A&QT-R 2002 (THETA 13)
2002 IEEE-TTTC International Conference on Automation, Quality and Testing, Robotics
May 23 – 25, 2002, Cluj-Napoca, Romania

# PRELIMINARY EXPERIMENTS ON BEHAVIOR PATTERN DETECTION FOR COOPERATING AGENTS

**Ioan Alfred Letia, Radu Razvan Slavescu**

*Department of Computer Science, Technical University of Cluj-Napoca*
*Baritiu 28, RO-3400 Cluj-Napoca, Romania*
*{letia, srazvan}@cs-gw.utcluj.ro*

Abstract

This paper presents the results we obtained in developing a mechanism for partner selection in a multi-agent system. This is based on the trust level of different agents and also on behavior pattern mining. Trust level is measured for every agent in the system and for every task s/he could perform. For every task, the agent having the highest trust rate will be selected; however, this criterion could be overridden if a certain behavior pattern was detected for some agent. This combined approach could lead to a better accuracy in partner selection.

Keywords: agent cooperation, data mining, trust

## 1. INTRODUCTION

In many applications, one agent alone cannot accomplish the goal. Consequently, cooperation between agents is necessary in order to fulfil the goal of the system. In this context, the notion of delegation appears to be of paramount importance in multi-agent systems, basically because it is strongly connected with that of cooperation.
However, agents in real systems are neither able nor committed to cooperate with someone else. That's why a certain agent needs to be endowed with a mechanism which could help him in taking the proper decision when choosing another agent in order to delegate him a task. Two different approaches of the problem were suggested. The first one asks to cooperate only with those agents who are issued by a reliable source; in order to prove this thing, an agent must be able present a confidential information (e. g. a password) which is known only by that source. All the other agents are considered not to be reliable, so cooperation with them is avoided. However, this mechanism allows an agent to cooperate with another one only if there exists a previous agreement between their owners, which is a serious constraint in open multi-agent systems. Moreover, once the agent is accepted (i.e. once he is granted the password), cooperation with him will be accepted no matter how bad its performance will be. This also imposes an artificial restriction over the set of the potential partners of an agent: agents with good performances will be rejected if there is no

agreement with their sources, while agents with poor performances will be accepted if such an agreement exists, despite the bad results of such a cooperation.

An alternative approach is to base the decision of delegation on the concept of reputation or trust. The trust an agent (the trustor) has in another agent (the trustee) will be the result of the trustee's previous behaviour. This approach offers more flexibility when addressing the issues we have just presented. In the following sections, we will present some models of trust that were developed so far and how these models could be used for plan selection and for appointing agents for different tasks in a plan.

The rest of this paper presents a combined approach of the same issue, built on trust and pattern discovery. Section 2 explains a trust-based approach of the partner selection problem. Section 3 briefly describes a data mining algorithm designed for discovering patterns in large amounts of data. Section 4 gives an image of the agent environment we used for simulations, while Section 5 describes an example and the global results we obtained on synthetic simulation data. Section 6 presents the advantages of the method we started to develop, some of its possible limitations together with the goals for future work.

## 2. TRUST AND PARTNER SELECTION

The notion of trust is very well known at common sense level. Basically, trust is the attitude a certain agent has is a situation where the turn of events is uncertain and the agent expects the situation will move into a favourable direction [7,8].

A complex analysis of trust could be found in [3]. The following ideas concerning an agent endowed with the notion of trust are pointed:

- the agent is confronted with an ambiguous path, that could lead either to a beneficial or to a harmful event;
- the agent perceives that the occurrence of both events is contingent on some other agent's behaviour;
- the strength of the harmful event is perceived to be greater than that of the beneficial one.

In [4], trust is regarded as a subjective probability: " trust is a particular level of subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such an action (or independently of his capacity ever to be able to monitor it) and in context in which it affects its own actions". The trust formalisation has three underlying ideas: delegation depends on the trust level; the trust level depends on the particular task that is to be done; the trust level depends on the previous behaviour.

In [1,2] a cognitive model of trust is presented. According to this model, an agent's decision of delegating a task to another agent is the result of the trust the first agent has in the second one. Trust is seen as "the mental counter-part" of delegation [2]. So, the trustor should be a cognitive agent, i.e. an agent endowed with beliefs and goals.

The second important idea regarding trust modelling is presented in [7,8]. In this approach, three types of trust are identified: a basic trust, evaluating the general disposition of the trustor; a general trust of x in y, which is the view agent x has about the general capabilities of agent y; and the situational trust of x in y regarding situation s, which estimates the opinion agent x has about the capability of agent y to perform well in the given situation s. To conclude, we can say the most accurate evaluation is the last one,

which takes into consideration both the particular agent and the particular task which has to be performed.

