CAP: A Decision Support System for Crew Scheduling using Automated Planning

Aditya Prasad Mishra* Sailik Sengupta* Sarath Sreedharan*

Tathagata Chakraborti† Subbarao Kambhampati*

ABSTRACT

Task allocation or scheduling is known to be a difficult problem, especially in multi-agent settings. Thus, in order to solve these problems, a centralized entity that has a holistic view of (1) the entire task and (2) the capabilities of the individual agents is responsible for coming up with a schedule that needs to be followed. We look at a scenario where a centralized human entity (or planner) is given the task of preparing a day's schedule for a set of astronauts located at the International Space Station (ISS). Given the complexity of each individual task and constraints relating to the individual agents, coming up with a good schedule, let alone an optimal one, is often difficult. In this poster, we introduce a software system CAP that acts both as an editing tool to help the human planner make schedules and, powered by automated planning technology in artificial intelligence, can aid the human in (1) validating their plans, (2) getting suggestions about new ones and lastly, (3) asking for explanations whenever the automated planner's suggestion is inexplicable to the human in the loop.

INTRODUCTION

Planning and Scheduling problems have been an integral part of large Aeronautical, Mechanical and Electrical Systems. The main goal in these problems is to come up with a plan or a schedule that respect domain, temporal and organizational constraints. In this abstract, we focus on a particular instance of the crew scheduling problem– NASA Crew Scheduling (NCS) that involves a centralized human planner coming up with a schedule for the astronauts located on the International Space Station. The planner needs to reason about multiple kinds of tasks–ranging from scheduling physical exercises for astronauts to specific scientific experiments that may have additional constraints– when making these schedules. There exists a variety of challenges in this domain. First, each astronaut in the team may have a unique set of expertise that makes them suitable for some of the tasks. Second, multiple astronauts may need to work together on a particular task and thus, their schedules have to allow for it. Third, a generated schedule may need to be immediately fixed when (1) an astronaut decided to change their part of the schedule or (2) the execution of the plan goes off-course. Lastly, the centralized human planner may, due to cognitive overload, ignore some of the constraints at hand when coming up with a schedule. To overcome these challenges, we design an automated decision support system that assists the centralized human planner by leveraging automated planning techniques developed by the Artificial Intelligent (AI) community.

Unfortunately, although there has been substantial progress made in the AI research community in making such systems that can solve scheduling and planning problems, they are not ubiquitous in real-world contexts because these systems are often designed to be end-to-end, i.e. given as input a problem definition using some particular representation, they simply output a complete plan. A human, who is held responsible for the output plan, (beyond saying 'looks good') hardly has a way to interact with this system that helps explain the plan or modify parts of the plan.

Works in the areas of human-in-the-loop planning [1] and mixed-initiative planning [2] seek to address some of these concerns. In [3], researchers try to put the human in the driver's seat and create a system based on automated planning techniques that can assist the human in coming up with plans in a naturalistic decision-making scenario. In this poster, we follow suit and try to create a software called

*These authors are with the School of Computing, Informatics, and Decision Systems Engineering (CIDSE) at Arizona State University, Tempe, AZ, USA. Email: {amishra28, sailiks, ssreedh3, rao}@asu.edu

†The author works for IBM Research, Cambridge, MA, USA. Email: {tathagata.chakraborti1@ibm.com}

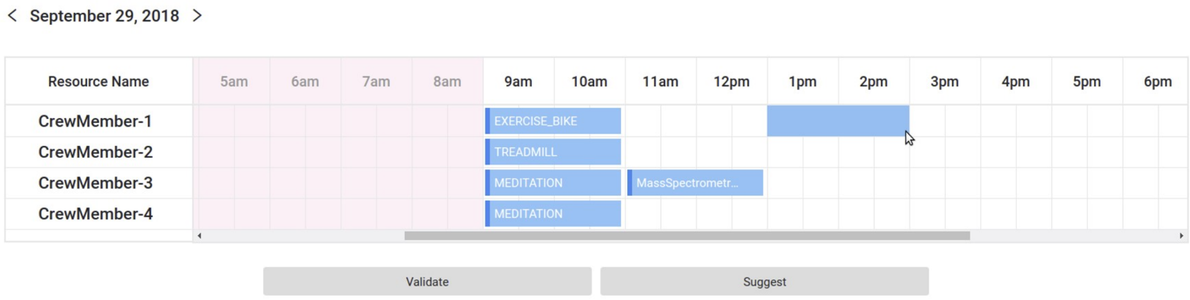

Figure 1: Illustration of the CAP interface that supports schedule authoring as well as validation and completion of schedules for four crew members.

