Multi-Agent System for University Course Timetable Scheduling

Mihaela Oprea[1]

(1) University Petroleum-Gas of Ploieşti, Department of Informatics
Bd. Bucureşti Nr. 39, Ploieşti, RO-100680, ROMANIA
E-mail: mihaela@upg-ploiesti.ro

Abstract

Many real-world applications are mapped into combinatorial problems. An example of such problem is timetable scheduling. In this case, the two basic characteristics can be defined by its distributed and dynamic environment. One efficient solution could be provided by an agent-based approach. A timetable scheduling problem can be modelled as a multi-agent system that provides the final schedule by taken into account all the restrictions. In this paper it is presented a preliminary research work that involves the development of a multi-agent system for university course timetable scheduling. We focus on the architecture of the multi-agent system, and on the evaluation of the communication process, by using the interaction diagrams.

Keywords: Artificial intelligence, Multi-agent systems, Timetable scheduling

1 Introduction

The general task of solving timetable scheduling problems is iterative and time consuming. In real world applications, the participants to the timetable scheduling have conflicting preferences, which make the search for an optimal solution an NP-hard problem. In order to solve the problem it is necessary to find a compromise between all the professors' requirements, usually conflicting (e.g. day, time). The constraints are related to the availability, timetabling and preferences of each professor, to rooms availability, number of students, and curricula. In order to solve this problem for the particular case of university course timetable scheduling we have adopted the agent-based approach.

Multi-agents systems (MAS) are concerned with coordinating behavior among a collection of autonomous intelligent agents (e.g. software agents) that work in an environment. Sometimes, software agents are designed to reconcile their own interests with the constraints implied by other agents. One type of software agents is given by expert assistants who enable us to automate certain manual tasks and who work more efficiently. Expert assistant is a term given to an intelligent software agent that performs certain tasks on our behalf (Wooldridge and Jennings, 1995; Weiss, 1999). For example, our daily organiser is an assistant. The complexity of multi-agent systems is generally higher than that corresponding to conventional software systems and their success rely on properly designed and well tested subsystems. Also, in the particular case of timetable scheduling, the MAS could find an optimal or a sub-optimal solution using mainly inter-agent communication (with minimal message passing).

In this paper, it is presented the architecture of a multi-agent system, MAS_UP-UCT, that is under developping, and has as main purpose the modelling of the university courses timetable scheduling. We shall describe the architecture of the multi-agent system, focusing on the mapping of a course timetable scheduling in terms of intelligent agents, and finally, we shall make a preliminary evaluation of the multi-agent system.

2 University Course Timetabling Problem

The scheduling problem can be defined as a problem of finding the optimal sequence for executing a finite set of operations (tasks or jobs) under a certain set of constraints that must be satisfied. A scheduler usually attempts to maximize the utilization of individuals and/or resources and minimize the time required to complete the entire process being scheduled. There exist a number of different types of scheduling problems, such as job shop problems, sport leagues games scheduling, timetabling, service timetable problem for transportation networks, etc. Many scheduling problems share some features with the timetabling problem. In (Schaerf, 1995) it is presented a survey of automated timetabling. The formulation of the university course timetabling problem (as given in (de Werra, 1985) and (Schaerf, 1995)) is the following:

Input data:

q *courses* K_1, ..., K_q, and for each *i*, course K_i consists of k_i lectures

r *curricula* S_1, ..., S_r, which are groups of courses that have common students

p - the number of periods

l_k - the maximum number of lectures that can be scheduled at period *k* (i.e. the number of rooms available at period *k*)

Goal:

$$find \quad y_{ik} \qquad (i = 1,...,q; k = 1,...,p)$$

[1] $$s.t. \quad \sum_{k=1}^{p} y_{ik} = k_i \qquad (i = 1,...,q)$$

[2] $$\sum_{i=1}^{q} y_{ik} \le l_k \qquad (k = 1,...,p)$$

[3] $$\sum_{i \in S_l} y_{ik} \le 1 \qquad (l = 1,...,r; k = 1,...,p)$$

[4] $$y_{ik} = 0 \ \ or \ \ 1 \quad (i = 1,...,q; k = 1,...,p)$$

The constraints are: each course is composed by the correct number of lectures (eq. [1]); each time there aren't more lectures than rooms (eq. [2]); avoid conflicting lectures to be scheduled at the same period (eq. [3]).

The objective function:

$\max \sum_{i=1}^{q} \sum_{k=1}^{p} d_{ik} y_{ik}$, where d_{ik} is the desiderability of having a lecture of course K_i at period *k*.

