A multiagent framework to animate socially intelligent agents

Francisco Grimaldo, Miguel Lozano, Fernando Barber

Computer Science Dept., University of Valencia
Dr. Moliner 50, 46100 Burjassot (Valencia)
{francisco.grimaldo, miguel.lozano, fernando.barber}@uv.es

Abstract. This paper presents a multiagent framework designed to animate groups of synthetic humans that properly balance task oriented and social behaviors. The work presented in this paper focuses on the BDI agents and the social model integrated to provide socially acceptable decisions. The social model provides rationality, to control the global coordination of the group, and sociability, to simulate relations (e.g. friends) and reciprocity between members. The multiagent based framework has been tested successfully in dynamic environments while simulating a virtual university bar, where several types of agents (groups of waiters and customers) can interact and finally display complex social behaviors (e.g. task passing, reciprocity, planned meetings).

Keywords: Artificial societies, social multiagent systems.

1 Introduction and related work

Socially intelligent agents are autonomous problem solvers that have to achieve their goals by interacting with other similarly autonomous entities [7]. Bearing this in mind, multiagent systems can be referred to as *societies of agents*. Within the field of behavioral animation for autonomous synthetic characters, this points to the design of an adequate framework to produce good quality animations for groups of virtual humans. When designing such agents, the main concern has normally been with the decision-making mechanism, as it is the responsible for the actions that will be finally animated. Virtual humans normally play a role (e.g. a virtual guide, seller, client…), thus, it is very important not to avoid the social aspects involved. The notion of socially acceptable decisions has long been of interest in human societies and also in the multiagent community. Traditionally, designers have sought to make their agents rational so that, they can "do the right thing" efficiently (i.e. the shorter the plan the better). Since agents operate in dynamic resource bounded contexts, obstructions rapidly appear when characters act and compete for the use of shared resources (i.e. objects in a virtual environment). Therefore, self interested agents (i.e. agents devoted to accomplish a set of goals) easily come into conflicts even though their goals are compatible, producing low quality animations. Many researches have tried to achieve social skills through multiagent coordination. For example in Generalized Partial Global Planning (GPGP) [4], agents merge the meta-plans describing their operational procedures and figure out the better action in order to maximize the global utility. Another example is Multiagent Planning Language (MAPL) [3], which assigns the control over each resource to a unique agent and uses speech-acts to synchronize planned tasks. Collaboration is supported in the RETSINA system [5] thanks to the use of communicative acts that synchronize tasks and occasionally manage conflicts. Team formation and task coordination were applied to heuristic planning based characters in [6] to adapt better to the dynamism of shared environments. MAS-SOC [1] is a similar system in the literature that aims at creating a platform for multiagent based social simulations. In this context, work is ongoing in order to incorporate social-reasoning mechanisms based on *exchange values* [9]. All these approaches focus on improving the global efficiency of a multiagent society, however, an excess of coordination can produce unethical behaviors which are less interesting to animate. Instead, egalitarian societies of agents are more interesting when simulating groups of virtual humans. In these simulations, autonomous characters should display social behaviors (e.g. interchange information with their partners or grouping and chatting with their friends) and task oriented behaviors, in accordance with the role defined (e.g. a waiter serving at a virtual bar). This kind of socially intelligent animation agents is required in many complex simulation environments: military/civil simulations, virtual cities (e.g. social pedestrians), games, etc. The multiagent framework presented here is directed at designing socially intelligent agents able to balance properly their rationality and sociability, as it is necessary to finally display high quality behavioral animations.

2 Social multiagent framework

The social multiagent framework presented in figure 1 has been developed over Jason [2], which allows the definition of BDI agents using an extended version of AgentSpeak(L) [8]. The animation system (virtual characters, motion tables, etc) is located at the 3D engine, which can run separately. The agent decision-making is defined in the *Agent Specification File*. This file contains the initial beliefs as well as the set of plans that make up the agent's finite state machine. The *Task Library* contains the set of plans that sequence the actions needed to animate a task and achieve a particular goal. Here, modularity is guaranteed since the *Task library* can be changed depending on the environment and the roles being simulated.

