A system model for university course timetabling using mobile agents

Yan Yang\textsuperscript{a,b}, Raman Paranjape\textsuperscript{a,b,*}, Luigi Benedicenti\textsuperscript{a,b} and Nancy Reed\textsuperscript{c}
\textsuperscript{a}Electronic and Software Systems Engineering, Faculty of Engineering, University of Regina, Regina, SK, S4S 0A2, Canada
\textsuperscript{b}TRLabs Regina, University of Regina, Regina, SK, S4S 0A2, Canada
\textsuperscript{c}Department of Information and Computer Sciences University of Hawai’i at Manoa, Honolulu, HI 96822, USA

Received 1 June 2005
Accepted 12 April 2006

Abstract. A mobile multi-agent system is proposed to create solutions for the university course timetabling problem. It is composed of four kinds of agents: (mobile) Course Agents, and (stationary) Signboard, Publisher and Interface Agents. The key strength of this new approach is to use a fundamental attribute of Agents that of autonomy. This autonomy is manifested in this work in the Course Agent. Each Course Agent in the system is responsible for negotiating with other Course Agents to find satisfactory class resource for the course they represent. This negotiation occurs initially indirectly through a Signboard Agent. A set of rules is used to structure Agent-to-Agent negotiation to find mutually acceptable class resources. The scheduling problem is executed in a natural parallel structure using one Signboard Agent to represent a weekday. The experimental results show that this new approach has merit and can lead to acceptable and flexible solutions to the course timetabling problem.

Keywords: Multi-agent system, course timetabling, system model

1. Introduction

The course timetabling problem has as its goal to find an appropriate timetable for a set of courses to be scheduled within limited resources such as classroom and class time. This timetable must be found while simultaneously satisfying other constraints such as efficient use of resources, convenience to students and instructors, etc. There are generally two types of constraints in timetabling: hard and soft constraints. Hard constraints are those that must be satisfied and cannot be violated, any violation will mean an infeasible timetable. For example, a professor can’t give two lectures at the same time in two different classrooms. Soft constraints are those that are preferably satisfied, but may be relaxed if necessary in order to meet hard constraints. Violations of soft constraints may be unavoidable as they can be mutually conflicting. However, in general, failure to meet a number of soft constraints will lead to a poor quality timetable. The course timetabling problem fundamentally involves properly combining courses, classrooms, time periods and instructors.

There are a number of interesting approaches suggested for solving the course timetabling problem. Hertz and Robert [2] proposed decomposing the problem into a series of easier sub-problems. Genetic algorithms [19] and Tabu Searches [16] have been effectively used in this problem also. Simulated annealing is another technique used by Elmohamed et al. [10] and Co-evolutionary algorithms were proposed by Chan et al. [4].
Agent technology [7,12] has also been applied to solve specific scheduling problems. Xie et al. [18] solved a network management problem by scheduling the necessary resources using agents. Annealing algorithms with multiple agents were used for the dynamic job shop scheduling problem [13]. Multi-feature action selection is done with negotiating agents by Scerri [14]. The classic problem of meeting scheduling has been addressed using agent technology by Chun et al. [1]. While multi-agent systems show real promise, their application to university course timetabling has as yet not been fully considered.

Only a few researchers [5,9,11] have attempted to apply the agent technology to solve course timetabling problems. Ho et al. [11] propose a multi-agent based approach to perform a high school timetabling problem. The whole system is composed of two types of agents: Teacher Agents and Class Coordinator Agents. Through an automatic negotiation between Class Coordinator Agents and Teacher Agents, an acceptable course schedule is produced. Gaspero et al. [9] apply a multi-agent architecture to a university course timetabling problem. Three kinds of agent are defined: Solver Agent, Negotiator Agent, and Manager Agent. The Solver Agent generates a solution to the timetabling problem for a department. The Negotiator Agent communicates with other Negotiator Agents and they exchange resources with each other. The Manager Agent stores and updates information necessary for the Negotiator and Solver agents to make their decision about the schedule. This work presents a system design but does not present any course scheduling results. Kaplansky and Meisels [5] also propose a multi-agent system to solve the university course timetabling problem. They focus on the inter-agent negotiation procedures. In their model, they construct the university course timetabling as a constraint satisfaction problem and there are two kinds of agents: Scheduling Agents and Room Agents. The lectures are the variables, the time slots form the domain values for the Scheduling Agents and the required classrooms form the domain values for Room Agents.