CAP that goes beyond the present state-of-the-art schedule authoring tools such as [4] and leverages automated planning techniques to provide assistance to a human planner who is trying to come up with a functional schedule for the astronauts located on the International Space Station.

RELATED WORKS

Although Crew scheduling problems have been studied earlier using Operational Research Methodologies [5, 6], organizations like NASA have relied more on algorithmic scheduling methodologies for Deep Space missions [7]. The systems developed although meant to be used by a centralized human planner, provide a complete plan as output that often makes it difficult for a human in the loop to understand and debug the final schedule. This lack of understanding, in the long run, can lead to an automation bias and in turn, result in omission and commission errors [8]. To address some of these concerns, NASA has recently developed a tool called Playbook that is based on the principle of "Self-Scheduling" by crew members [4] and used Earth Analogs of the International Space Station (ISS) for preliminary tests. The idea behind self-scheduling, at a high level, is that the crew is expected to resolve scheduling conflicts by themselves by rearranging tasks. The only aid the system provides is to block out areas where the crew cannot schedule their tasks. This decentralized method has a set of disadvantages, as also pointed out by [4]. The key issue is the lack of a holistic view of the entire team and their limited knowledge accurate only with respect to their individual objectives. For example, if a particular crew member is not aware of the priority of all tasks, they might take up resources that were necessary for a task with higher priority, thus resulting in highly sub-optimal schedules. Thus, we stick to the centralized approach and push forth the philosophy of creating assistive software that looks under the hood of these AI technologies and makes them more suitable for human in the loop.
For this poster, we build upon two related works. First, the work [9] provides us with some ideas as to how we can efficiently encode the various types of organizations constraints as a planning problem in PDDL [10]. Second, the authors [3] warn us against the use of out-of-the-box AI techniques which can do more harm than good for the human in the loop and give us valuable guidance how to situate our system on the automation ladder [11, 12] so as to be a valuable decision support system.

CAP– SYSTEM DESCRIPTION

In this section, we briefly describe the three most important components of our system – the planning domain, the user interface and the back-end technologies in automated planning that help us support the various use-cases necessary for effective decision support which we showcase later.

The Planning Domain

Our domain is inspired by the NASA Crew Scheduling domain which involves creating detailed schedules concerning the daily activities of crew members aboard a space station (such as ISS). Currently, this is done mostly done be a person with little support (in terms of decision-making) beyond the use of a well-designed user-interfaces [4], as we discussed in the Related Works. The planning task requires the ability to deal with a multitude of complex (and often evolving) constraints, and as such remains a challenging task in the mission planning pipeline. Thus, this domain provides an ideal test-bed to illustrate the usefulness of decision support components using the power of automated planning techniques. We start by modeling a fragment of the NASA Crew Scheduling domain in the Planning Domain Definition Language (PDDL) [10]. The salient features of the domain are described below[1].

- **Universal Actions** have to be done by all crew members individually. These include tasks such as doing exercises, eating meals, etc. and have time constraints specified either by the individual crew members or the organization up front.
- **Science Experiments** can only to be conducted by certain members based on their expertise in operating certain machinery or conducting certain tests. A subclass of these experiments consists of taking photos at particular times to capture certain images of earth or other celestial objects. Most of these come with hard temporal or line of sight constraints. Furthermore, some of the tasks need to be done by a team of astronauts as opposed to an individual one.
- **Communication Tasks** send across the results of experiments to the base station and/or receive updates about new tasks that need to be completed. These tasks often have temporal constraints mostly due to the time zone at the base station and orbital speeds.
- **Maintenance Tasks** involve repair and cleaning of equipment used for experiments. These tasks often have constraints based on the expertise of certain members (one particular member knows how to clean a specific scientific instrument). while ensuring that a single person is not always allocated maintenance tasks.