As stated above, only rational behaviors are not enough to simulate agent societies, therefore, we have added to the architecture a *Social library* to manage different types of social situations. This library is based on an auction model that uses the social welfare concepts to avoid conflicts and allow the agents to behave in a coordinated way. The auction model is a numerical model, in contrast with the logical model of the task library. Both libraries do not necessarily propose the same behaviour for the agent. In those cases, the conflict is resolved by means of a *sociability factor*, which allows choosing one of the behaviours. The *Social library* also incorporates a reciprocity model in order to promote social interactions between the members of a society. The *Conversational library* contains the set of plans that handle the animation of the interactions between agents (e.g. planned meetings, chats between friends...). Finally, the *Semantic Layer* is in charge of perceiving the state of the world and executing the actions requested by the agents, while ensuring the consistency of the *World Model*.

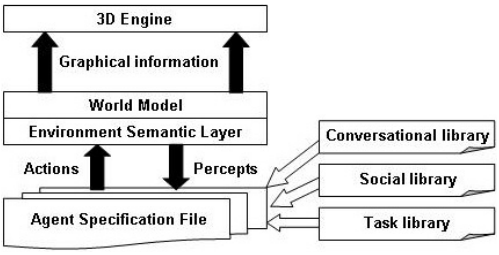

Fig. 1. Social multiagent framework.

3 Social library

The simulation of worlds inhabited by interactive virtual actors normally involves facing a set of problems related to the use of shared limited resources and the need to animate pure social behaviors. Both types of problems are managed by the *Social library* by using a Multiagent Resource Allocation approach [7]. This library allows the auctioning of tasks by any agent in order to reallocate them so that the global social welfare can be increased. Tasks are exchanged between agents using a first-price sealed-bid (FPSB) auction model where the agents express their preferences using *performance* and *social utility functions*.

The performance utility function $U^{i}_{perf}(<i \leftarrow t>)$ of a bidder agent *i* reflects the efficiency achieved when he performs the task *t*. There can be many reasons for an agent to be more efficient: he might perform the task faster than others because of his know-how or it might be using a resource that allows several tasks to be performed simultaneously (e.g. a coffee machine in a virtual bar can be used by a waiter to make more than one coffee at the same time). The utility function has to favor the performance of the agents, but high performances can also be unrealistic for the animation of artificial human societies. For example, if all agents work as much as they can, they will display unethical or robotic behaviors. Furthermore, agents should also show pure social behaviors to animate the normal relations between the members of a society.

Whereas the performance utility function modeled the interest of an agent to exchange a task from an efficiency point of view, we introduce two additional social utilities to represent the social interest in exchanging a task. The aim of social utilities is to promote task allocations that lead the agents to perform social interactions with other agents (e.g. planned meetings with their friends). Negotiation of long sequences of actions is not very interesting for interactive characters, as plans will probably be broken due to the dynamism of the environment and to other unpredictable events. Thus, we define the following social utility functions:

- Internal social utility ($U^{i}_{int}(<i \leftarrow t, j \leftarrow t_{next}>)$): is the utility that a bidder agent *i* assigns to a situation where *i* commits to do the auctioned task *t* so that the auctioneer agent *j* can execute his next task t_{next}.
- External social utility ($U^{i}_{ext}(<j \leftarrow t>)$): is the utility that a bidder agent *i* assigns to a situation where the auctioneer agent *j* executes the auctioned task t while *i* continues his current action.

The winner determination problem has two possible candidates coming from performance and sociability. In equation 1 the welfare of a society is related to performance, hence, the winner of an auction will be the agent that bid the maximum performance utility. On the other hand, equation 2 defines the social winner based on the maximum social utility received to pass the task to a bidder ($U^{*}_{int}(t)$) and the maximum social utility given by all bidders to the situation where the task is not exchanged but performed by the auctioneer j ($U^{*}_{ext}(t)$).

$$winner_{perf}(t) = \left\{ k \in Agents \mid U^{k}_{perf}(t) = \max_{i \in Agents} \left\{ U^{i}_{perf}(< i \leftarrow t >) \right\} \right. \quad \textbf{(1)}$$

$$winner_{soc}(t) = \begin{cases} j & U^{*}_{ext}(t) >= U^{*}_{\mathrm{int}}(t) \\ i & U^{*}_{ext}(t) < U^{*}_{\mathrm{int}}(t) \text{ and } U^{i}_{\mathrm{int}}(t) = U^{*}_{\mathrm{int}}(t) \end{cases} \quad \textbf{(2)}$$

