{x,n+1} = \mathrm{T}_k$, where T_k is the set of τ_k's subtasks. Of course, x *chooses* one of τ_k's subtasks only if τ_k has a PAR ordering-constraint attached. If a SEQ-ORD ordering-onstraint is attached to τ_k x has to do the first subtask of τ_k that is not done yet.

Set of Agents on which x Reasons. When making a choice, people tend to anticipate how their choice would impact others. They use their theory-of-mind capability to make a decision that takes others into account. At the first step of the recursive task-selection process, x takes into account all its teammates. But then in the following steps of the recursive process, x only takes into account the relevant agents, namely those that x thinks will choose the same task τ_k as itself in the previous step of its recursive decision-making process. Similarly to $\mathrm{T}_{x,n}$, we define $\mathcal{A}_{x,n} \subseteq \mathcal{A}$ the set of agents on which x reasons during Step n of the recursive task-selection process. At the first step of the recursion, $\mathcal{A}_{x,1} = \mathcal{A}$. If τ_k is the task that x selects at Step n, the agents $y_i \in \mathcal{A}_{x,n}$ that x thinks will also choose τ_k compose the set of agents $\mathcal{A}_{x,n+1} \subseteq \mathcal{A}_{x,n}$ on which x reasons at Step $n+1$.

7.2.2 Task Utility

Let o_k^+ be a state of the world that corresponds to a success outcome for the task τ_k and let $U_x(o_k^+) \in [0,1[$

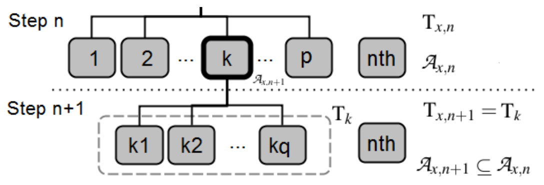

Figure 3: Step n and Step $n+1$ of the recursive task-selection process of agent x.

be the utility for agent x to achieve o_k^+. At the step n of the recursive task-selection process, x computes $U_x(o_k^+)$ for every task τ_k such that $\tau_k \in \mathrm{T}_{x,n}$. To compute $U_x(o_k^+)$, x has to consider if it can, if it wants, and if it should do the task.

Can x do the Task? x first checks that the task is doable by the team of agents by reasoning on the feasibility condition. Then x checks if it can participate to τ_k by reasoning on the progress-representative condition. When doing so, x reasons on its abilities and on what it thinks of the abilities of its teammates.

Does x Want to do the Task? By reasoning on the task trees, x checks if τ_k contributes to any of its goals, according to the contributes relationship defined by (Lochbaum et al., 1990). If it is not the case, then x has no interest to do τ_k and $U_x(o_k^+)$ is set to 0. Otherwise x considers how much it is skilled to do τ_k.

Should x do the Task? x's abilities influence x's choice to do the task: x has to compute $a_{x,\tau}$ a general ability value on the task. If the task is a leaf task τ_l, then x takes into account its ability values on the skills that are attached to τ_l, if any. If the task is an abstract task τ_k, x reasons on the progress-representative condition attached to τ_k to compute a_{x,τ_k}. Due to space limitations, we do not further develop the calculation process of a_{x,τ_k}.

Task Utility Value. Finally, if τ_k contributes to one of x's goals, x computes $U_x(o_k^+)$ with Equation 4, which formula allows $U_x(o_k^+)$ to stay in the interval $[0,1[$. $U_x(o_k^+)$ is proportional to $fImp_x(\gamma)$ and if x is skilled on the skills attached to τ_k, $U_x(o_k^+)$ is increased proportionally to a_{x,τ_k}.

$$U_x(o_k^+) = fImp_x(\gamma) \times (1 + a_{x,\tau_k}(1 - fImp_x(\gamma))) \quad (4)$$

Task Utility for Others. x also computes the utilities it thinks its teammates have for the task outcomes. We define $U_{y_i}^x(o_k^+)$ what x thinks of the utility for y_i to achieve o_k^+. At Step n of the recursive task-selection process, x has to compute $U_{y_i}^x(o_k^+)$ for all agents y_i such that $y_i \in \mathcal{A}_{x,n}$, $y_i \neq x$ and for all tasks τ_k such that

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

$\tau_k \in \mathrm{T}_{x,n}$. To compute $U^x_{y_i}(o^+_k)$, x uses the same process than for itself but reasons on $a^x_{y_i,j}$ its ability trust-beliefs about y_i what it thinks of y_i's goals $\gamma^x_{y_i,team}$ and $\gamma^x_{y_i,self}$.

7.2.3 Task Distribution

Once utility values are computed, x generates task distributions among agents and computes their utilities to select the task that best serves its interests.

