Available online at www.sciencedirect.com
ScienceDirect
Engineering Applications of Artificial Intelligence 20 (2007) 857–873
Engineering Applications of
ARTIFICIAL INTELLIGENCE
www.elsevier.com/locate/engappai

An agent-based approach to solve dynamic meeting scheduling problems with preferences

Ahlem BenHassine[a,*,1], Tu Bao Ho[b]

[a]*Computational Linguistics Group, Language Grid Project, Knowledge Creating Communication Research Center (NICT), 3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan*
[b]*Knowledge Creating Methodology Laboratory, School of knowledge Science, JAIST 1-1, Asahidai, Nomi-shi, Ishikawa 923-1292, Japan*

Received 5 April 2005; received in revised form 22 September 2006; accepted 17 October 2006
Available online 12 January 2007

Abstract

Multi-agent systems are widely used to address large-scale distributed combinatorial applications in the real world. One such application is meeting scheduling (MS), which is defined by a variety of features. The MS problem is naturally distributed and especially subject to many alterations. In addition, this problem is characterized by the presence of users' preferences that turn it into a search for an optimal rather than a feasible solution. However, in real-world applications users usually have conflicting preferences, which makes the solving process an NP-hard problem. Most research efforts in the literature, adopting agent-based technologies, tackle the MS problem as a static problem. They often share some common properties: allowing the relaxation of any user's time restriction, not dealing with achieving any level of consistency among meetings to enhance the efficiency of the solving process, not tackling the consequences of the dynamic environment, and especially not addressing the real difficulty of distributed systems which is the complexity of message passing operations.

In an attempt to facilitate and streamline the process of scheduling meetings in any organization, the main contribution of this work is a new scalable agent-based approach for any dynamic MS problem (that we called MSRAC, for Meeting Scheduling with Reinforcement of Arc Consistency). In this approach we authorize only the relaxation of users' preferences while maintaining arc-consistency on the problem. The underlying protocol can efficiently reach the optimal solution (satisfying some predefined optimality criteria) whenever possible, using only minimum localized asynchronous communications. This purpose is achieved with minimal message passing while trying to preserve at most the privacy of involved users. Detailed experimental results on randomly generated MS problems show that MSRAC is scalable and it leads to speed up over other approaches, especially for large problems with strong constraints.

Keywords: Dynamic valued constraint satisfaction problems; Meeting scheduling problems; Local consistency; Multi-agent system; Distributed problem solving

1. Introduction

Multi-agent systems are widely used to address many real-world combinatorial applications such as meeting scheduling (MS). This problem embodies a decision-making process affecting several users, in which it is necessary to decide *when* and *where* one or more meeting(s) should be scheduled. To satisfy real-world efficiency requirements, in this work we focused on two challenging characteristics: the distributed and dynamic nature of the problem. The MS problem is inherently distributed and hence cannot be solved by a centeralized approach; it is dynamic because users are frequently adding new meetings or removing scheduled ones from their calendar. This process often leads to a series of changes that must be continuously monitored.

The general task of solving an MS problem is normally time-consuming, iterative, and sometimes tedious, particularly when dealing with a dynamic environment. In other words, solving the MS problem involves finding a compromise between all the attendants' meeting

*Corresponding author. Tel.: +81 90 20 98 1458; fax: +81 774 98 6967.
E-mail addresses: ahlem@nict.go.jp (A.B. BenHassine), bao@jaist.ac.jp (T.B. Ho).

[1]This work was done when the first author was at the Japan Advanced Institute of Science and Technology.

0952-1976/$ - see front matter © 2006 Published by Elsevier Ltd.
doi:10.1016/j.engappai.2006.10.004

requirements[2] (i.e., date, time and duration) which are usually conflicting. Thus, this problem is subject to several restrictions, essentially related to the availability, calendars and preferences of each user. Automating MS is important mainly because it can lead to more efficient and satisfying schedules within organizations (Feldman, 1987).