To balance task exchange, social utilities are weighted with a reciprocity matrix (see equations 3 and 4). We define the reciprocity factor w_{ij} for two agents i and j, as the ratio between the number of favors (i.e.tasks) that j has made to i (see equation 5).

$$U^{*}_{\mathrm{int}}(t) = \max_{i \in Agents} \left\{ U^{i}_{\mathrm{int}}(< i \leftarrow t, j \leftarrow t_{next} >) * w_{ji} \right\} \quad \textbf{(3)}$$

$$U^{*}_{ext}(t) = \max_{i \in Agents} \left\{ U^{i}_{ext}(< j \leftarrow t >) * w_{ij} \right\} \quad \textbf{(4)}$$

$$w_{ij} = Favors_{ji} \big/ Favors_{ij} \quad \textbf{(5)}$$

At this point, agents can decide whether to adopt this kind of social allocations or to be only rational as explained previously. They choose between them in accordance with their *Sociability* factor, which is the probability to select the social winner instead of the rational winner. *Sociability* can be adjusted in the range [0,1] to model intermediate behaviors between efficiency and total reciprocity. This can provide great flexibility when animating characters, since *Sociability* can be dynamically changed thus producing different behaviors depending on the world state.

4 Application example

In order to test the presented social multiagent framework, we have created a virtual university bar where waiters take orders placed by customers (see figure 2a). The typical locations in a bar (e.g. a juice machine) behave like resources that have an associated time of use to supply their products (e.g. 7 seconds to make an orange juice) and they can only be occupied by one agent at a time.

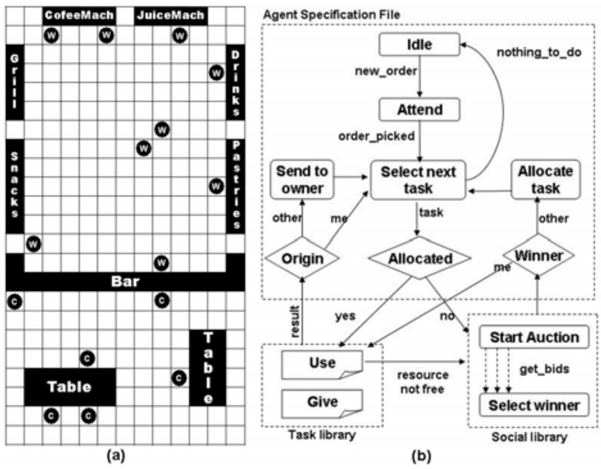

Fig. 2. (a) Virtual university bar environment, (b) Waiter specification.

The waiters are governed by the finite state machine[1] shown in figure 2b, where orders are served basically in two steps: first, using the corresponding resource (e.g. the grill to produce a sandwich) and second, giving the product to the customer. Tasks are always auctioned before their execution in order to find good social allocations. Equations 6 and 7 define the utility values returned by the performance utility function for these tasks. This function aims at maximizing the number of parallel tasks being performed and represents the waiters' willingness to serve orders as fast as possible. Social behaviors defined for a waiter are oriented to animate chats between his partners. Therefore, waiters implement the internal and external social utility functions detailed in equations 8 and 9, where *Near* computes the distance between the agents while they are executing a pair of tasks. These functions evaluate social interest as the chance to meet a partner in the near future (i.e. a planned meeting).

$$U^{i}_{perf}(i \leftarrow \text{'}Use\text{'}) = \begin{cases} 1 & \text{if } [(\mathrm{i} = \mathrm{Auctioneer}) \text{ and } (\mathrm{IsFree(Resource)})] \text{ or} \\ & [\mathrm{IsUsing(i, Resource)} \text{ and } \mathrm{not(IsComplete(Resource))}] \\ 0 & \text{Otherwise} \end{cases} \cdot \quad \textbf{(6)}$$

$$U^{i}_{perf}(i \leftarrow \text{'}Give\text{'}) = \begin{cases} 1 & \text{if } [(\mathrm{i} = \mathrm{Auctioneer}) \text{ and } (\mathrm{nextAction} = \mathrm{NULL})] \text{ or} \\ & [\mathrm{currentTask} = \text{'Give' and } \mathrm{not(handsBusy} < 2)] \\ 0 & \text{Otherwise} \end{cases} \cdot \quad \textbf{(7)}$$

$$U^{i}_{\mathrm{int}}(< i \leftarrow t, j \leftarrow t_{next} >) = \begin{cases} 1 & \text{if } \mathrm{IsFriend(i, j)} \text{ and } \mathrm{Near(t, t_{next})} \text{ and} \\ & \mathrm{ExecTime(t_{next})} > \mathrm{RemainTime(currenTask)} \\ 0 & \text{Otherwise} \end{cases} \cdot \quad \textbf{(8)}$$

