Distributed Examination Timetabling

Eliezer Kaplansky[1], Graham Kendall[2], Amnon Meisels[1], and Naimah Hussin[2]

[1] Department of Computer Science
Ben-Gurion University of the Negev
Beer-Sheva, Israel
{kaplan_e,am}@cs.bgu.ac.il
[2] School of Computer Science & IT
The University of Nottingham
Nottingham, UK
{gxk,nbh}@cs.nott.ac.uk

1 Introduction

Most commercial and academic organizations are currently constructing their timetables by some central authority. However, many of those organizations are, in fact, composed of several departments that need to timetable their activities in an independent way. If such organizations allow each department to manage its own timetable, then the timetables of all the departments must be combined to yield a coherent, consistent solution. Multi-Agent System (MAS) technology is the concurrent paradigm that can be used to achieve the goal of (globally) compatible schedules. Agents are sophisticated computer programs that act autonomously on behalf of their users, across distributed environments. A multi-agent system is a loosely coupled network of software agents that interact to solve a global problem that is beyond the individual capacities or knowledge of each agent. In order to satisfy the organizational global constraints, negotiations among agents and methods for changing their local solutions are needed.

A well-defined example of a real-world *Timetabling Problem* (TTP) is the construction of university examination timetables [2, 3, 7]. In this paper we consider the examination timetabling problem of University Technology MARA. This problem has been first introduced by Cowling, Kendall and Mohd Hussin [4]. University Technology MARA (UiTM) is the largest university in Malaysia with a total number of students approaching 100,000. The university has 13 branch campuses, one in every state in Malaysia with 144 programs offered by 18 faculties. All the students, in all the 13 campuses, that are taking the same course, have to sit the exam at the same time. Currently, the examination timetable is solved manually by a central office utilizing the previous semester's timetable and manually updating required changes to scheduled exams. The timetable is drafted very early in the semester and the scheduler updates the timetable on the basis of feedback from the campuses and faculties.

Malaysia has quite a number of public holidays that are **not** shared between states. In addition, different states have **different** weekend days. This enforces UiTM to consider, apart from the constraints that are common for the basic examination timetabling problem, additional constraints that arise from the differences in holidays and weekends among the different states. For example, if an exam falls on a state public holiday for some state and there are students from that state sitting the exam, then the exam must be moved to another slot. The density of the exams along the examination period is, anyway, too high, and therefore it is not possible to just exclude all public holidays.

An important objective of UiTM's examination timetabling is to avoid weekend exams. Different states have different weekend days, which makes this objective different for different states. Currently, the solution produced by the manual schedulers, include exams that are to be schedule during weekends. This is **highly undesirable**. More details on the UiTM's examination problem and its formulation can be found in [5].

The schedules of the individual campuses are constrained by additional local requirements. Two families of local constraints are involved. Each campus has its own classroom capacity during the examination periods. In addition, campuses have to supply invigilators for their exams, using their pool of teachers. Naturally, each team of teachers, in the different campuses, has its own local time constraints.

The need to satisfy constraints that are local to each campus generate a *Distributed Timetabling Problem* (DisTTP) (formulation has been introduced by Kaplansy and Meisels [6]) . By using the distributed mode, local data and local constraints can be utilized to achieve a better balance between global consistency and the quality of the local solution.

2 Distributed examination timetabling

From the above description of the UiTM problem and the existing need for an automated solution, one can proceed in two ways. One can solve the examination timetabling problem for all the campuses by a single, central, process. In this way the central process attempts to improve and automate the existing manual solution by optimizing a global objective function. The optimization process of the global program is expected to produce a better global cost, but, may not be as good in satisfying specific local objectives for the different campuses.

Alternatively, one can solve the problem by using a multi-agent system paradigm. In this model each campus is represented by an agent. The set of negotiating agents can represent more closely the different objectives of the different campuses. Furthermore, there are several reasons why a global (central) search procedure is not suitable. These relate to features like private information, dynamic evaluation function, local preferences and more (see section 4).

The process of a distributed search, for a global solution, starts by agents computing the best local schedule for their campus. Next, agents negotiate with other agents (campuses) for a feasible global solution. This has the potential of achieving better schedules for particular campuses. The agents that generate the schedules for the different campuses are termed *Scheduling Agents (SAs)* and a timetabling problem that is modeled as a MAS is termed a *distributed timetabling problem* [6]. By its geographical structure and different cultures involved, the UiTM defines an interesting case of a DisTTP.

One of the major differences between central and distributed timetabling is that each agent is trying to produce a timetable that is suitable for a client (campus) which it represents. In addition, each agent may have a different set of requirements (cost function) from other agents. So although, all the agents are cooperating in order to produce a timetable, which is globally feasible, they are also trying to optimize their own local timetable. of requirements (cost function) to that of the other agents. In the context of examination timetabling one agent (which could be representing a school, department, campus etc.) may require that the students, which the agent represents, are not allowed to have examinations in consecutive time periods (i.e. a hard constraint), whereas another agent may only consider this as a soft constraint and just penalize its occurrence.

