Real-time in Multi-Agent Systems

Claude Duvallet and Bruno Sadeg
LIH, Faculté des Sciences et Techniques
25, Rue Philippe Lebon
76058 Le Havre Cedex, France
{duvallet, sadeg}@univ-lehavre.fr

Alain Cardon
LIH, IUT du Havre
Place Robert Schuman
76610 Le Havre Cedex, France
cardon@iut.univ-lehavre.fr

Abstract

In a lot of real-world applications, computer systems assist the human in executing some tasks, particularly in crucial activities. These systems are used to detect incidents that may take place in these applications and then allow the managers to take the appropriate decisions in order to stop things getting any worse. These decisions must be taken as earlier as possible: they are submitted to temporal constraints. The applications where risks (technological or economical) may happen are usually complex and distributed through a lot of sites. In order to manage efficiently these applications, the multi-agent paradigm seems to be the best approach. Furthermore, in some situations, an inaccurate or incomplete result obtained early is more useful than an accurate or complete result obtained late. In this paper, we present ANYMAS (ANYtime Multi-Agent System), a new approach for real-time multi-agent systems based on anytime algorithms. In ANYMAS model, real-time aspects are considered not only at an agent level but at the organization system level. The idea is based on two components added to an existing component based model: a celerity component and a temporal component. Besides, it is possible to evaluate dynamically the system behavior.

Keywords

Real-time systems, Multi-agent systems, anytime algorithms.

1 Introduction

Systems that could manage efficiently applications where technological or economical risks may occur must be able to respond timely to abnormal situations (that is to avoid risks or to limit their consequences as earlier as possible). Moreover, in a lot of situations, partial or inaccurate results obtained early are more useful than complete or accurate results obtained late. Such systems could be Multi-Agent Systems (MAS) [9] where are added real-time aspects (RT-MAS)[6][7]. The applications where RT-MAS may be an appropriated approach are stock market management, multimedia systems and so on.

A promised approach to include real-time aspects in MAS is anytime approach [1] which can lead with partial and/or progressive responses. In this paper, after a brief description in section 2 of classical real-time systems and anytime techniques, we describe ANYMAS model (ANYtime Multi-Agent System), in section 3, a model we design to take into account real-time aspects in MAS. In section 4, we give an example of ANYMAS's implementation and we conclude by describing ANYMAS's future extensions.

2 Real-Time Multi-Agent Systems

Real-time systems are systems which are submitted to temporal constraints (deadlines, periods,...). They must be able to react timely facing the environment which they control. However, classical real-time systems[8] are not suited to manage some real-time applications where approximate responses obtained timely may be more useful than complete responses obtained late. For example, in a market management's application, a buyer or a seller must take a decision as earlier as possible. So, a new approach, called anytime, has been appeared which allow to build progressive solutions. Indeed, anytime techniques allow the system to provide useful partial results at any time. Anytime approach is used to take into account real-time in MAS [5] because in a lot of applications managed by MAS the timeliness of a result is privileged, and the quality of the result may be altered. Indeed, in applications like market management or process control, we have not enough time to wait for a complete result in or-

der to take a decision. In the next section, we present ANYMAS, a MAS model where anytime approach is used to implement real-time aspects in order to manage such situations.

3 ANYMAS model

ANYMAS architecture belongs partially to a component-based architectures [2] where some improvements are done. We add an introspection component agent architecture in order to allow agents to predict the time needed to execute tasks. This component may modify dynamically previous predictions if necessary (if there is not enough time to terminate execution). A task being composed of subtasks, an agent's celerity compute subtasks necessary time during a MAS training mode. To this purpose, agents can create temporal agents, called reification of a temporal agent, used to reduce communications between agents.

The MAS execution is tightly dependent on the communications between agents. According to the available time, it might be interesting to influence the behavior of agents so that the system can provide useful partial results. Therefore, in ANYMAS, we encourage some groups of agents that will allow getting this result. In summary, to control duration of system's execution, it is necessary to control interactions between groups of agents. This control is done by temporal agents.

We call agent's celerity its capacity and its speed to build a result. This notion of celerity also allows to measure agents performances.