Kaplansky and Meisels [5] also propose a multi-agent system to solve the university course timetabling problem. They focus on the inter-agent negotiation procedures. In their model, they construct the university course timetabling as a constraint satisfaction problem and there are two kinds of agents: Scheduling Agents and Room Agents. The lectures are the variables, the time slots form the domain values for the Scheduling Agents and the required classrooms form the domain values for Room Agents.

It is expected that significant benefit may be obtained if the specific and unique characteristics of software agents can be harnessed in addressing the course timetabling problem. These unique characteristics of Agents including: autonomy, social ability, proactiveness and reactivity with mobility (in the case of mobile agents).

At the application level, it is believed that the proposed approach will be superior to existing approaches because of its flexibility with respect to hard and soft constraints. It is anticipated that the approach would permit highly individual constraints for instructors, students and the actual courses. In addition, it is expected that this type of system may be more able to adjust to last minute changes in requirements for the schedule. This is because only the courses (and hence their agents) that are affected by the change will need to adjust in the event of, for example, an added or deleted course. In some sense the system dynamically partitions the global timetabling problem into a number of smaller sub-problems that can be independently solved.

These application level benefits are currently only hypothesized, as the necessary comparative studies have not yet been conducted.

In this paper, a mobile multi-agent system based on the TEEMA (TRLabs Execution Environment for Mobile Agents) platform [3,15,17] is used. TEEMA has been developed jointly by TRLabs Regina and the University of Regina, and it is a relatively standard Agent Execution Environment (AEE) or agent platform. TEEMA provides addressing and naming, messaging, mobility, security, logging, and a procedure for the addition of services like many AEE. TEEMA and all the agents on this work are programmed with Java. While TEEMA was used in this work because of its familiarity to the authors, no unique or special features are required of the AEE, and any other AEE with these same basic functions could be used in this work.

### 2. System design

This section presents a detailed look at the system design of the multi-agent approach to solving the course timetabling problem. A summary overview of the operation of the multi-agent timetabling process is first presented. Each component of the multi-agent system is then discussed in detail.

#### 2.1. System architecture overview

In this work, the key structures are Course Agents. These Course Agents operate on five TEEMA platforms, which represent five school days of the week (weekday platforms).

On each weekday platform, there is a ‘Signboard Agent’, which acts as a communication mechanism to record course information, such as the course title and classroom. A Course Agent is initially started on the platform corresponding to the weekday, which is the
Course Agent’s first priority. The Course Agents negotiate with each other through the coordination of a Signboard Agent to find a mutually acceptable time slots. If an acceptable time slot is not available on a particular weekday platform the Course agent will eventually move to a new weekday platform and repeat the negotiation process. Various checks are used by the Signboard Agent to limit the possibility of an infinite loop. After the Course agent negotiation is complete a new and effective timetable has been generated. A Publisher Agent then collects the individual weekday schedules and creates the final output, which is a complete course timetable. The system architecture is shown in Fig. 1.

2.2. Weekday platform design

In the weekday platform (shown in Fig. 2), there are three kinds of agents: Interface, Signboard and Course Agents.

The Course Agents are mobile and can move between weekday platforms. There is only one Interface Agent and one Signboard Agent on each weekday platform. These agents are stationary. In the figure, the solid line with bi-directional arrows between agents represents a communication channel. The dotted arc with uni-directional arrows represents migration of a mobile Course Agent to a new weekday platform. The small rectangle represents other weekday platforms to which the Course Agent can move.

Through the Interface Agent, course information is entered and a Course Agent is subsequently created. During course timetabling, all the Course Agents communicate their preferred class resources (class time and classroom location) to the Signboard Agent. If these resources are available, they are assigned to the requesting Course Agent. If they are not available, the Course Agent chooses a secondary class resources allocation and requests the Signboard Agent again if this new class resource allocation is acceptable. If such class resources are still unavailable, the Course Agent begins to negotiate with other Course Agents to see whether it can get its preferred class resources by exchanging preferences with these other Course Agents. If all of these negotiations fail, the Course Agent will move to another weekday platform (as indicated by the dotted arc with arrow and the small rectangle shown in Fig. 2) to look for its preferred class resources on another platform. Another situation in which a Course Agent will migrate occurs when there are no available class resources on its current weekday platform. In order to perform the functions of class resource selection, negotiation and migration, each Course Agent has a set of timetabling rules, namely, “Class Resource Selection Rules”, “Negotiation Rules” and “Migration Rules”. These rules are presented in Section 3.

