7
Multi-Agent Meeting Scheduling

In multi-agent systems (MAS), applications that directly interface with daily human activities represent a rich avenue for exploration. This paper dives into a potentially impactful application of MAS, targeting a well-known real-world challenge: meeting scheduling. While there have been previous efforts to address this challenge, we believe that the time is right to revisit this task as a blue-sky challenge for the MAS community.

Traditional scheduling methodologies rely on static, sub-optimal support tools that are susceptible to inefficiencies that include repeated rescheduling, and the overhead for the humans affected per scheduling attempt remains substantial. This opens an intriguing challenge for the MAS community: What if a collection of autonomous agents could extend human capabilities, designed to adapt and negotiate, making scheduling more dynamic and less time-consuming? The potential of collective time saved is substantial, not only in a reduction of human effort due to fewer rescheduling attempts, but also in better alignment of schedules. Furthermore, the privacy of participants can be better preserved.

We argue that the richness of this domain is of interest to the MAS community and that recent advances in AI open up new ways for tackling this challenge. In this paper, we set the stage for this research direction, focussed on the use of MAS to support an age-old, yet fundamental and pervasive task.

This chapter has been published at the Adaptive and Learning Agents (ALA) workshop at AAMAS 2024 [125].

7.1 Introduction

Meetings with others, for social and work-related interactions, form a crucial part of our daily lives. A 2014 survey by Ovum[1] found that employees meet eight times per week on average and that this number has been rising over the years. More specifically, executive management and higher meet on average 12 times a week and VPs, directors and C-level roles in highly collaborative industries reach an even higher average of 17 meetings per week.

These business meetings need to be scheduled, which takes an average of 26–30 minutes per meeting per participant according to a blog by Doodle[2]. This makes scheduling meetings a major time investment for the average employee, who likely has to schedule their own meetings. Higher-ranking roles often have assistants who perform this scheduling task on their behalf and only occasionally ask their bosses for confirmation. Furthermore, manual scheduling can lead to sub-optimal schedules, due to the complexity of the problem. Tools that support solving this problem are popular[3], but are often suboptimal as they only solve part of the problem. There is much to be gained from improving the process of scheduling, which is also recognised by industry[4].

We informally define the meeting scheduling problem (MSP) as the problem of finding a time slot of a desired duration in which the intended set of participants (or an acceptable subset thereof) commit to attending the meeting at an agreed location. We note that the location can be on-site, online, or a mixture thereof. Furthermore, the notion of an "acceptable subset" makes this *de facto* a family of problems, as it leaves unspecified who determines what defines the acceptability of that subset. In terms of complexity, the problem becomes easy, if this is determined by the one that initiates the scheduling (authority), and most complex if acceptability is determined by a group process amongst the intended participants.

The meeting scheduling problem, being such a major part of daily human life, has seen decades of attention from the computer science community, first appearing in the 80s [63, 102], often modelled as a (form of a) constraint satisfaction problem (CSP) [150]. Researchers have attempted solving the multi-agent MSP using market-based approaches [45] and negotiation approaches, where agents, representing users, negotiate over meeting time slots [136, 78]. In the years after, the problem consistently continued to receive attention among researchers (e.g., [60, 55, 34, 167, 94, 85, 155]) across several communities.

Despite the MSP being a common and relatable problem that has seen considerable effort from the research community, we are still not close to a system that alleviates most of the burden. Difficulties in learning human preferences, communication with humans, and the complexity of decentralized mixed-motive multi-agent problems render the MSP challenging. Many of these challenges are recognised as open problems in cooperative AI [36]. With the recent successes in (multi-agent)

[1] Ovum 2014 - Collaboration 2.0: Death of the Web Conference (As We Know It)
[2] https://doodle.com/en/resources/blog/study-reveals-time-spent-with-scheduling/
[3] https://www.gartner.com/reviews/market/scheduling-automation-software
[4] https://www.mckinsey.com/capabilities/operations/our-insights/smart-scheduling-how-to-solve-workforce-planning-challenges-with-ai

(deep) reinforcement learning [146], large language models (LLM), and reinforcement learning from human feedback (RLHF) [144], we believe that now the time is right to revisit the MSP as a rich and rewarding real-world challenge for the MAS community. We also believe that the various communities within computer science that have studied this problem can come together to meet this challenge and jointly achieve far better solutions than currently available.

In this paper, we lay out the necessary groundwork for tackling this problem. We discuss the characteristics of the problem and try to isolate its distinct components. We believe that a decentralised negotiation-based solution is the best-fitting approach to solving the MSP for scalability and practical reasons; this, therefore, forms the basis of our effort.