Several solutions (manual or automated) were proposed in the literature. Some automated solutions are given by tabu search (Costa, 1994), constraint satisfaction (Schaerf, 1995), genetic algorithms (Corne *et al,* 1994), logic programming (Fahrion and Dollanski, 1992), and combination of different methods (Picard *et al*, 2004).

3 The architecture of MAS_UP-UCT system

We have designed the architecture of a multi-agent system, MAS_UP-UCT, that tries to solve optimally the university courses timetable scheduling. In Figure 1 it is shown the architecture of the multi-agent scheduling system. We briefly describe how it is usually made the manual university course timetabling. Suppose the university includes five faculties, each of them having a number of specializations. The timetabling for each specialization is done by a person who is dedicated to this job, which we shall name *specialization course scheduler*. This person will provide five, four or three timetablings corresponding to the specialization's number of study years. The specialization course scheduler will receive a list of options from each professor that is teaching a course to a certain year of study at that specialization. The list of options will include the professor's options ordered by their desirability, and will include also, the list of impossible timetable schedulings. After course timetable scheduling is done at every faculty, it is started the activity of rooms allocation at university level. So, the university course timetable scheduling problem is divided in two subproblems: 1) faculty course timetable scheduling (which involves only allocation of course day and time), and 2) university course rooms allocation (which involves allocation of rooms for courses). When all courses have allocated time intervals (day and time) and rooms, the university course timetable scheduling is ended with success. Whenever a problem occur, it is started a communication process which will involve mainly a negotiation activity.

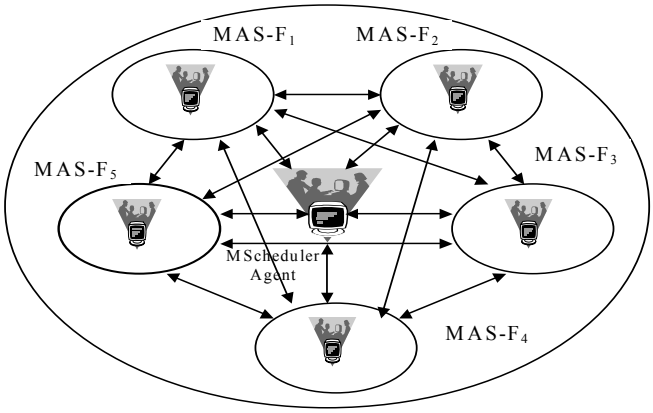


Figure 1. The architecture of MAS_UP-UCT

In most Romanian universities, the university course timetable scheduling is done either manually or partial automatically. In order to improve the efficience of the whole activity, we have mapped the course timetabling in terms of autonomous intelligent agents. Each faculty has a scheduler multi-agent system (MAS-F_i), which has to

schedule the courses of that faculty. The main scheduler agent (the university scheduler agent) which will allocate the rooms is *MScheduler Agent*. Because most professors teach courses to different faculties, every faculty scheduler agent has to communicate with the others scheduler agents, in order to solve some critical situations that may arise. The negotiation strategy used by the agents is similar to that described in (Oprea, 2003). In Figure 2 it is presented the MAS at faculty level, which includes a faculty scheduler agent, and expert assistants for each specialization of that faculty.

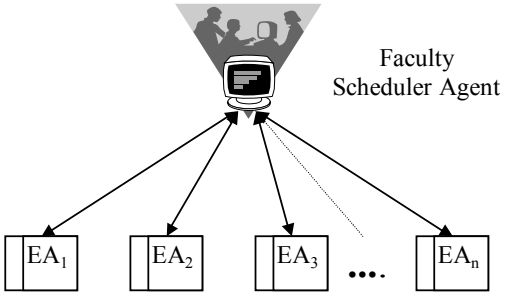


Figure 2. MAS-at faculty level

For each specialization it is developed an expert assistant which has to do all the activities connected with that specialization (e.g. evidence of students, course curricula etc). An important activity that should be done by an expert assistant is course timetable scheduling (day and time). A lot of constraints should be satisfied in order to solve the course timetabling. For example, one constraint is that all courses of a specialization are teached for all the groups of that specialization, and this constraint may become more severe in the case of courses that are teached for more than one specialization (this case appear for specializations that have courses with the same curricula). The faculty scheduler agents, who act autonomously, can schedule university course timetable on professor's individual behalf. Ideally, all profesor's preferences should be accepted. Unfortunately, we cannot reach an agreement among agents taking in consideration all professor's preferences. In course timetable scheduling, agents must quantify the professor's subjective preferences. In the worst cases (when a classical negotiation will have no results), we can reach a collective agreement by using a persuasion protocol (similar to that presented in (Ito and Shintani, 1997)). The persuasion protocol is based on the rationality of agents. Agents should satisfy some criteria of rationality (e.g. maintaining logical concistency). The advantage is that negotiation using persuasion protocol can reach more agreements compared with existing negotiation protocols and it can improve the rate of agreement in course timetable scheduling.