In ANYMAS model, an Augmented Transition Network (ATN) is used to allow the system to stabilize agents in different states. ATN is inherited from DIMA agents architecture [2]. During training mode, agent's speed is constantly computed in order to get an average value of successive speeds. These average time values are discretized in a unit of time. This construction is made in the following way:

Given an ATN with n states (see figure 1, where n is set to 5), $d_{i,i+1}$ $(1 \leq i < n-1)$ is the transition duration between state i and state i+1; Let a variable k be a subdivision of the time and $n_{i,i+1}$ the number of k subdivisions between states i and i+1.

Therefore, we obtain the following equation:

$$d_{i,i+1} = k * n_{i,i+1}, \text{ where i=1,2,3,...} \Rightarrow \text{k} = \frac{d_{i,i+1}}{n_{i,i+1}}$$

For example, in figure 1, we obtain:

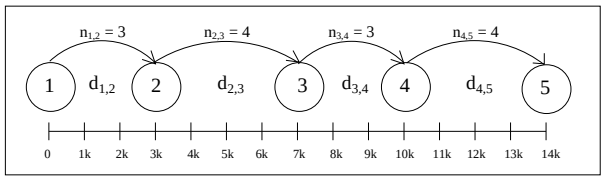

Figure 1: Example of discretisation of time in ATN

$$k = \frac{d_{1,2}}{n_{1,2}} = \frac{d_{2,3}}{n_{2,3}} = \frac{d_{3,4}}{n_{3,4}} = \frac{d_{4,5}}{n_{4,5}}$$

The number of subdivisions for the first interval being set, an algorithm (see figure 2) allows to determine (1) the value of the k variable and (2) the time subdivisions between states. ε is a variable providing a degree of time precision allowed. k value represents the computed unit value which measure the transition time between two successive states of the ATN.

```
n1,2 ← 1
Increase ← true
Last State ← false
WHILE ((|di,i+1 - k * ni,i+1| > ε)
        AND ((i+1) Not Last State))
        i ← 1
        n1,2 ← n1,2 + 1
        k = round (di,i+1 , ni,i+1)
        WHILE (|di,i+1 - k * ni,i+1| < ε
                AND (i+1) Not Last State))
                i ← i + 1
                di,i+1 = di,i+1 + di−1,i - k * ni−1,i
                ni,i+1 = round (di,i+1 / k)
        END WHILE
END WHILE
```

Figure 2: Algorithm of discretisation of time

The algorithm was implemented in Java. A simulation of this algorithm (see figure 3) illustrate how it works. In the first and the second column, we represent two successive states of the algorithm's execution. The third column represents the execution time interval between two states. The fourth column represents the results obtained after the execution of the algorithm. In this example, for $\varepsilon = 5$, the k subdivision obtained is equal to 4. In our experimentations the algorithm behavior is compliant to our expectations. In summary, based on slack time's execution, the goal of the algorithm is to evaluate agent's celerity in order to determine whether or not it will terminate excution before deadline. We outline here the ANYMAS

predictive aspect.

State i	State i+1	Time in ms	Time in K units
1	2	10556 ms	2640 K
2	3	20173 ms	5043 K
3	4	5443 ms	1361 K
4	5	34256 ms	8564 K
5	6	29654 ms	7413 K
6	7	32432 ms	8108 K
7	8	11284 ms	2821 K
8	9	27236 ms	6809 K
9	10	13255 ms	3314 K
10	11	34256	8564 K
11	12	11284	2821 K
Value of K : 4			

Figure 3: Simulation of the algorithms

The reification agents are produced by agents that should have a component used to reify temporal agents. A temporal agent component acts like a threshold function : it is triggered when the number of agents or the number of communications between them exceed a certain threshold. When the threshold is exceeded, it is necessary to synchronize agents concerned by the communication exchanges. The objective is to verify if it is possible or not to connect these agents to an existing temporal agent, that has yet controled other communications. If such a temporal agent exists then a new agent is merely inserted in acquaintances of the temporal agent, otherwise a new temporal agent is reified and its acquaintances are initialized with the two agent's identity, which have triggered this reification.