The basic function of the Signboard Agent is to check for potential conflicts among Course Agents’ resource requests and record the course information for successful requests in order to create an effective weekday course schedule. After the weekday timetabling pro-
cess is over, the Signboard Agent contains the schedule of all courses for its corresponding weekday. In this way, a weekday course timetable is produced through the negotiation and cooperation of all the involved agents on that weekday platform.

2.3. Agent module design

As shown in the system design diagrams (Figs 1 and 2), there are four kinds of agents in the timetabling system: Interface Agents, Course Agents, Signboard Agents and a Publisher Agent. In this section, these four agent modules are described in greater details.

2.3.1. Interface Agent

The Interface Agent is a stationary agent and provides a GUI interface (as shown in Fig. 3) to input course information such as the course title, class time, classroom, instructor’s name and individual preference.
The course information can be entered either through the GUI components one by one or from a data file. After the necessary information for a course is obtained, the Interface Agent starts up a Course Agent and transmits the entered information to the newly created Course Agent. All the other Course Agents in the system are created in the same way.

2.3.2. Course Agents

The Course Agents are mobile agents and they carry specific course information with them. Among the course information, individual preferences related to each course instructor are optional. The default preference is none. Preference may include teaching times, teaching locations, or teaching duration etc. The Course Agents negotiate with one another in order to find a non-conflicting course timetable. Using mobile agent attributes of mobility and autonomy, and by the application of a set of timetabling rules, the Course Agents respond to changes in their environment to adjust their expectations of class resources such as class time and classroom location.

The main functions of course agents

– Request class time-room resources from the Signboard Agent according to the course preferences.
– Decide autonomously whether to change the class resources, negotiate with other Course Agents or move to another weekday platform.
– Inform the Signboard Agent of its decisions so that the recorded course information on the signboard can be updated promptly.
– Change the class resource randomly without repetition according to the “Class Resource Selection Rules”. (Without repetition means if some class resource has to be changed, the new resource will be any one in the available resource list other than the current or previously chosen one(s). In this way, the conflict possibilities are reduced and the system is likely to become stable more quickly.)
– Negotiate with other Course Agents to obtain the preferred class resources according to the “Negotiation Rules”.
– Move to another weekday platform to look for the preferred class resources according to the “Migration rules”.

Class resource selection rules

– Choose the class time period randomly without repetition according to the time preference when a class time conflict occurs.
– Choose the classroom randomly without repetition when the requested classroom is not available.
– When the preferred class resource is not available, negotiating with other Course Agents and try to exchange class resources according to the “Negotiation Rules”.
– When the preferred class resource cannot be found on any weekday platform, accept any available class resource without considering preferences. In such a case, the course is assigned an undesired class resource.

Negotiation rules

– Negotiate only with those Course Agents who have the preferred class resources that the individual Course Agent wants.
– If one Course Agent has no preference or is assigned an undesired class resource, it will agree to exchange the resource. Otherwise the Course Agent will refuse to exchange class resources when receiving a request from another Course Agent.
– When a course resource request is refused, the Course Agent chooses randomly without repetition other available Course Agents with whom to negotiate acceptable course resource allocations. Class size constraints are met.
– When all negotiations fail or there is no Course Agent with whom to negotiate, move to another weekday platform.

Migration rules

– When there is no any available class resource left on the current weekday platform, move to another weekday platform to continue searching for the class resources.
– When no preferred class resource can be obtained even after negotiation with other Course Agents, move to another weekday platform to continue searching for the class resources.
– There are at most two moving rounds for each Course Agent to move through all the weekday platforms. The first move round is to search for the preferred class resource and the second one is to look for any available class resource when no preferred class resource can be found after the first move round.
2.3.3. Signboard Agent

The Signboard Agent is a stationary agent. Just as its name implies, it plays the role of a signboard. It compares requests from Course Agents against hard constraint requirements in the timetable and requests from other Course Agents. It advises Course Agents of course conflicts and hard constraint issues. So in some sense, it acts as a coordinator by identifying conflicting Course Agents as well. The main functions of Signboard Agent are listed below:

- Carry a class time-room resource table and assign available class resources requested by Course Agents.
- Check the potential conflicts among Course Agents when some Course Agent requests class resource.
- Record the course information if a course succeeds in scheduling on the current weekday platform (Signboard Agent).
- Update the course information recorded on the virtual signboard when informed some changes by Course Agents.
- After a weekday course timetabling process completes, send the daily course schedule of the corresponding weekday to the Publisher Agent.