We make a brief discussion of two critical situations that may arise during a course timetabling:

1) - *at faculty course timetabling*: *day* and *time* timetable conflict (two or more professor's options are identical) - Solution: start a negotiation process between the expert assistant of that specialization and the professors involved (or their personal agents). A message is sent by the specialization expert assistant to all those professors

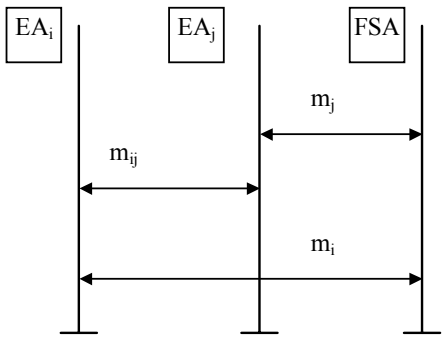
that are involved in a conflict, and will wait for a solution of the negotiation. If it will receive an answer it will do a rescheduling. If it will receive no solution, it will start a persuasion process of negotiation, suggesting a solution.

2) - *at university course timetabling*: - no room is available for a certain day and time course. In this case the MScheduler agent will start a negotiation process between faculty scheduler agents that are involved in the conflict, by given some options. Each faculty scheduler involved in the conflict will pass the message to the corresponding expert assistants, or, in some cases will continue to pass the message to professor's personal agents, who will than negotiate directly. If after this negotiation no solution will be find out (e.g. some courses cannot be moved in other module or day), the main scheduler agent will start a persuasion dialog between the faculties agents that are in conflict, which in turn will transfer the problem at the lower level.

4 Evaluation of the multi-agent system

As an evaluation method of our MAS we have chosen the interaction diagram method (Ronnquist and Low, 1997). An interaction diagram is a graph showing the processing of each agent symbollically as one or more vertical bars, and the messaging between agents as horizontal or oblique arrows between agents (from sender to receiver), decorated with message indications. In Figure 3 it is presented an example of interaction diagram, which illustrates a negotiation process at faculty level, between two expert assistants (EA_i and EA_j). In Figure 4 it is shown the interaction diagram in the case of a critical situation.

In order to evaluate the multi-agent system we can use interaction diagrams to design the communication process between agents (expert assistants, personal agents etc) and to verify that the system executes the correct communication sequences. We have used message flow fragmentation in order to realize an analysis of the communication process.



$EA_{i,j}$ - Expert Assistant for specialization i, j
FSA - Faculty Scheduler Agent

Figure 3. Example of interaction diagram

The direct sequence and a part of the inverse sequence of the message flow fragmentation that corresponds to the negotiation process shown in Figure 4 is given bellow.

<beg(MAS), beg(FSA_l), beg(FSA_k), beg($EA_l t$), beg($EA_l r$), beg($EA_k i$), beg($EA_k j$), snd(MAS, m_1), split(m_1, m_{1k}, m_{1l}), rcv(FSA_k, $m_1 k$), rcv(FSA_l, $m_1 l$), split($m_1 k$, $m_k i$, $m_k j$), split($m_1 l$, $m_l t$, $m_l r$), rcv($EA_k i$, $m_k i$), rcv($EA_k j$, $m_k j$), rcv($EA_l t$, $m_l t$), rcv($EA_l r$, $m_l r$), snd($EA_k i$, $m_k i^{-1}$), snd($EA_k j$, $m_k j^{-1}$), join($m_k i^{-1}$, $m_k j^{-1}$, $m_1 k^{-1}$), ..., end(MAS), end(FSA_l), end(FSA_k), end($EA_l t$), end($EA_l r$), end($EA_k i$), end($EA_k j$)>