This paper presents an experimental MAS model that can be used to improve the quality of examination timetabling of University Technology MARA. We compare the central approach to our model. To perform this comparative study, two different solving mechanisms have been implemented. One solver utilizes the Scheduling Agents (SAs) system [1, 6]. The other solver uses a centralized Hyper-heuristic Tabu search [5]. Before comparing the quality of the solution achieved, one has to note that each agent is self-interested, meaning that the final solution may be the best for the agents involved, but not for the the university as a whole.

While there are natural benefits to the DisTTP approach there are several considerations which must be addressed before implementing such an approach. The minor consideration is the additional computation effort that is required to enable each campus to define different quality goals and to allow for the exchange messages between agents. The major consideration is the additional hardware and software resources that are needed in order to implement the distributed solution. However, the worldwide trend towards distributed solutions is a significant force that may eventually push UiTM to implement a MAS system.

The focus of this paper is on the negotiation procedures that can detect and avoid the occurrence of conflicts between Scheduling Agents that shared common exams.

3 Negotiation between the Scheduling Agents

The distributed method for solving the UiTM examination timetabling problem uses a multi stage inter-agent negotiation protocol. The proposed negotiation protocol consists of three main stages. The first stage focuses on the search for a local solution for the timetabling problem of each agent. In the second stage, the target is a stable global timetable that eliminates any conflicts between the schedules produced by the SAs. In the third stage the SAs negotiate with each other in order to optimize their local solution.

There are significant differences between the campuses of UiTM with regards to the number of students, exams, rooms and teachers involved. Therefore, the complexity of the local problem that each agent has to solve varies widely. In order to offset these differences correctly, the first step of the negotiation protocol, is the *Self Assessment* step. Each agent estimates the rate of finding solutions for its local problem, expressed by the *Average Computation Cost* (ACC) per solution found. ACC is measured by the average number of *constraint checks* per solution found. Each agent sends its ACC to all agents that are sharing its list of exams. For each examination, the agents sharing the exam select one of them as the *exam manager* (ExMan) for this exam. This is done in such a way that the more complex (slower) agent is the *Exam Manager agent* and the other agents are the *Exam Participant Agents*. In this way agents agree on the direction of the constraint between them. To avoid cycles, the direction of constraints over the whole network of SAs are forced to be a DAG.

At this point, each agent receives a number of *bid tokens* that it can use, in the *bid stage*, to buy changes in the global timetable. The number of tokens should represent the complexity of the local solution of the agent. An agent with a large and complex local problem should get more tokens than an agent that tries to solve smaller or less complex problem. There are several ways in which one can assign tokens to the SAs:

Manual: Let the system manager set these tokens as it sees fit. This is the most popular way.

Calculated Difficulty: Use a formula that takes into account the number of students, number of exams and rooms in each campus.

Experimental Difficulty: Use the *Average Computation Cost* (ACC) that was found in the *Self Assessment* procedure, as described above, as the measurement unit.

Dynamic: Defer the decision to a later stage, and then assign a different number of tokens to each agent for each bid separately. The procedure should assign more tokens to an agent that are farther away from their $AMinC$, at the time of the bid. In this way, the global solution has a better chance of moving towards a solution that is more balanced and fairer to each campus.

For each exam there are two different types of agents, an *Exam Manager* and an *Exam Participant* (ExPar). At any time, any one agent can be an ExMan, an ExPar or both but for different exams. This is similar to the *Initiator* and *Participant* relationship of the *Contract-Net Protocol* (CNP) of FIPA [8]. The ExMan requests proposals for the assignment of a time-slot to the examination, from the set of ExPars.

In the second step the ExMan selects the proposal it prefers, or none at all. Every ExMan sends the time slots of the exams it manages to the agents that participate in these exams on its outgoing links. In response, each *Exam Manager agent* sends back its computed current cost when using this time slot.

In the third step of the protocol, each agent tries to improve its local timetable by changing the time slot of one of its exams and re-solving its local problem. When an agent finds a better solution, it sends a *Request for Change* (RfC) to the *Exam Manager Agent.* The RfC message is accompanied by an *Expected Gain* (EG) and a list of suggested *Alternative Time Slots* (ATS) (part of its open time slots). When a *Exam Manager Agent* receives a *Request for Change*, it searches for the best solution using one of the slots of the ATS list accompanying the RfC. If the *Manager Additional Cost* is lower than the *Expected Gain* an *Approve Change Procedure* is started:

1. The exam manager sends a *Change Offer* to all agents optimizing participating in this exam.
2. All *Participating Change Costs* are collected.
3. If the sum of all *Change Costs* is lower than the *Expected Gain* then this *Request for Change* is approved. Otherwise, all involved agents move to the next step - bidding.