7.2 THE MEETING SCHEDULING PROBLEM

The following anecdotical meeting scheduling process illustrates the richness of the MSP: Alice must schedule a meeting with 4 colleagues, of which 2 are notoriously busy. In his role as organiser, Alice first asks the 2 busy participants about their constraints and options. The first slot found is too far in the future, so the organiser reduces the meeting duration, allowing for two earlier slots. Alice proposes these slots to the other participants. One of them, Bob, already has other obligations conflicting with both slots, but might be able to reschedule a meeting and requests the others to agree on one of the two slots. After the agreement is made, Bob commits to that slot after rescheduling his previous commitment.

In the introduction, we provided a simple and informal definition of MSP that nonetheless already introduces complexity by referring to "an acceptable subset". Furthermore, under the hood of this definition lurk additional complexities, as mentioned, *e.g.*, by Berger et al. [23]: Each participant must be able to reach the meeting location, attend for the entire duration and reach the next meeting location on time. This refers to travel time between meetings and to means of transportation. Even in online meetings, one must be in a place where one is allowed to speak, and that is quiet enough to hear what is being discussed. Aside from such practicalities that complicate MSPs, there are also numerous human aspects to consider, e.g., participants having ulterior motives and/or hidden agendas, strategic voting, powerplay, and incomplete revelations of potential meeting slots. Any and all of these have an impact on what information they are willing to share and when, how much importance they attach to the meeting and some of its intended participants, and how many attempts have to be made to arrive at a feasible solution.

7.2.1 EARLIER FORMALISATIONS

The MSP naturally lends itself to be formalised as a constraint satisfaction problem (CSP) [150]. Formalised in this manner, all the techniques for solving CSPs are applicable. This includes centralised and distributed approaches. Centralised approaches boil down to efficient search strategies in the solution space defined by means of hard constraints or strategies for optimising the utility when using soft constraints. In the distributed approaches, solving the CSP is distributed to

Table 7.1: Characteristics of the Meeting Scheduling Problem

	Problem	**Participant**			
Description	**Type**	**Type**	**Values**		
Substitutability	Boolean	Set	Other participants		
Importance of attendance	Boolean	Continuous	0 - Irrelevant	1 - Crucial	
Calendar observability	Boolean	Categorical	None	Availability	Full
Rescheduling	Boolean	Boolean			
Role	Boolean	Categorical	Organiser	Participant	Observer
Repeated encounters	Boolean	Categorical	Yes	Maybe	No
Multiple rounds	Boolean	N/A			
Preferences	Boolean	Categorical	Invisible	On options	Visible
Arguments	Boolean	Categorical	None	Restricted format	Free text

the agents as a local problem [164]. This often scales better than centrally solving a given CSP instance and is more sensitive to information sharing on a need-to-know basis. Examples of models of specific MSPs as CSP include the assignment problem [37], private incremental multi-agent agreement problem (piMAP) [49], group activity selection problem (GASP) [38, 39], valued constraint satisfaction optimisation problem (VCSOP) [133], group scheduling problem (GSP) [93], and stable group scheduling problem (SGSP) [94].

All of the above-mentioned formalisations of the MSP simplify part of the problem and fail to capture the full richness of the MSP. Such simplifications include full visibility of other agents' preferences and assuming that decisions are made centrally. In the following, we attempt to describe the full richness of the MSP as a basis for future research on the topic. In doing so, we refrain from fully formalising the problem, as multiple viable approaches exist. Instead, we focus on the characteristics of the MSP that must be considered when solving this problem.

7.2.2 CHARACTERISTICS

We identified a set of characteristics of the problem of scheduling meetings as well as of the participants involved in the meetings. An overview of these characteristics can be found in Table 7.1. We deliberately make a distinction between these two sets of characteristics. The problem characteristics map the different aspects of the problem that we can either consider or exclude when scheduling meetings. The participant characteristics describe the potential differences that participants have in relation to others within the considered characteristics of the problem. As soon as at least one participant has a certain characteristic, the problem must accommodate this.

To give a few examples of participant characteristics; within an MSP, a participant might have full visibility of the calendar of only part of the group of participants. Participants might also have different views on the importance of other participants' attendance and substitutability. For example, Alice might feel that Bob can be substituted by Carol, whereas Bob feels he cannot be substituted at all. As another example, Alice might feel that Dan's attendance is crucial, whereas Dan does not think the meeting is that important and will only attend if Erin will.

The characteristics we listed can be used to approach the problem in a systematic manner. Due to the richness of the problem, the complexity is also high, and we might want to approach it in incremental steps of complexity. Our characteristics form a structure of challenges that can be attempted in isolation. We will describe the characteristics one by one.