2.3.4. Publisher Agent

The Publisher Agent is a stationary agent whose main functions are as follows:

Fig. 4. A partial weekly timetable.
Y. Yang et al. / A system model for university course timetabling using mobile agents

Table 1
Comparison of distributed and centralize architecture

<table>
<thead>
<tr>
<th></th>
<th>Distributed Architecture</th>
<th>Centralize Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheduling Duration</td>
<td>36(s) 719(ms)</td>
<td>33(min) 58(s) 719(ms)</td>
</tr>
<tr>
<td>Number of Messages</td>
<td>2136</td>
<td>6962</td>
</tr>
</tbody>
</table>

Table 2
Effect of decreasing classrooms and increasing timeslots

<table>
<thead>
<tr>
<th></th>
<th>Distributed Architecture</th>
<th>Centralize Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheduling Duration</td>
<td>45(s) 78(ms)</td>
<td>16(min) 25(s) 750(ms)</td>
</tr>
<tr>
<td>Number of Messages</td>
<td>2140</td>
<td>3058</td>
</tr>
</tbody>
</table>

Table 3
Effect of increasing the preference ratio

<table>
<thead>
<tr>
<th>Scenario No.</th>
<th>Course No.</th>
<th>Professor No.</th>
<th>Preference Ratio (%)</th>
<th>Message No.</th>
<th>Scheduling Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>100</td>
<td>0</td>
<td>754</td>
<td>4(s) 328(ms)</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>100</td>
<td>50</td>
<td>764</td>
<td>4(s) 344(ms)</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>766</td>
<td>4(s) 562(ms)</td>
</tr>
</tbody>
</table>

– Collecting the weekday timetables from Signboard Agents.
– Sorting the course schedules according to their time sequence.
– Saving the sorted schedules into a formatted text file to produce a weekly course timetable.

A typical example of the output of the Publisher Agent is presented in Fig. 4.

3. Experimental results

In this section a set of results are presented for various timetabling scenarios. In the first experimental scenario the following conditions are used: 100 courses, 100 professors, 5 days of 9 time slots each and 10 classrooms. Two versions of the system are presented, a distributed architecture with 5 parallel platforms running on the same computer and a centralized architecture with all courses in one large platform and no break out of individual weekdays. In the distributed architecture there are 90 time slots on each weekday platform (450 total) while in the centralized form there are 9 time slots and 50 classrooms creating 450 total time slots. In these no individual instructor preference for time of day for classes is considered. The results are presented in Table 1.

In the second experimental scenario the same conditions are used as in the first for the distributed architecture however for the centralized architecture 45 time slots in one day and 10 classrooms are used. Thus the examination is made for the effect on the centralized architecture with more time slots versus more classrooms. The results are presented in Table 2.

Next the effect of increasing the complexity of the soft requirement of instructor time preference is examined. The percent of instructors who have a time preference is modified from 0% to 100%. This is to examine the effect on the processing time and number of messages when the soft requirements are made more difficult. These experiments are performed with the distributed five weekday platforms architecture. The results are presented in Table 3.

4. Discussion

An examination of the results in Table 1 shows that the distributed architecture has a huge advantage over the centralized architecture in terms of the processing time and in terms of the number of messages needed to create the schedule. It may be speculated that in the centralized system, a single signboard is not able to handle the number of requests put out by the agents creating a bottleneck in the system. As both the distributed and centralized architectures are executed on a single computer – both systems are given access to the same basic computing power. However there is in fact greater operating system overhead for having five weekday platforms on the same machine.

In Table 2 the centralized approach is modified to a slightly unrealistic model of having 450 time slots for the 10 classrooms. This is like having 45 time slots in one day for any one classroom. The modification reduces the computational time for the central control.
algorithm by almost 50%. Thus it is observed that the 100 courses can more easily find time slots in these conditions than in the condition of 50 classrooms. It can be also observed from this table that the distributed architecture operates in a very similar manner as in the first experiment, suggesting that the measured parameters are representative of the distributed architecture’s performance.