User Interface

The user interface for CAP (shown in Figure 1) supports plan authoring by letting users drag their mouse over a time scale to create activities. The user can specify the activity type and the astronauts who should be assigned to that activity. It also lets the user move the different tasks and edit its properties such as the duration, the assigned individual, etc. On hovering over a particular activity (the blue boxes in 1), it shows the details associated with the activity. Lastly, it provides three buttons– two of them (validate and suggest) show up by default and the other that enables the user to ask for an explanation is displayed only after the user asks for a suggestion from the system. We now describe the AI technology that provides the functionality behind these buttons.

Automated Planning Technology

Let us denote the plan or schedule that needs to be made in order to solve a planning problem as π. This can be viewed as a concatenation of two partial plans, i.e. $\pi = \langle \pi_p, \pi_f \rangle$ where π_p denotes the past, i.e. the partial plan that has already been made and π_f denotes the future tasks that need to be done for achieving the goal. We now define how automated planning technologies can help in validating π_p and coming up with π_f.

- *Plan Validation* – When a human makes a schedule using the software, they might not have considered all the constraints imposed by the organization or the individual astronauts. Thus, the partial plan π_p may not be realizable in practice. Plan validation using VAL [13] allows them to check if the partial plan π_p is executable. When it isn't, the software can point out the constraints that are presently being violated, thereby helping the human on how to fix it.
- *Plan Suggestion / Completion* – Given a partial plan π_p, which may or may not be empty, the human planner can ask the system to generate the remaining schedule, i.e. π_f. To do this, we use an existing compilation in [14]. Originally, this approach was used for plan recognition but we use an effective tool that can be used for plan completion [15]. Fortunately, this completion method can often fix some of the validation errors that existed in the π_p made by the human.
- *Plan Explanations* – Often, a plan suggested by the system might be inexplicable to the human in the loop. In such case, we allow the human to ask the planner for explanations and provide explanations based on the model reconciliation technique [16].

USE CASES

In this section, we now show particular scenarios form the NASA Crew Scheduling domain that highlight how CAP can help the human planner.

[1] A detailed version of the PDDL domain can be found at `https://bit.ly/2V8cTbH`

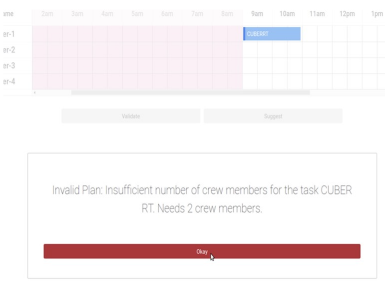

Figure 2: Validating a user's partial schedule and reporting constraint violations.

Figure 3: Suggesting the user a completion for their partial input schedule. Tasks added by CAP have a left red border.

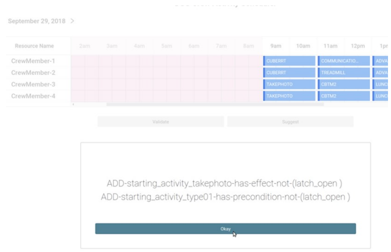

Figure 4: CAP provides explanations as to why it suggested a particular completion of a given partial plan.

Plan Validation

In the example shown in Figure 2, the user selects an action CUBERRT, which is a Science Experiment that requires two crew members to work together. The human, having forgotten this constraint, assigns the task to a single crew member. Upon clicking 'validate', CAP reports that this constructed plan π_p is invalid and highlights the constraint that the human has ignored.

Plan Suggestion

In Figure 3, the user selects three actions (indicated with a blue border to the left of the action name) and asks the planner to complete the schedule for the day, i.e. given some π_p, come up with π_f. Note that not only does the planner come up with the entire schedule (the actions added by CAP have a red border), but also adds appropriate actions before the blue actions (added by the human) to overcome constraint violations of the human's initial plan.

Plan Explanation

In the example shown in Figure 4, the human is surprised as to why a particular photo-capturing task is scheduled before a daily activity? In the human's model, the priority of the latter task is more than the first one and more importantly, completing the first task is not a necessity for doing the second one. The planner points out a particular effect of the former action enables the latter action thus, justifying the ordering of these tasks in the suggested plan.