[1] Specified by means of plans in Jason's extended version of AgentSpeak(L)

$$U^{i}_{ext}(j \leftarrow t) = \begin{cases} 1 & \text{if IsFriend(i, j) and Near(currentTask, t)} \\ 0 & \text{Otherwise} \end{cases} . \quad \textbf{(9)}$$

On the other hand, customers place orders and consume them when served. Now, we are not interested in improving customer performance but in animating interactions between the members of a social class. Thus, we have implemented three classes of customers that use auctions to solve the problem of *where to sit*. A finite state machine similar to that in figure 2b governs the actuation of customers. Depending on his or her sociability factor, a customer can randomly choose a chair or start an auction to decide where to sit and consume. This auction is received by all customers in the bar, which use the external social utility function defined in equation 10 to promote meetings with others of the same class. We define the performance and the internal social utility functions as 0 since task passing is not possible in this case (i.e. no-one can sit instead of another customer). Finally, when a social meeting emerges, both waiters and customers use the plans in the *Conversational Library* to sequence the speech-acts needed to animate commitments, greetings or simple conversations.

$$U^{i}_{ext}(j \leftarrow 'Sit') = \begin{cases} 1 & \text{if IsSameClass(i, j) and IsConsuming(i, auctionedTable)} \\ 0 & \text{Otherwise} \end{cases} . \quad \textbf{(10)}$$

5 Results

To illustrate the effects of the social techniques previously applied we have animated the virtual university bar example with 10 waiters serving 100 customers both with different sociability factors (visit http://www.uv.es/~agentes to download some animation videos). To measure the efficiency of a group of waiters we use the ratio between the optimal simulation time and the real simulation time (see equation 11). *Throughput* is an indicator in the range *[0,1]* that estimates how close a simulation is to the ideal situation in which the workload can be distributed among the agents and no collisions arise.

$$Throughput = \frac{N_{tasks} * \overline{T_{task}} \big/ N_{agents}}{T_{sim}} . \quad \textbf{(11)}$$

Figure 3a shows the *Throughput* obtained by different types of waiters versus self-interested agents (i.e. agents with no social mechanisms included). Self-interested agents collide as they compete for the use of the shared resources and these collisions produce high waiting times as the number of agents grows. We can enhance this low performance with elitist agents (*Sociability* = 0) which exchange tasks with others that can carry them out in parallel thus reducing the waiting times for resources. Nevertheless, they produce unrealistic outcomes since they are continuously working if they have the chance, leaving aside their social relationships (e.g. chats between friends). The *Sociability* factor can be used to balance rationality and sociability. Therefore, the *Throughput* for the sort of animations we are pursuing should be placed somewhere in between elitist and fully reciprocal social agents (*Sociability*=1). On the other hand, figure 3b demonstrates that the higher the *Sociability* factor is, the larger the number of social meetings that will be performed by the customers when they sit at a table.

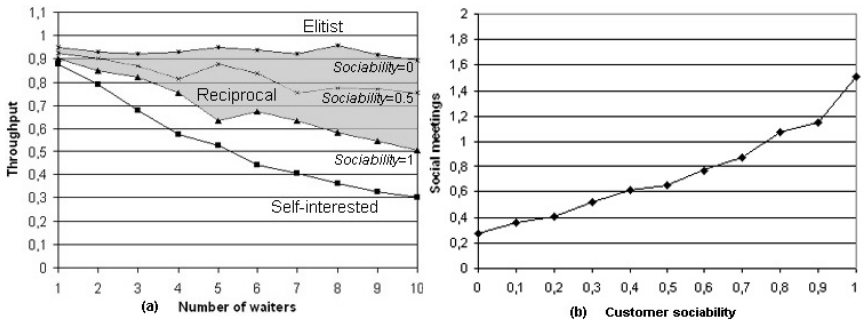

Fig. 3. (a) Waiter *Throughput*, (b) Customer social meetings.