In the *Bidding Step* the *Initiator Improvement Agent* attempts to buy its RfC from the *Exam Manager Agent* using its *bidding tokens.* The *Exam Manager Agent*, tries to use these tokens to buy the RfC from all the other agents that participate in the same exam. If the bid succeeds, the RfC is approved.

4 Central vis. Distributed

It is quite complex to compare the centralized approach to DisTTP method. A direct comparison of the quality of the best solution achieved is not meaningful. The central timetable attempts to achieve a minimal centralized-cost, while the agents in the distributed timetabling problem try to minimize their local cost. In other words, the two approaches have different evaluation functions. One way around this problem is to let the central solver optimize an evaluation function that is composed of different terms for the different campuses. This is actually can be done in the UiTM examination timetabling problem, by summing up penalties for weekend examinations across all campuses. The result of the central solver optimization, for such a composite evaluation function, can be compared to the distributed solution by comparing the quality of the schedule of each agent separately. Clearly, the central optimal solution cannot be optimal for all campuses separately.

However, there are certain important features of the distributed paradigm that cannot be compared to the central approach. One such feature is the issue of privacy. In many cases, the different campuses (or departments) do not want to share their evaluation function considerations with other departments. Policies of respecting personal constraints of teachers that have to attend examinations may differ from one campus to another. Moreover, it is common to keep these policies private to the department or campus.

An important feature of distributed timetabling is its ability to deal with dynamic conditions. The user interface of the distributed system enables users in each campus to select their own objectives and their own weights for their selected objectives. When a scheduling agent negotiates its schedule with other agents, it must re-schedule in response to requests by constraining agents [6, 1]. This dynamic feature of scheduling agents is of practical importance. Imagine a manager in a campus who watches the timetabling procedure, through the user interface of its agent. Since that in the distributed model, in principle, it is the local manager that define the evaluation function for its campus this campus manager can react to the current examination timetable by slightly changing its agent evaluation function. This can give the corresponding agent more flexibility. On the other hand, adding a dynamic element to the evaluation function of a global search strategy, is difficult, if not impossible.

5 Advanced distributed model

The basic distributed model proposed in this paper, deals only with the family of *hard constraints* which are also known to the main campus. These are the constraints to minimize weekend and public holidays examination slots.

A more advanced model includes two additional sets of constraints:

1. The room capacity of each campus. If too many exams are scheduled for one day in a campus, there will not be enough rooms available for all students that need to take those exams. (During exams students are siting apart from each other, and therefore the room capacity is much smaller than in a regular class.)
2. The time constraints of teachers that are assigned as exam invigilators. Each invigilator will be able to input to the system the days they prefer and those where they are not available. This includes an option for accommodating personal preferences.

In the academic world of today, higher degree of students and staff satisfaction is a fundamental factor for the success of an institution. Implementing the advanced DisTTP paradigm presented in this paper will enable the University Technology MARA to offer its students and staff more satisfying examination timetables and to reduce the burden of responsibility on managers of the main campus.

References

[1] E. Kaplansky A. Meisels. Negotiation among scheduling agents for distributed timetabling. In *Submitted to the 5th International Conference on the Practice and Theory of Automated Timetabling PATAT'04, Pittsburgh, PA USA*, August 2004.

[2] M. W. Carter. A survey of practical applications of examination timetabling algorithms. *Operations Research Society of America*, 34, 2:104–118, March-April 1986.

[3] M. W. Carter and G. Laporte. Recent developments in practical examination timetabling 1996. In *The Practice and Theory of Automated Timetabling", E.K. Burke and P. Ross (eds)*, volume 3-21, pages 3–21, 1997.

[4] P. Cowling, G. Kendall, and N. Mohd Hussin. A survey and case study of practical examination timetabling problems. In *In Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling PATAT'02, Ghent, Belgium*, pages 258–261, July 2002.

[5] G. Kendall and N. Mohd Hussin. A tabu search hyper-heuristic approach to the examination timetabling problem at university technology mara. In *Submitted to the 5th International Conference on the Practice and Theory of Automated Timetabling PATAT'04, Pittsburgh, PA USA*, August 2004.

[6] A. Meisels and E. Kaplansky. Scheduling agents - distributed employee timetabling (detp). In *In Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling PATAT'02, Ghent, Belgium*, pages 166–80, July 2002.

[7] A. Schaerf. A survey of automated timetabling. *Artificial Intelligence Review*, 13(2):87 – 127, 1999.

[8] R. G. Smith. The contract net protocol: High level communication and control in a distributed problem solver. *IEEE Transaction on Computers*, 29, 12:1104–1113, Dec. 1980.