CONCLUSION AND FUTURE WORK

In this poster, we showcase a software system CAP that follows the guidelines laid in the human automation interaction community and uses automated planning technologies to aid a centralized human planner in making daily schedules for NASA astronauts located in the International Space Station (ISS). The scarce availability of real human planners at NASA and testing scenarios with actual space crew makes the task of precisely evaluating this software a challenging task. Thus, we hope to perform human-factor studies that involves (1) training subjects to become experts in a subset of the domain and (2) creating a computer simulation of ISS for verifying the effectiveness of the generated schedules. As a first step, we performed preliminary studies with human-experts using a synthetic domain that helps us get by with only the second step [17]. We noticed, using various evaluation metrics under different conditions– there was an improvement in the quality of decisions made and efficiency of making them when decision support was used.

Acknowledgements

This research is supported in part by the ONR grants N00014-16-1-2892, N00014-18-1-2442, N00014-18-1-2840, the AFOSR grant FA9550-18-1-0067, and the

References

[1] Subbarao Kambhampati and Kartik Talamadupula. Human-in-the-loop planning and decision support. In *AAAI Tutorial*, 2015.

[2] James F Allen. Mixed initiative planning: Position paper. In *ARPA/Rome Labs Planning Initiative Workshop*, 1994.

[3] Sailik Sengupta, Tathagata Chakraborti, Sarath Sreedharan, Satya Gautam Vadlamudi, and Subbarao Kambhampati. Radara proactive decision support system for human-in-the-loop planning. In *2017 AAAI Fall Symposium Series*, 2017.

[4] Jessica J Marquez, Steven Hillenius, Bob Kanefsky, Jimin Zheng, Ivonne Deliz, and Marcum Reagan. Increasing crew autonomy for long duration exploration missions: self-scheduling. In *Aerospace Conference, 2017 IEEE*, pages 1–10. IEEE, 2017.

[5] Richard Freling, Ramon M Lentink, and Albert PM Wagelmans. A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm. *Annals of Operations Research*, 127(1-4):203–222, 2004.

[6] Alberto Caprara, Matteo Fischetti, Pier Luigi Guida, Paolo Toth, and Daniele Vigo. Solution of large-scale railway crew planning problems: The italian experience. In *Computer-aided transit scheduling*, pages 1–18. Springer, 1999.

[7] Mark D Johnston. Spike: Ai scheduling for nasa. In *Sixth Conference on Artificial Intelligence for Applications*, pages 184–190. IEEE, 1990.

[8] R. Parasuraman and D. H. Manzey. Complacency and bias in human use of automation: An attentional integration. *Human Factors: The Journal of the Human Factors & Ergonomics Society*, 2010.

[9] David E Smith, Jeremy Frank, and Ari K Jónsson. Bridging the gap between planning and scheduling. *The Knowledge Engineering Review*, 15(1):47–83, 2000.

[10] D. Mcdermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL - the planning domain definition language. Technical Report TR-98-003, Yale Center for Computational Vision and Control,, 1998.

[11] T. B. Sheridan and R. Parasuraman. Human-automation interaction. *Reviews of human factors and ergonomics*, 2005.

[12] Raja Parasuraman, Thomas B Sheridan, and Christopher D Wickens. A model for types and levels of human interaction with automation. *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, 30(3):286–297, 2000.

[13] R. Howey, D. Long, and M. Fox. VAL: Automatic Plan Validation, Continuous Effects and Mixed Initiative Planning Using PDDL. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)*, pages 294–301, 2004.

[14] Miquel Ramírez and Hector Geffner. Plan recognition as planning. In *IJCAI*, 2009.

[15] Sailik Sengupta, Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. RADAR - A Proactive Decision Support System for Human-in-the-Loop Planning. In *ICAPS Workshop on User Interfaces for Scheduling and Planning (UISP)*, 2017.

[16] Tathagata Chakraborti, Sarath Sreedharan, Yu Zhang, and Subbarao Kambhampati. Plan Explanations as Model Reconciliation: Moving Beyond Explanation as Soliloquy. In *IJCAI*, 2017.

[17] Sachin Grover, Sailik Sengupta, Tathagata Chakraborti, Aditya Prasad Mishra, and Subbarao Kambhampati. ipass: A case study of the effectiveness ofautomated planning for decision support. *Naturalistic Decision Making*, 2019.