- **Substitutability**: If included, allows for the substitution of (some or all) participants by others. This information is to be observable by the participants. As a participant characteristic, we consider two cases: `Simple`; the set of participants that this participant can be substituted with. This models only the view of a given participant on who can substitute them. `Full`; for this participant, their full view on who (including themselves) can be substituted by whom. Differences in opinion need to be evaluated by all participants.

- **Importance of attendance**: If included, considers that some participants are more important for the meeting than others. Per participant, this is a value in $[0,1]$ representing the importance of attendance of this or another participant, according to this participant.

- **Calendar observability**: If included, allows sharing of calendars between participants. Per participant, there are three possibilities: `None` if no part of this participant's calendar can be directly observed. `Availability` if only the availability of this participant can be observed. `Full`: the participant's calendar is fully observable by others.

- **Rescheduling**: If included, meetings can be rescheduled to clear slots for other more important meetings. Per participant, whether this participant has the authority and capability to reschedule existing meeting commitments to free a slot.

- **Roles**: If included, participants can have different roles in the MSP. Per participant, these are: `Organiser` if this participant is the organiser. `Participant` if this participant is intended to attend the meeting as a participant. `Observer` if this participant is intended to attend the meeting as an observer.

- **Repeated encounters**: If included, allows for multiple encounters between agents over the course of scheduling different meetings. Per participant, whether the participant is encountered repeatedly.

- **Multiple rounds**: If included, allows for multiple rounds of back-and-forth communication before an agreement is made.

- **Preferences**: If included, consider preferences over meeting slots instead of simple 'yes' or 'no' answers on availability. `none` if this participant only answers 'yes'/'no' to offered slots. `on options` if this participant provides preferences over offered slots. `full` if this participant provides access to their full preference profile.

- **Arguments**: if included, permits arguments to be added to answers regarding availability. Per participant, these are `None`: the participant neither has the capability or authority to add arguments to their answers, nor the ability to interpret arguments made by others. `Restricted format`: the participant can only add or interpret arguments of a prespecified restricted format. `Free text`: the participant can add and interpret text arguments free of other restrictions. The arguments characteristic can be used, for example, to model that a participant provides a conditional answer, *e.g.*, needs consultation with the human user, or that a 'yes' is only valid if an agreement can be found within a given time.

7.2.3 PERFORMANCE CRITERIA

We argue that the following abstract criteria should be considered when evaluating the performance of a solution to the MSP:

- **Obtained utility**: When considering preferences in the MSP, one can measure properties over the utilities attained by all agents when used as a global performance measure. Examples are average utility, Pareto-optimality, distance to Nash product or Rawls point, [117, 120]. One can also look at the utility attained by an individual agent as a local measure.

- **Scheduling success**: The percentage of the meetings that could be scheduled. This measures the effectiveness of a meeting scheduling solution in finding common slots and aligning calendars. It should be easier to obtain a perfect score when the number of agents involved is lower or when the density of meetings is low, but becomes an interesting measure when the opposite is true.

- **Privacy preservation** [56, 54, 49]: When observability is (partially) enabled, which is likely true for real-world scenarios, then it becomes important not to reveal too much information, nor share that information with others.

- **Need for rescheduling**: This can be considered an efficiency measure. If a need for rescheduling meetings arises frequently, this could indicate that the agents are not good at estimating future conflicts and that they schedule meetings too easily.

- **Time investment of humans**: As we advocate to approach the MSP via a human-in-the-loop hybrid intelligent approach, humans must be included in the scheduling process, *e.g.*, for preference elicitation or permission in exceptional situations. However, not bothering the human too much is essential for any system to achieve advantages over conventional meeting scheduling methods.

- **Trust and acceptance**: If humans do not trust that the agent will properly schedule their meetings, adoption will be compromised. We note that asking for too little input from humans might be detrimental to trust in the system and the quality of its solutions.

- **Computational cost**: Considering the complexity of the many variants of the MSP, it is important to pay attention to the computational cost incurred by systems for solving this problem.

7.3 NEGOTIATION IN MEETING SCHEDULING

As mentioned in Section 7.1, we can distinguish between market-based and negotiation-based approaches to solving the MSP from a multi-agent perspective. Market-based approaches assume that agents are self-interested [45] and are generally based on the ideal that fairness (*e.g.*, maximum social welfare) can be guaranteed through mechanism design, where the goal is to design a mechanism that satisfies both the incentive-compatible (IC) property (*i.e.*, agents are truthful about preferences) and the individually rational (IR) property (*i.e.*, you cannot receive a negative pay-off from the mechanism). Some success has been achieved using, for example, Clarke tax [30, 44] under simplified conditions, which do not hold up in real-world applications. Designing effective mechanisms for real-world multi-agent systems is theoretically challenging [34].