Task Distribution Generation. At the step n of the recursive task-selection process, x generates task distributions with all tasks τ_k such that $\tau_k \in \mathrm{T}_{x,n}$ and with all agents y_i such that $y_i \in \mathcal{A}_{y_i,n}$. At this point x takes into account the number-of-agent constraints that were propagate through the progress-representative condition on the tasks: if a task τ_k necessitates a minimal number of agents $\mathrm{n_{min}}(\tau_k)$ and a maximal number of agents $\mathrm{n_{max}}(\tau_k)$, then x only generates arrangements where $\mathrm{m}(\tau_k)$ agents are assigned to τ_k such that $\mathrm{m}(\tau_k) = 0$ (i.e. in this case, τ_k is not executed) or $\mathrm{n_{min}}(\tau_k) \leq \mathrm{m}(\tau_k) \leq \mathrm{n_{max}}(\tau_k)$.

Task Distribution Utility. Agent x then selects the action that corresponds to what it thinks is the best task distribution. In order to do so, x computes the utility of a task distribution as the average of all utilities that x thinks agents have for the outcomes of the tasks they are assigned to in the distribution.

Task Selection. x finally selects the task τ_k for which the utility of the task distribution is maximized. If this task is a leaf-task, then the recursive task-selection process ends and x does the action corresponding to τ_k. Otherwise x continues the recursive task-selection process, and as explained in Section 7.2.1, at step $n+1$, $\mathrm{T}_{x,n+1} = \mathrm{T}_k$ and $\mathcal{A}_{x,n+1}$ is the set of agents that were assigned to τ_k in the task distribution (i.e. x and maybe others). We underline that x does not make a decision for its teammates: x selects a task that it thinks corresponds to the best task distribution. When doing so, x does not distribute tasks to other agents. When making a decision, other agents may or may not choose the task x projected they would choose.

8 EXAMPLE

We develop in this section a small example of the functioning of our system. Let $\mathcal{A} = \{xenia, yuyu, zoe\}$ a team of agents who have one team goal $\gamma_1 =$ *lab set up*. *xenia* has a personal goal, $\gamma_2 =$ *paper written*. In formulas we designate the agents by their initial letter (e.g. $b_{x,y}$ is *xenia*'s benevolence toward *yuyu*). In this scenario *yuyu* and *zoe* are rather upright: $i_y = i_z = 0.75$, unlike *xenia*: $i_x = 0.25$. They all are rather highly benevolent toward their teammates (i.e. all benevolence values are 0.75). For simplicity reasons, we consider that agents have the true models of their teammates.

Goal Importance Computation. We develop *yuyu*'s process of goal-importance-value computation:

- She computes her initial importance value for γ_1 as described in Section 7.1. She obtains $iImp_y(\gamma_1) = 0.75$.
- She uses her theory-of-mind capability and computes $iImp^y_x(\gamma_1) = 0.25$, $iImp^y_x(\gamma_2) = 0.75$ and $iImp^y_z(\gamma_1) = 0.75$: she thinks that *xenia* will favor her personal goal and that *zoe* gives a high importance to the team goal.
- Then she uses Equation 1 and computes $d_{y,x}(\gamma_1) = 0.125$, $d_{y,x}(\gamma_2) = 0.375$ and $d_{y,z}(\gamma_1) = 0.375$: she rather likes *xenia* and *zoe* and so she wants them to have the goals they value achieved.
- Using Equation 2 and Equation 3 she finally obtains $fImp_x(\gamma_1) \simeq 0.417$ and $fImp_x(\gamma_2) = 0.375$: she is benevolent toward *xenia* so she adopts her goal, but it is still more important to her to achieve the team goal.

Influence of Integrity and Benevolence. Agent integrity plays a crucial role in computing the goal-importance value computation: although her teammates give importance to the team goal and even taking into account their teammates preferences, her personal goal is more important to *xenia*: from *xenia*'s point of view, we obtain: $fImp_x(\gamma_1) \simeq 0.333$ and $fImp_x(\gamma_2) = 0.75$. Agent benevolence also plays a crucial role: if in the same example, *yuyu* is not benevolent toward *xenia* (i.e. $b_{y,x} < 0.5$) then she will have a negative desirability to see *xenia*'s goal achieved. Hence *yuyu* would compute $fImp_y(\gamma_2) = 0$ and would not help *xenia*.

Task Selection. We consider that all agents have the same ability value on σ_1 ($\forall i \in \mathcal{A}, a_{i,1} = 0.5$), but different abilities values on σ_2 ($a_{x,2} = 0.75, a_{y,2} = 0.25$ and $a_{z,2} = 0.25$). *yuyu* will reason here on the *set up lab* activity instance described in Section 4. We consider the task τ_2 as the root of the activity-instance tree *write paper*.