Another important idea for trust evaluation is exposed in [5]: the trust level is seen as a function which maps the previous experiences sequences into a set of trust values. Such a function is called a trust evolution function. In [5], the notion of trust update function is introduced. This function has as arguments the current trust value and the current experience value and returns the next trust value. An algorithm for conversion between the two types of functions could be found in the mentioned article [5]. The same paper emphasises the importance of the recent experiences over the older ones and presents a function, which gives a specific weight to every experience, accordingly to its distance in the past.

Our previous work was concerned with using these ideas in order to set up a partner selection mechanism. In [6], we presented the results we obtained in this direction, by evolving a system for doing this type of experiments. The system performs the following actions: an agent decides to achieve a certain goal. In order to do that, he will choose a plan among a library of plans; the problem is to delegate every step of the plan to one agent. In order to evaluate the competence level, we have used a particular function having the so-called property of "sincere history representation" (i.e. which produces different values for different sequences of experiences). Actually, we used that function in form of a trust update function. For a demonstration of the equivalence of the two, see [5].

To be more specific: trust values are taken from [-1, 1]; for trust actualisation, we used the following formula:

$$f(e,t) = 0.5 * e + 0.5 * t$$

where:
t is the current trust value (a real number from [-1,1])
e is the value of the last experience (and could be either -1 or 1)
f is the new trust value, which will become the new current trust value of the planning agent x in agent y (who has performed the task t), about his capacity of performing tasks of type t.

This function assigns different trust values to different sequences of experiences and gives quite a high importance to those experiences that had place recently.

## 3. MINING SEQUENTIAL PATTERNS

This section sketches the algorithm suggested in [9]. The goal of this algorithm is to discover sequential patterns in a large list of data. The algorithm offers the opportunity to deal with time constraints and time sliding windows and also works well for large amounts of data.

In brief, the algorithm makes multiple passes over data. In the first pass, it determines the number of data-sequences that include a certain item and gets the most frequent items. These are regarded as 1-element frequent sequences. In the next pass, the algorithm determines longer sequences than those in the previous one. The algorithm terminates when there are no frequent sequences at the end of one pass.

Candidate generation involves 2 steps:

1. Join: this phase combines $L_{k-1}$ with $L_{k-1}$. Two sequences $s_1$ and $s_2$ could be joined if the subsequence obtained by dropping the first item of $s_1$ is the same as the sequence obtained by dropping the last item of $s_2$.
2. Prune: this phase deletes candidate sequences having a contiguous (k-1)-subsequences whose support count is less than the minimum support

Candidate counting aims to detect all candidate sequences which are contained in a data sequence. In order to do this, 2 techniques are used:

1. a hash tree to reduce the number of explored candidates
2. some transformation of data sequence representation in order to improve search efficiency.

For more details on this pattern detection algorithm, see [9].

## 4. AGENT ENVIRONMENT

A set of Prolog agents has been developed and used for simulations in order to study trust-based cooperation for achieving a certain goal. The aimed goal is to appoint agents to tasks belonging to a certain plan.

As showed in the previous sections, trust depends on three factors: the trustor, the trustee and the task [8]. So, for every agent in the system and for every possible task, the planning agent keeps a trust value; this is computed on the basis of the previous performances the trusted agent has had when performing that task. Appointing agents to different tasks in the plan is done by using these levels of the trust: for every step in the plan, the most trusted agent regarding that task is chosen. After a step execution is completed, the system checks to see if there would have been a better option for agent selection and updates the trust function. The program displays whether the choices for agents were correct

## 5. RESULTS

In this section, we will present the results obtained for a number of 2 agents, each of which being able to perform 3 tasks. The planning agent appoints one of the three agents to a task in the plan based on the likelihood of that task to be accomplished as explained above. After that, the planning agent observes each agent's performance and modifies the trust value in that particular agent according to the agent's behaviour.

Let us consider the data test in the table below:

| Step | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Agent 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Trust 1 | .0 | .5 | .25 | .125 | .5625 | .2812 | .1406 | .5703 | .2852 | .1426 |
| Agent 2 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| Trust 2 | .0 | .0 | .0 | .5 | .25 | .625 | .8125 | .4062 | .7031 | .8515 |
| Chosen | 1 | 1 | 1 | 2 | 1 | 2 | 2 | 1 | 2 | 2 |
| Best | 1 | 1 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 1 |

**Table 1: Agent selection with no behaviour pattern detection**

The rows Agent1 and Agent2 present the performance of agent 1, respectively agent 2 at each step. The rows Trust1 and Trust2 present the trust value the planning agent has about agent 1, respectively agent 2 before the delegation decision is taken. The last 2 rows show the agent chosen for delegation (based on its trust level) and the best option for each step.