We argue that negotiation-based approaches are a good fit for the MSP. Firstly, a negotiation approach fits naturally with how humans agree on meeting times. Delegating the legwork to AI agents does not interfere with this and would enable an effective hybrid intelligent solution to this problem, where human capability is extended with AI. Secondly, negotiation is distributed in nature and does not *per se* require a trusted central authority. Thirdly, in negotiation, the practice is only to reveal information on a need-to-know basis, which promotes privacy and is part of the responsibility by-design approach we subscribe to [42].

If we do not require participants to reveal all their preferences and constraints and allow multiple scheduling attempts, then we are basically in a negotiation setting. This is how humans schedule meetings without tooling, often via email, which is cumbersome and inefficient. Tools like Doodle and When2meet can be considered single-shot negotiations [2] as they eliminate the multiple-round component while lowering participants' privacy. We believe negotiation methods make the most sense as we aim for multiple-round, privacy-preserving scheduling.

7.3.1 NEGOTIATION PROTOCOLS

Agents must communicate with each other to find agreements. Open communication with other agents in the form of "cheap talk" [35] or with humans in the form of natural language is possible but renders the problem more complex. We deem it more efficient to use negotiation protocols to aid the negotiation process in finding cooperative solutions. Such protocols restrict the type of messages and order in which they are sent [143].

We are not the first to propose negotiation for MSP; examples of proposed protocols for MSP are the single proposer mechanism (SPM) [93] and the distributed score-based multi-round (DSM) negotiation mechanism [49].

7.3.2 Human preferences

Agents representing humans in negotiations should attempt to optimise outcomes based on human preferences. We, therefore, consider preference elicitation and estimation ([27, 154, 153, 138]) as core components in negotiation-based approaches for solving the MSP.

In general, a preference model can be bootstrapped based on available historical data in the form of preference pairs through direct preference optimisation (DPO) [116]. In the case of MSP, it can be based on historical calendar data [85] and on the current state of the calendar [33]. Estimating the preferences of other humans can help in finding mutually beneficial outcomes. Opponent modelling techniques can be used to estimate these preferences while negotiating, see *e.g.*, [12].

7.3.3 Learning to negotiate

Given a protocol and preference profile, agents need to learn how to negotiate with other agents, focusing on maximising individual utility, optimising for cooperativeness (*e.g.*, social welfare), or a mixture of those, depending on the characteristics of the MSP at hand. Such agents can be trained using, *e.g.*, automated algorithm configuration [124] or reinforcement learning [19, 137, 97].

In MSP, one can assume that the environment is highly dynamic. New agents will be encountered, other agents will change their behaviour, human preferences over preferred slots will change, etc. Optimising performance means that continuous adaptation is required. An agent's policy can be retrained at fixed times based on historical interactions [95]. After training on a dataset, an agent can be guided by expert annotations to improve exploration online [90].

7.4 Discussion

The MSP is a challenging problem. Finding optimal Nash equilibrium solutions in such cooperative AI problems is known to be NP-hard [61, 32, 31]. We therefore believe emphasis should be placed on finding solutions that are good enough, but not necessarily optimal, to avoid the need for exponential time solvers. Finding sufficiently good solutions also avoids problems caused by agents aiming to maximise utility deviating or rescheduling for minuscule improvements, which hurts mutually beneficial cooperation [119].

Another important point concerns interaction with humans that are not represented by agents. In the adoption of automated meeting scheduling systems, there will be a transition period during which some humans are represented by agents and others are not. Communication methods change and humans are likely to be less responsive, both in terms of the frequency of interaction and response time. Naturally, agents need to consider this in their scheduling behaviour.

We also have to ensure a degree of fairness in such systems. It cannot be the case that the calendar of some users will be inefficient or that they are being exploited by other agents, simply because they are not properly represented by their agent. Extra care must be taken when a group of agents is dealing with a single participant who is not represented by an agent. A lack of scheduling capabilities should not lead to a drastically less desirable outcome compared to other participants.

If we solve this problem, a societal implication is that humans might change their view on appointments as being somewhat more fluent than is currently the case. Whether this is net beneficial remains to be seen, but an effort should be made to be alert to potential negative side effects.

Finally, we reiterate that, in our view, the time is right to take on this challenge. Recent interest and advances in dealing with human preferences, aligning AI systems, cooperative AI and multi-agent systems can all come together within the domain of meeting scheduling. After a long period of off-and-on attention, the tools might now be available to tackle this problem in a manner that brings substantial benefits to the many individuals who have to regularly schedule meetings and to their organisations.