Most significant research efforts in the literature adopt agent-based technology for the distributed and dynamic aspects of MS problems. Initial meeting schedule research is based on constraint satisfaction problem formalism (CSP) (Montanari, 1974). The problem is formalized as centeralized CSP (Abdennadher and Schlenker, 1999; Bakker et al., 1993). These works are essentially focused on over-constrained CSPs. However, among more recent typical agent-based approaches, some works focused on using distributed autonomous and independent agents to solve the MS problem while maximizing users' preferences (Garrido and Sycara, 1996). This work is based on the communication protocol proposed by Sycara and Liu (Sycara and Liu, 1994), where agents are capable of negotiating and relaxing their constraints in order to reach an agreement on a schedule with high join utility. Another work also based on the multi-agent system, was described in (Sen et al., 1997). This work focuses on the problem of how an application domain for intelligent surrogate agents can be analyzed, understood and represented such that the underlying agents can make appropriate adaptations to their environment, to carry out tasks on behalf of human users. The authors' prior work focused on agents adapting to environmental changes (Sen and Durfee, 1994), however Sen et al. directed their efforts towards the integration of user preferences (Sen et al., 1997). Three other multi-agent approaches to MS problems, using the Partial CSP formalism introduced by (Freuder and Wallace, 1992), were given in the literature. The first work (Luo et al., 2000) is a new approach for MS problems using fuzzy constraints. The second work (Tsuruta and Shintani, 2000) proposes the distributed valued constraint satisfaction problem (DisVCSP) formalism to model MS problem. This approach is used in our experimental evaluation. The third work, based on multi-agent systems and using fuzzy constraints to express users' preferences, was presented in (Franzin et al., 2004). This MS system was based on an existing system that includes hard constraints (Franzin et al., 2002). The authors proposed to integrate preferences to their system and focused on observing the behavior of this new system under several conditions (Franzin et al., 2004). Their main objective was to evaluate the relations among solution quality, efficiency and privacy.

Nevertheless, the majority of these works share the following properties:

(1) Dealing only with non-dynamic problems (among which (Abdennadher and Schlenker, 1999; Bakker et al., 1993; Tsuruta and Shintani, 2000; BenHassine et al., 2003; Franzin et al., 2004); BenHassine et al., 2004a,b).
(2) Allowing the relaxation of any user's preferences, even those related to non-availability of this user in order to arrive at consensus choices for a meeting's time. However in real-world applications this is not always permitted. For example, when the user is traveling on business, such a constraint would oblige the user to stop his/her travel to attend the meeting, and this is not always possible (amongst Sycara and Liu, 1994; Garrido and Sycara, 1996; Sen et al., 1997; Luo et al., 2000; Tsuruta and Shintani, 2000; Franzin et al., 2004).
(3) Not integrating the enforcement of local consistency in their solving process, in spite of the pre-eminent role of the filtering techniques in the efficiency of solving an NP-complete problem. Only the authors in (Franzin et al., 2002, 2004) deal with the use of some inferred knowledge to maintain coherence between meetings in order to steer the selection of the next proposal, while, none of the other works try to maintain any level of consistency during the negotiation process.
(4) Judging all the meetings of the whole system with the same level of importance (among others Garrido and Sycara, 1996; Luo et al., 2000; Franzin et al., 2004; Tsuruta and Shintani, 2000). In real life, this is not always true. Obviously, the great significance of a meeting depends especially, but not only, on the leader of the event, the number of participants, and the meeting's main subject. Especially in a dynamic environment, such discrimination may lead to conflicting meetings, and may also increase the number of meetings to reschedule.
(5) Not considering the high complexity of message passing operations in real distributed systems (Garrido and Sycara, 1996; Sen et al., 1997; Luo et al., 2000; Tsuruta and Shintani, 2000; Franzin et al., 2004).

In addition, in (Yokoo and Hirayama, 2000) the authors described a complete and generic solution strategy, called asynchronous backtracking (ABT), to solve any distributed problem using DisCSP (distributed constraint satisfaction problem) formalism (Yokoo et al., 1990). In this approach, the agents act asynchronously by sending point-to-point messages according to their predetermined priority[3] order. Nevertheless, this approach presents, on the one hand some limitations for large and complex problems.[4] On the other hand, ABT can be applied only to non-dynamic problems, where no incremental constraint propagation is required. Therefore, we chose to use ABT as a witness approach in our experimental evaluation on *static* instances of the utilized MS problems, in order to empirically prove the correctness of our results.

[2]To simplify the problem, we assume that all the attendants are in the same city.

[3]This order is used to avoid the fall of agents into an infinite processing loop and then to guarantee the completeness of the algorithm.

[4]The proposed methods applied to the ABT algorithm, to make it able to handle multiple local variables, are neither efficient nor scalable (Yokoo and Hirayama, 2000).