The temporal agent is a lightweight agent that scrutinizes the state of a small group of agents and controls permanently their communications. It requests these agents about their celerity and the number of exchanged communications. They had to take into account the global deadline and must cooperate with other temporal agents in order to negotiate the agent group importance. When it becomes necessary, temporal agent may decide to reduce the communicative activity of the agent group which it supervises. Reducing the activity of certain agents groups allows other groups to reach an exploiting result more quickly. Then a progressive solution can be constructed.

4 An example of ANYMAS's application

In order to validate ANYMAS model, MarketPlace management is used which is characterized by negociation process. Negotiation process may be viewed as a progressive process and then can be implemented by anytime techniques. Numerous messages are exchanged. Three types of agents have been introduced in this application : seller agent, buyer agent and message agent. The seller agent and the buyer agent have to satisfy time contraints in order to take good decisions early and then to produce the best performances as possible.

This application was implemented using the platform MadKit [3][4]. In order to implement ANYMAS model, we have to increase functionnalities of the agents of the Madkit platform. As we can see in the figure 4, an ATN, a clock and a function of discretisation are added to the initial structure of the agent. The celerity's component was implemented with both an ATN and a disretisation function.

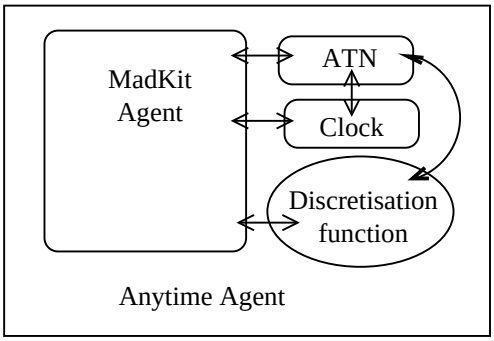

Figure 4: Construction of an anytime agent with MadKit

5 Conclusion and future work

In this paper, we have presented ANYMAS model to take into account real-time aspects in multiagent systems. This model based on anytime techniques allows to construct real-time complex applications. We have chosen to implement an application of Marketplace. This application has allowed to validate a part of ANYMAS model. An extended version of the Marketplace is being currently implemented in the context of a system used to help users to make their transactions (sell and buy products) in the best conditions. Our perspective is to implement ANYMAS

model in hard real-time application [8], that is in sirens'manager system of a town. Indeed, ANYMAS need to be completed in order to take into account emergence aspects in anytime component of multiagent systems. Other aspects need to be detailled or completed (anytime query manager, RT-MAS design methodologies, time constraints specification, ...).

References

[1] J. Grass and S. Zilberstein. Programming with anytime algorithms. In *Proceedings of the 14^{th} Internationnal Joint Conference on Artificial Intelligence*, pages 22–27, Montreal, 1995.

[2] Z. Guessoum and J.-P. Briot. From concurrent objects to autonomous agents. In *8^{th} European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Ronneby, 1997.

[3] O. Gutkneco and J. Ferber. Madkit: Organizing heterogeneity with groups in a platform for multiple multi-agent systems. Technical report, LIRMM, UMR 9928, Université de Montpellier II, 1999.

[4] MADKIT. *http://www.lirmm.fr/~gutkneco/madkit.*

[5] A.I. Mouaddib, F. Charpillet, Y. Gong, and J.P. Haton. Real-time specialist society. In *Proceedings IEEE, Singapore International Conference on Intelligent Control and Instrumentation*, pages 751–754, 1992.

[6] M. Occello, Y. Demazeau, and C. Baeijs. Designing organized agents for cooperation with real time constraints. In *1^{st} International Workshop on Collective Robotics*, pages 25–37, Paris, 1998.

[7] S. Ren and G. Agha. Rtsynchronizer: Language support for real-time specifications in distributed systems. In *ACM SIGPLAN Workshop on Languages, Compilers and Tools for Real-Time Systems*, La Jolla, California, 1995.

[8] J.A. Stankovic and K. Ramamritham. *Hard real-time systems: a tutorial.* IEEE Computer Society Press, 1988.

[9] M. Wooldridge and N.R. Jennings. Agent theories, architectures and language : A survey. In M. Wooldridge and N. Jennings, editors, *Intelligent Agents, ECAI 1994*, volume LNAI 890, pages 1–32. Springer Verlag, 1994.