Finally, table 1 compares the amount of time devoted to execute each type of task in executions with 10 elitist waiters (*Sociability*=0) and 10 fully reciprocal social waiters (*Sociability*=1). The irregular values in the columns T_{use} and T_{give} on the left side of the table demonstrate how some agents have specialized in certain tasks. For instance, agents 2, 5, 9 and 10 spend most of their time giving products to the customers while agents 3 and 7 are mainly devoted to using the resources of the bar (e.g. coffee machine, etc). Although specialization is a desirable outcome in many multiagent systems, egalitarian human societies need also to balance the workload assigned to each agent. On the right side of the table, fully reciprocal social waiters achieve equilibrium between the time they are giving products and the time they are using the resources of the environment (see columns T_{use} and T_{give}). Furthermore, the reciprocity factor balances the number of favors exchanged among the agents (as shown in *Balance* column). A collateral effect of this equilibrium is the increase in the waiting times, since social agents will sometimes prefer to meet his friends in a resource than to reallocate the task (compare columns T_{wait}).

6. Conclusions and Future Work

The animation of groups of intelligent characters is a current research topic with a great number of behavioral problems to be tackled. We aim at incorporating human style social reasoning in character animation. Therefore, this paper presents a technique to properly balance social with task-oriented plans in order to produce realistic social animations. The multiagent animation framework presented allows for the definition of different types of social agents: from elitist agents (that only use their interactions to increase the global performance of the group) to fully reciprocal agents.

These latter agents extend the theory of social welfare with a reciprocity model that allow the agents to control the emergence of unethical behaviors and promote social interactions among the members of a group.

Work is ongoing to provide the agents with mechanisms to self-regulate their *Sociability* factor depending on their social relations and on their previous intervention. Thus, agents will be able to dynamically adjust to the situation in order to stay within the boundaries of good quality animations at all times.

Table 1. Time distribution for 10 waiters in the bar (time values are in seconds).

	Sociability = 0				*Sociability* = 1			
Agent	T_{wait}	T_{use}	T_{give}	*Balance*	T_{wait}	T_{use}	T_{give}	*Balance*
1	0	32	19	-6	16	69	34	-2
2	3	4	26	-3	18	58	24	-2
3	14	52	1	28	41	45	16	0
4	3	16	28	-3	48	60	27	3
5	0	7	30	-16	34	58	12	-1
6	3	37	17	-1	48	64	14	-2
7	0	67	4	21	18	48	24	1
8	0	45	17	1	33	45	24	4
9	7	5	23	-11	46	36	21	0
10	1	6	41	-10	27	56	20	-1

Acknowledgments. This work has been jointly supported by the Spanish MEC and European Commission FEDER funds under grants Consolider Ingenio-2010 CSD2006-00046 and TIN2006-15516-C04-04.

References

1. R.H. Bordini, A.C. da Rocha, J.F. Hübner, A.F. Moreira, F.Y. Okuyama and R. Vieira. A Social Simulation Platform Based on Agent-Oriented Programming. *Journal of Artificial Societies and Social Simulation*, vol.8, no.3, 2005.
2. R.H. Bordini and J.F. Hübner. Jason, 6th of March 2007. http://jason.sourceforge.net/.
3. M.Brenner. A multiagent planning language. *In Proc. of ICAPS'03: Workshop on PDDL*, 2003.
4. K.S. Decker and V.R. Lesser. Designing a family of coordination algorithms. *Readings in Agents*. Huhns and Singh editors, 1997.
5. J. A. Giampapa and K. Sycara. Team-Oriented Agent Coordination in the RETSINA Multi-Agent System. *On Tech. Report CMU-RI-TR-02-34*, Robotics Institute-Carnegie Mellon University, 2002.
6. F.Grimaldo, M.Lozano, and F.Barber. Integrating social skills in task-oriented 3D IVA. *In Proc. of IVA'05: International Conference on Intelligent Virtual Agents*. Springer, 2005.
7. L.M. Hogg and N.Jennings. Socially intelligent reasoning for autonomous agents. *IEEE Transactions on System Man and Cybernetics*, 31(5), 2001.
8. A.S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. *In Proc. of MAAMAW'96*, LNAI 1038, pages 42-55, 1996.
9. M.Ribeiro, A.C. da Rocha and R.H. Bordini. A System of Exchange Values to Support Social Interactions in Artificial Societies. *In Proc. Of AAMAS'03: Autonomous Agents and Multiagent Sistems*. ACM, 2003.