Table 3 begins to look at the effect of instructor preference on the system. Preferences of 0%, 50% and 100% are considered for morning or afternoon classes by the instructor. In each run however, it is found that there is virtually no difference in the amount of processing time or messaging with these changes in preference. This suggests that the timetabling system is not being adequately challenged. It can be identified that in these experiments only 100 courses are scheduled into 450 time slots. Thus it is reasonable to expect that this problem is still not adequately difficult to show the differences in behavior among the systems.

In this work, the multi-mobile agent course timetabling system presented is an example of the power and effectiveness of the distributed and cooperative model of intelligence [6,8], where each agent is responsible for its own component of the problem-solving process. It can be seen that although the individual agents have incomplete and insufficient information to solve the entire course timetabling problem, by working on their individual sub-problem in a system of agents they can together produce a complete timetable. Unlike a traditional centralized solution the multi-agent model offers the advantages of distributed and interactive problem solving and thus should have better scaling properties on large problems.

It may be observed that the proposed approach can not effectively resolve application-specific issues in university course timetabling. For example, when constraints in two different course timetabling problem instances are changed, the design for some agents (such as the Course Agents) may need corresponding modifications. Another potential problem is that when the number of agents in the system is increased dramatically, the messages communicated among agents will be increased accordingly and the communication costs will become high. As well, the management of the agents is not currently considered in the system. For instance, the tracking problem for mobile agents might be a challenge, especially when the number of mobile agents is large.

5. Conclusion and future work

The experimental results suggest that the mobile multi-agent system design can provide a feasible and flexible approach to performing course timetabling. The proposed mobile multi-agent system solves this problem in a distributed and dynamic way. A key strength of this approach is that it uses a fundamental attribute of Agents, that of autonomy. This autonomy is manifested in this work in the Course Agent. As the name suggests, the Course Agent represents all aspects of a course within the system that is to be timetabled. The Course Agent has attributes that may be described as hard and soft requirements with respect to the class. These attributes can include features such as class size (hard requirement), class location preference (soft requirement), and class time slot (soft requirement) etc. Each Course Agent in the system is responsible for negotiating with other Course Agents to find satisfactory time slots for the course they represent. This negotiation occurs initially indirectly through a Signboard Agent. Thus the system uses other important features of agent systems including proactivity, reactivity, and communicativity.

This Agent system uses negotiation among agents (market principle) where each player in the system acts in its own self interest resulting in the system as a whole producing an optimal or close to optimal schedule. No evidence is presented here with respect to optimization, as these are some initial experiments. The next step in this work will be to compare existing mature course timetabling algorithms with the proposed multi-agent system. The testing process has also been begun for the system to examine its behavior under increased and more complex soft requirements. However, the results in this case indicate that the system must be under greater load in order for it to show its characteristic response to these types of challenges.

While it is not possible (at this point) to provide a clear justification, it is anticipated that the timetabling system which is being developed in this study will have natural applications (with some modifications) to other important timetabling problems such as those of airline schedules, airport docking bay schedules, and the movement of goods and services by logistic and service companies.

References


Authors’ Bios

Yan Yang is a Ph.D. candidate of Electronic System Engineering in the Faculty of Engineering at University of Regina. His main research interests include: multi-agent system; course timetabling problem; and combinatorial optimization problem.

Dr. Raman Paranjape is a professor of Electronic System Engineering in the Faculty of Engineering at University of Regina. His main research interests include: multi-agent agent systems (e-commerce, medical data retrieval, graphical representation of agent societies); sonics and ultra-sonics for underground and non-destructive visualization and testing; fibre optic sensors for oil and liquid sensing; robotic sensor systems (vision, touch, sonar); biomedical signal and image processing; fundamentals of signal and image processing.

Dr. Luigi Benedicenti is an associate professor of Software System Engineering in the Faculty of Engineering at the University of Regina. His main research interests include: software engineering and software product lines; mobile agents and their application in emergent behaviour; augmented reality; new media technology and scientific visualization; and distributed computing.

Dr. Nancy Reed is an assistant professor of Department of Electrical Engineering at University of Hawaii. Her main research interests are in the areas of artificial intelligence, autonomous agents, biomedical informatics; cognitive modeling; knowledge-based/expert systems; knowledge acquisition; data mining; machine learning; and real-time systems. Current and recent research activities include developing autonomous agents for simulation environments; developing computational models of diagnostic problem-solving methods; the collection and analysis of verbal protocols from experts; and modeling human expert reasoning.