As one can see, agent 1 chosen to cheat, using a repetitive strategy: it first makes a correct move in order to get a high trust value and then cheats 2 times. This strategy is successful from its point of view and leads to 3 wrong decisions (in step 3, 5 and 8) due to the artificially accumulated trust value.

The improvement we suggest in order to avoid this is to use a behaviour pattern detection system as that described in Section 3. We use it to monitor each agent's behaviour and to try to discover some repeated sequences (e.g. 1 0 0, for agent 1, in our case). We'll delegate a task to an agent only if it has the highest level of trust and no pattern in its behaviour predicts a cheating in that step.

With this approach, the delegation decision for the same case would look like this:

| Step | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Agent 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Trust 1 | .0 | .5 | .25 | .125 | .5625 | .2812 | .1406 | .5703 | .2852 | .1426 |
| Agent 2 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| Trust 2 | .0 | .0 | .0 | .5 | .25 | .625 | .8125 | .4062 | .7031 | .8515 |
| Chosen | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Best | 1 | 1 | 2 | 1 | 2 | 2 | 1 | 2 | 2 | 1 |

**Table 2: Agent selection with behaviour pattern detection**

In this particular case, cheating was avoided altogether.

Let M be the number of situation in which a good agent is chosen instead of a bad one and N is the total number of situation in which we have to choose between an agent with a good behaviour and an agent with a bad behaviour. The ratio M/N could be seen as prediction accuracy; its evolution versus the number of tests is presented in the table below:

| Test | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Accuracy | 0 | 0.33 | 0.5 | 0.6 | 0.66 | 0.7 | 0.6 | 0.5 | 0.5 | 0.6 |
| New accuracy | 0 | 0.4 | 0.56 | 0.66 | 0.75 | 0.8 | 0.8 | 0.85 | 0.8 | 0.85 |

**Table 3: Selection accuracy compared**

## 6. CONCLUSIONS

Our experiments have showed that, after a certain number of iterations, the system will choose those agents who best fit, in that very moment, for the task aimed to be accomplished. In systems where being appointed with a certain task could mean some kind of advantage (such as gaining some money), some agents could try to cheat in order to be selected for a task even if their performance is not the best. Experiments proved that it is possible to cheat, but an "honest" agent, who really tries to improve itself will be eventually preferred to an agent who tries to accumulate reputation first and, after that, doesn't try to

get good results any more. Adding a behaviour pattern detection could lead to future enhancements of this approach because it prevents an agent from cheating based on a repetitive strategy.

The problem remains the trade-off between the trust function stability and sensibility. It should be stable enough to give a long term vision of the agent's behaviour, but sensible enough to make sure that, in case of distrustful behaviour, an agent will not be protected too long by his previous reputation. Another drawback could be a cheating strategy including a random element. It is not clear so far whether this kind of strategy could be addressed by the approach we presented. These issues are to be addressed in further developments.

## 7. REFERENCES

[1] Castelfranchi, C. and Falcone, R. (1998) Principles of trust for MAS: "Cognitive anatomy, social importance, and quantification", *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS-98)*, Paris, France, 72-79.

[2] Castelfranchi,C. and Falcone, R. (1999). The dynamics of trust: from beliefs to actions. *Proceedings of the First International Workshop on Deception, Trust and Fraud in Agent Societies*.

[3] Deutsch, M. (1962): Cooperation and trust: Some theoretical notes, *Nebraska Symposium on Motivation, Nebraska University Press*.

[4] Gambetta, D.: Trust. Basil Blackwell, 1990.

[5] Jonker, C. and Treur, J. (1999): Formal analysis of models for the dynamics of trust based on experiences, *In F.J. Garijo and M.Boman, editors, Multi-Agent System Engineering (LNAI 1647),* Springer-Verlag, 221--232.

[6] Letia, I.A. and Slavescu, R. R. (2000) Preliminary experiments on credibility in a large group of cooperating agents, Timisoara.

[7] Marsh, S. (1992) Trust and reliance in multi-agent systems, In Artificial Social Systems -*Selected Papers from the Fourth European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, MAAMAW-92 (LNAI Volume 830)}.

[8] Marsh, S. (1994): Formalising trust as a computational concept, PhD thesis, University of Stirling.

[9] Skirant R. and Agrawal, R. (2000) Mining Sequential Patterns: Generalisation and performance improvements, IBM research report.