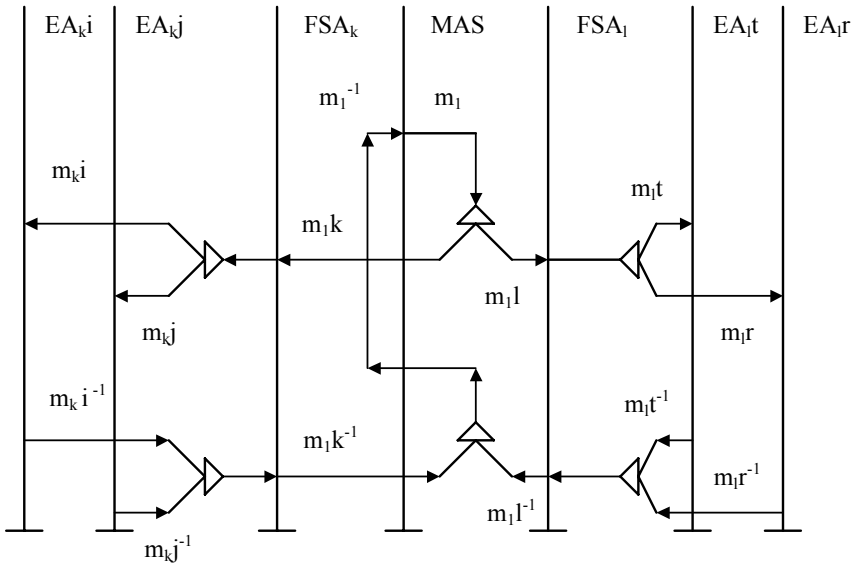


Figure 4. An example of negotiation in a critical situation

The inter-agent communication is done by using the agent language FIPA ACL. Figure 5 shows an example of such a message $m_{\text{A12 – EA-IME-AC}}$ exchanged during a negotiation.

```
(inform
    :sender A12
    :receiver EAIME-AC
    :in-reply-to message3
    :content ((conflict  Monday  M2  IME-AC2  (course  POO))
              (options   Monday  (M3 M5 M7)
                         Tuesday (M1 M2)
                         Wednesday (M1-M5)
              ))
    :protocol fipa-iterated-contract-net-protocol
    :ontology univ_timetabling
    :language sl
)
```

Figure 5. Example of a FIPA ACL message

5 Conclusion

The paper presented the current state of a research work that involve the development of a multi-agent system for university course timetable scheduling. The purpose of our work was to analyse the benefits of using an agent-based approach for the university course timetable scheduling, which involves a lot of communication, cooperation and negotiation processes. We have described the architecture of a multi-agent system for university course timetable scheduling, MAS_UP-UCT, and briefly discussed about the evaluation of the multi-agent system.

We can conclude that the main benefits of the agent-based approach adopted for university course timetabling are given by the possibility of doing negotiation between agents as a solution to the conflicts that may arise, and by the analysis of the exchanged messages flow between agents with the interaction diagrams.

References

[8] Corne, D., Ross, P., and Fang, H.-L (1994): Fast practical evolutionary timetabling, *Lecture Notes in Computer Science*, LNCS *865*, 251-263.

[9] Costa, D (1994): A tabu search algorithm for computing an operational timetable, *European Journal of Operational Research*, *76*, 98-110.

[10] de Werra, D. (1985): An Introduction to Timetabling. *European Journal of Operational Research*, *19*, 151-162.

[11] Fahrion, R. and Dollanski, G. (1992): Construction of university faculty timetables using logic programming techniques, *Discrete Applied Mathematics*, *35, 3*, 221-236.

[12] Ito, T., and Shintani, T. (1997): An Agenda-scheduling System Based on Persuasion Among Agents. Technical report: Nagoya Institute of Technology.

[13] Oprea, M. (2003): The Use of Adaptive Negotiation by a Shopping Agent in Agent-Mediated Electronic Commerce, *Lecture Notes in Artificial Intelligence, LNAI 2691*, Springer-Verlag, Berlin Heidelberg, 594-605.

[14] Picard, G., Bernon, C. and Gleizes M-P. (2004): Cooperating Agent Model within ADELFE Framework An Application to a Timetabling Problem. In *Proceedings of The 3rd International Joint Conference on Autonomous Agents & Multi Agent Systems*, New York, USA, 1506-1507.

[15] Ronnquist, R., and Low C.K. (1997): Analysing Expert Assistants through Interaction Diagrams. In *Proceedings of Autonomous Agents 97*, ACM Press, 500-501.

[16] Schaerf, A. (1995): A survey of automated timetabling. Technical report: CS-R9567, Centrum voor Wiskunde en Informatica.

[17] Weiss, G. (1999): *Multiagent systems*, The MIT Press, Cambridge, Massachusetts.

[18] Wooldridge, M., and Jennings, N.R. (1995): Intelligent agents: theory and practice. *The Knowledge Engineering Review*, *10*, *2*, 115-152.