As explained in Section 7.2.1, at the first step of her recursive decision-making process for task selection, *yuyu* reasons on the set of agents $\mathcal{A}_{y,1} = \{xenia, yuyu, zoe\}$ and on the set of tasks $T_{y,1} = \{\tau_1, \tau_2\}$. *yuyu* computes the success-outcome utilities of the tasks in $T_{y,1}$. She goes through the task utility computation process to compute $U_y(o_1^+)$ as described in Section 7.2.2: she can do τ_1 since she has the required abilities. She wants to do τ_1 since τ_1 contributes to her goal γ_1. She computes her general ability value for τ_1 and obtains $a_{y,\tau_1} = 0.5$. She finally computes $U_y(o_1^+)$ as described in Equation 4 and obtains $U_y(o_1^+) \simeq 0.54$. She applies the same process with o_2^+ and obtains $U_y(o_2^+) = fImp_y(\gamma_2) = 0.375$. She then uses her theory-of-mind capability to compute what she thinks of *xenia*'s utilities, applying the same process than for herself. She obtains $U_x^y(o_1^+) = 0.375$ and $U_x^y(o_2^+) = 0.75$. She does the same for *zoe* and obtains $U_x^z(o_1^+) \simeq 0.82$ and $U_x^z(o_2^+) = 0$.

yuyu then generates task distributions as explained in Section 7.2.3, which we do not list here, but it is obvious here that the maximal task-distribution utility is obtained when *yuyu* and *zoe* are assigned to τ_1 and *xenia* is assigned to τ_2. Hence *yuyu* selects the task τ_1. Because τ_1 is an abstract task, she recursively starts again the task-selection process as described in Section 7.2.1 to choose one of τ_1's subtasks: it is the step 2 of her recursive task-selection process. At this step, she reasons on the set of tasks $T_{y,2} = T_1 = \{\tau_{11}, \tau_{12}\}$ and on the set of agents $\mathcal{A}_{y,2} = \{yuyu, zoe\}$ since she thinks *zoe* will also choose τ_1. This second step of recursive decision-making is similar to the first one and we will not develop it here. At the end of this step, *yuyu* chooses the task τ_{12} (because both τ_{11} and τ_{12} contribute to her goal γ_1, she is skilled on τ_{12} and she thinks *zoe* will also choose $\tau_1 2$). This task is a leaf task that corresponds to an action, hence the recursion stops here, and *yuyu* will try execute the action that corresponds to τ_{12}.

9 CONCLUSIONS

We proposed in this paper mechanisms of decision-making for generating agent behavior in collective activities. We proposed an augmentation on ACTIVITY-DL that supports collective activity description. We defined activity instances that are representative of agents progress on the collective activity and on which agents can directly reason to select their actions. We proposed an agent model based on the trust model of (Mayer et al., 1995) and a trust-based decision-making system that allow agents to reason on activity instances and to take their teammates into account to select an action. We gave an example of the functioning of the activity-treatment module and of the decision-making system. Further work perspectives include testing the decision-making system when agents have false beliefs about others, and evaluating the credibility of the produced behaviors. The model could also be extended so that agents could to act purposely to harm the team or their teammates, which is not currently possible since agents can only decide *not to* help the team.

ACKNOWLEDGEMENTS

This work was carried out in the framework of the VICTEAMS project (ANR-14-CE24-0027, funded by the National Agency for Research) and funded by both the Direction Générale de l'Armement (DGA) and the Labex MS2T, which is supported by the French Government, through the program "Investments for the future" managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02).

REFERENCES

Barot, C., Lourdeaux, D., Burkhardt, J.-M., Amokrane, K., and Lenne, D. (2013). V3s: A Virtual Environment for Risk-Management Training Based on Human-Activity Models. *Presence: Teleoperators and Virtual Environments*, 22(1):1–19.

Carruthers, P. and Smith, P. K. (1996). *Theories of Theories of Mind*. Cambrige University Press, Cambrige.

Castelfranchi, C. and Falcone, R. (2010). *Trust theory: A socio-cognitive and computational model*, volume 18 of *John Wiley & Sons*.

Chevaillier, P., Trinh, T.-H., Barange, M., De Loor, P., Devillers, F., Soler, J., and Querrec, R. (2012). Semantic modeling of Virtual Environments using MASCARET. pages 1–8. IEEE.

Gerbaud, S., Mollet, N., and Arnaldi, B. (2007). Virtual environments for training: from individual learning to collaboration with humanoids. In *Technologies for E-Learning and Digital Entertainment*, pages 116–127. Springer.

Lochbaum, K. E., Grosz, B. J., and Sidner, C. L. (1990). Models of Plans to Support Communication: an Initial Report.

Marsh, S. and Briggs, P. (2009). Examining trust, forgiveness and regret as computational concepts. In *Computing with social trust*, pages 9–43. Springer.

Mayer, R. C., Davis, J. H., and Schoorman, F. D. (1995). An Integrative Model of Organizational Trust. *The Academy of Management Review*, 20(3):709.

Querrec, R., Buche, C., Maffre, E., and Chevaillier, P. (2003). ScuRVi: virtual environments for fire-fighting training. In *5th virtual reality international conference (VRIC03)*, pages 169–175.

Sanselone, M., Sanchez, S., Sanza, C., Panzoli, D., and Duthen, Y. (2014). Constrained control of non-playing characters using Monte Carlo Tree Search. In *Computational Intelligence and Games (CIG), 2014 IEEE Conference on*, pages 1–8. IEEE.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS