Decision Support for Crew Scheduling using Automated Planning

by

Aditya Prasad Mishra

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master Of Science

Approved April 2019 by the
Graduate Supervisory Committee:

Subbarao Kambhampati, Chair
Erin Chiou
Hemanth Kumar Demakethepalli Venkateswara

ARIZONA STATE UNIVERSITY

May 2019

ABSTRACT

Allocating tasks for a day's or week's schedule is known to be a challenging and difficult problem. The problem intensifies by many folds in multi-agent settings. A planner or group of planners who decide such kind of task association schedule must have a comprehensive perspective on (1) the entire array of tasks to be scheduled (2) idea on constraints like importance cum order of tasks and (3) the individual abilities of the operators. One example of such kind of scheduling is the crew scheduling done for astronauts who will spend time at International Space Station (ISS). The schedule for the crew of ISS is decided before the mission starts. Human planners take part in the decision-making process to determine the timing of activities for multiple days for multiple crew members at ISS. Given the unpredictability of individual assignments and limitations identified with the various operators, deciding upon a satisfactory timetable is a challenging task. The objective of the current work is to develop an automated decision assistant that would assist human planners in coming up with an acceptable task schedule for the crew. At the same time, the decision assistant will also ensure that human planners are always in the driver's seat throughout this process of decision-making.

The decision assistant will make use of automated planning technology to assist human planners. The guidelines of Naturalistic Decision Making (NDM) and the Human-In-The -Loop decision making were followed to make sure that the human is always in the driver's seat. The use cases considered are standard situations which come up during decision-making in crew-scheduling. The effectiveness of automated decision assistance was evaluated by setting it up for domain experts on a comparable domain of scheduling courses for master students. The results of the user study evaluating the effectiveness of automated decision support were subsequently published.

Dedicated to my family. It is because of your continuous support, I am here.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor and chair Dr. Subbarao Kambhampati, who has been a great mentor guiding me in my research, and helping me in the right direction for my research. I am forever grateful to him for giving me this opportunity to work in interdisciplinary areas like Explainable AI and Human Aware AI. I would also like to thank my other committee members Dr. Erin Chiou and Dr. Hemanth for their suggestions to further improve some of the sections of my thesis.

I want to thank everyone here at Yochan Lab at Arizona State University: for their constructive criticism, helpful suggestions, and support. They helped me a lot and greatly inspired me. I am particularly grateful to Sarath, Sailik, Tathagata, Sachin, and Alberto with whom I had the privilege to work with on multiple projects.

I would also like to thank my family for their help and support. You guys have stood by me in difficult of times. And I am grateful for your love and understanding. Without your support, this thesis would not have been possible.

TABLE OF CONTENTS

Page

LIST OF FIGURES vii

CHAPTER

1 INTRODUCTION 1

1.1 Contributions of This Thesis 4

1.1.1 `CAP` - Decision Support for Crew Scheduling 4

1.1.2 `iPass` - Evaluation of the Effectiveness of Automated Planning for Decision Support 5

1.2 Thesis Organisation 6

2 BACKGROUND AND RELATED WORK 7

2.1 Background 7

2.1.1 Automated Planning 7

2.1.2 Human-in-loop Planning and Scheduling 9

2.1.3 Explanations with Model Reconciliation 10

2.2 Related Work 11

2.2.1 Related Work in Human -in-loop Planning 11

2.2.2 Related Work in Explainable Ai 14

2.2.3 Related Work in Automated Decision Assistance 16

2.2.4 Related Work in Decision Assistance in Crew Scheduling 19

3 CAP - DECISION SUPPORT FOR CREW SCHEDULING 21

3.1 Modelling the Crew Scheduling Problem 21

3.1.1 Planning Formulation 22

3.1.2 Detailed PDDL Based Model Description 22

3.2 System Architecture 25

3.2.1 Control Flow 25

CHAPTER Page

3.2.2 User Interface 26
3.2.3 Parsing and Backend Architecture 29
3.2.4 Automated Planning Technology 30
3.3 USE CASES 31
3.3.1 Plan Validation 32
3.3.2 Plan Suggestion 33
3.3.3 Plan Explanation 34
3.4 Limitations of `CAP` 35
3.5 Conclusion 36
4 IPASS : EVALUATION OF THE EFFECTIVENESS OF AUTOMATED PLANNING FOR DECISION SUPPORT 39
4.1 Introduction 39
4.2 `iPass` – System Overview 40
4.2.1 The iPass Domain and Interface 41
4.2.2 Decision Support Components 42
4.2.3 Comparison with `CAP` 43
4.3 Aim of the Study 43
4.4 Experimental Results 46
4.4.1 User Study - Evaluation Process 46
4.4.2 User Study - Information about Participants 47
4.4.3 Hypothesis H1 48
4.4.4 Hypothesis H2 52
4.4.5 Hypothesis H3 55
4.5 Limitations 56

CHAPTER Page

4.6 Conclusion 57
5 CONCLUSION AND FUTURE WORK 58
REFERENCES 61
APPENDIX
A RAW DATA 64
A.1 Domain for `CAP` 65
A.2 A Daily Plan Generated for `CAP` 69

LIST OF FIGURES

Figure Page

1.1 A Schedule Prepared for the Crew by Mission Planners after Consulting with Various Stakeholders 2

2.1 Comparison Between Nasa's Playbook and Our System CAP 20

3.1 Front Panel 23

3.2 Control Flow Diagram of the `CAP` Interface. 26

3.3 Front Panel 27

3.4 Task Panel 28

3.5 Control Flow for Validating a User's Partial Schedule and Reporting Constraint Violations. 31

3.6 Validating a User's Partial Schedule and Reporting Constraint Violations. 32

3.7 Control Flow for Suggesting the User a Completion for Their Partial Input Schedule. 33

3.8 Suggesting the User a Completion for Their Partial Input Schedule. Tasks Added by `CAP` Have a Left Red Border. 35

3.9 Control Flow for How Explanations Were Generated for a Particular Completion of a given Partial Plan. 36

3.10 `CAP` Provides Explanations as to Why It Suggested a Particular Completion of a given Partial Plan. 37

3.11 An Example Scenario Demonstrating Explanation Using Model Reconciliation. 38

4.1 Illustration of the `Crew Scheduling` interface. 40

4.2 Average Time Taken (along with Confidence Interval of 95%) by a Participant to Complete the Two Parts of the Study for Each Condition c_i^1 and c_i^2. 47

Figure Page

4.3 Average Number of times Participants Added, Deleted, Rearranged Courses or Clicked 'check' While Making an Ipos for All the Conditions c_i^1 50

4.4 Average Number of times 'validate' Was Clicked in Condition c_1^1 and c_3^1 50

4.5 Average Number of Times 'suggest' Was Clicked in Conditions c_2^1 and c_3^1 51

4.6 Average Score for Subjective Q3 for Conditions c_i^1 51

4.7 User Agreement Metrics for the Statement 'q2: The Feedback from the Interface Helped the Ipos Making Process' for Each Condition c_i^1 52

4.8 Time Difference $\delta T(C_i)$ Between Two Tasks c_i^1 and c_i^2 of ipos Planning for Every Condition c_i (along with Confidence Interval of 90%) 53

4.9 Time Taken by Experienced (in Yellow) and Non-experienced (in Blue) Users to Make the First Ipos (c_i^1) 53

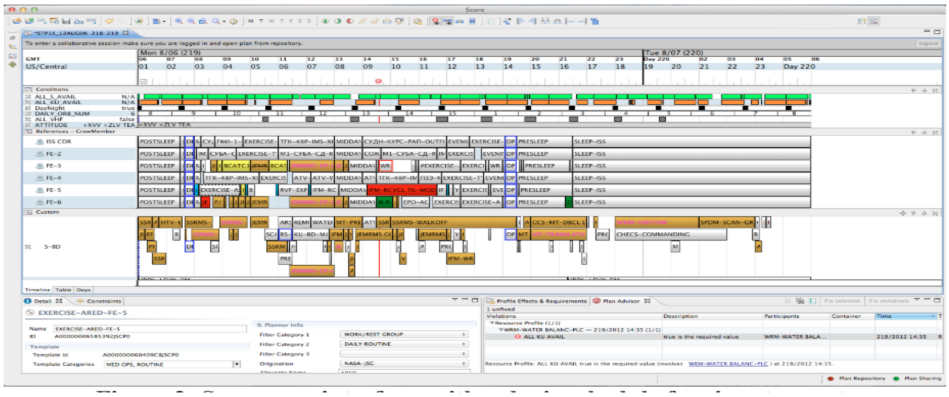
4.10 Feedback of Non-experienced Users about the Statement 'q1: The Planning Task Was Pretty Simple for Me' 54

Chapter 1

INTRODUCTION

The problem of mission-critical decision making is a ubiquitous one. Decision making under such scenarios is characterized by uncertainty, multiple stakeholders and complex interrelated resource constraints and the decision maker is often required to come up with a decision under given time constraints. In such a situation a mistake, even a minuscule one could lead to catastrophic failures. The critical nature of such a job asks for the decision maker to not only have expert knowledge about the task at hand but also keep track of any changes to the state of the system. Moreover, given the fact that the system can have multiple stakeholders, decision makers must try to arrive at a decision which is acceptable to all stakeholders and at the same time maximizes the overall utility. The decision-making process, therefore, becomes a lengthy and time-consuming one and the decision makers can easily get overwhelmed. Even after much thought and deliberations by the human decision maker, the final decision might still not be optimal or suited for the current state of the system. Therefore, some degree of assistance is often required by the human decision maker to reach a decision.

For example, consider the decision support for the International Space Station(ISS). These massive structures not only solely house a crew but also conjointly are tasked to perform scientific studies and routine activities. Scientists who are a part of the science team, perform various scientific experiments. Support crew performs auxiliary tasks like provision, repair or photo operation. Apart from their professional duties, crew members even have to portion time for daily routine activities like breakfast, lunch, exercise or sleep. Mission planners are responsible for coming up with daily



Picture credit -> Marquez, Jessica J., et al. "Supporting real-time operations and execution through timeline and scheduling aids." *43rd International Conference on Environmental Systems*. 2013.

Figure 1.1: A Schedule Prepared for the Crew by Mission Planners after Consulting with Various Stakeholders

schedules for the crew. Typically decision-making becomes a challenge for the mission planners because they need to arrive at a schedule that tries to maximize the objective achieved by each crew member. Additionally, they are responsible for ensuring that the plans are up to date and revised throughout the mission. While coming up with an idea, they may also need to prioritize various objectives and be able to satisfy resource constraints.

The mission planners act as supervisors and try to finalize the schedule along with mission objectives even before the mission take off. NASA has been following this approach for quite some time as mentioned in Marquez *et al.* (2013). However, the current practice often leads to unstable plans in a dynamically changing system like that of ISS. It also makes the routine schedule of the crew quite inflexible. It may happen that a small number of experiments did not produce the expected outcome or some tasks took more time than expected, which may, in turn, create discrepancies in the pre-planned schedule. In such scenarios, the crew cannot always be dependent on the mission planners to give them a new schedule on time, as there could be

significant communication delay and the team will have to spend additional time to provide latest updates to the ground staff. The need to include mission planners in the loop also restricts the crew's ability to update their schedule, since they may have to coordinate not only with mission planners on the ground but also with other peers making the entire process undesirable and even unreliable.

As we can see from the above discussion, decision making for crew scheduling is a tough task for human planners. So, in such scenarios instead of human-decision makers, what if we make use of an automated decision-making system. While, such a decision-making system can arrive at plans or schedule which are correct and optimal quite efficiently, however decisions by such a system (in our case, schedules for crew scheduling) may be inexplicable to the human planner. For example, In the crew scheduling domain, the system may come up with a schedule that precedes REPAIR_LATCH task before TAKE_PHOTO task. However if the human planner is unaware of this usability of LATCH for taking a photo can get confused as to why this ordering was done.

In Smith (2012), the author has suggested an explanation as a way to establish common ground among various stakeholders while discussing the schedule generated by the system. The author has also imagined a continuous iterative process until everyone agrees with a plan or schedule with the explanation being used at the end of each round to come to a conclusion. Thankfully, there is much literature on developing agents that are capable of explaining the approach taken by the assistant to reach a decision. The current consensus in works related to "Explainable AI" as stated by Miller (2017) is that the explanations should help the user not only understand the current decision but also provide the reason as to why it was chosen over alternative solutions. One approach for simplifying the problem of explanation generation is to make use of a symbolic decision-making model.

In addition to Explanation-based decision making, we make use of design guidelines from the Naturalistic Decision-Making framework. Naturalistic decision-making(NDM) Klein (2008) is a framework that studies the higher cognitive process of decision making by human actors for such mission-critical situations. It ensures that the user is always in control of the decision-making process. We also consider the principles of Human-In-The-Loop (HILP) decision making while designing the application use cases.

In this current thesis, we will develop an automated decision assistant for crew scheduling domain. We will ensure that the use cases in an automated decision assistant which comes up in naturalistic decision-making scenarios are appropriately tested. At the same time, we will evaluate such a system by performing a user study to evaluate the effectiveness of principal components of automated decision support.

1.1 Contributions of This Thesis

Our main contribution can be subdivided into two major sections. The first section will deal with an application named as `CAP` and the second section deals with a user study evaluation on an application named as `iPass` . I have described in brief below, what each of these sections entails.

1.1.1 `CAP` - Decision Support for Crew Scheduling.

We developed `CAP` which is an automated decision support system to provide decision support for human planners involved in creating schedules for crew members atop a space station like ISS. While developing this decision support application, we made sure that the human planner is at the center of the decision-making process rather than the automated planner. The idea that the human should be the primary

decision-maker is one of the core ideas of Naturalized Decision Making(NDM) and Human-in-Loop Planning(HILP). CAPbeing based on the rules of NDM and HILP makes the human planner the primary decision maker with the decision assistant playing a supportive role. The objective of the human planner is to reach a schedule acceptable to multiple stakeholders whereas the decision assistant aims to assist the human whenever there is an ask for such support from the human planner. We will discuss all of the features of CAP and then look at some use-cases which are real-time scenarios involving usage of decision support for crew scheduling.

1.1.2 iPass - Evaluation of the Effectiveness of Automated Planning for Decision Support

Due to the absence of domain experts (in our case mission planners), we were unable to evaluate the effectiveness of the decision support by CAP . Hence, we developed a second application with the principles of decision support from CAP . We have named this application iPass . The objective of this application was to assist the students of a particular University in coming up with a course plan or IPOS for their degree program which could be an undergraduate, graduate or a combined degree. This application along with allowing students to add or delete various subjects, also allowed them to select a committee for defense and allowed them to choose a specialization for their program. While selecting a course schedule, different constraints were enforced like mandatory inclusion of deficiency courses, selection of courses as per specialization and many more similar restrictions. We evaluated using various subsets of students with varying initial states and varying level of decision support which served as our control parameters. Our evaluation was dependent on how well or how efficient the participants were while generating the course schedule given various different decision support control parameters. We will discuss the methodology for

the user study; various hypothesis studied during the study and the limitations with our study. We have also evaluated the results of this study using statistical measure to show the degree of effectiveness of decision support in assisting the human planners.

1.2 Thesis Organisation

Chapter 1 introduces the challenges of decision making in mission-critical scenarios like that of crew scheduling. It gives a basic understanding of how difficult it is for a decision maker to decide on a plan or a schedule given a model. This chapter also introduces various solutions to support decision making utilizing automated decision support.

Chapter 2 provides us with the necessary background and the related work related to the field of automated decision support. In this chapter, we look in details at various approaches to develop personal decision assistants and the evolving theories behind such assistants. We will also compare and contrast features of an already existing tool with our application `CAP` .

Chapter 3 delves in detail about `CAP` . Here, we have mentioned the software architecture, the features and relevant use-cases in detail. We have also provided an overview of how the planning problem is formulated and how we can perform decision assistance to a human planner by keeping the human-in-the-loop.

Chapter 4 describes `iPass` in detail with all of its features. In the same chapter We have also described many of our hypothesis for the user study on `iPass` . Upon getting the results of the user study, we have evaluated it by utilizing various statistical measures. We then compare the results to check if our hypothesis were valid.

Chapter 5 is a short section which gives a conclusion to the thesis with a short discussion on the future work to be done.

Chapter 2

BACKGROUND AND RELATED WORK

2.1 Background

2.1.1 Automated Planning

Our software system CAP will make use of automated planning or AI planning methodologies to develop a decision-support system. In this section, we will briefly describe automated planning or AI planning. Planning in general sense is devising a plan of actions to achieve one's goals as per Russell and Norvig (2003). As per Ghallab *et al.* (2004), The purpose of planning is to search for a sequence of actions to reach a goal state. In the context of classical planning, planning algorithms are used to figure out a sequence of actions from an initial state to a final state. If the final state satisfies particular rules for success, we then call it a goal state. The objective of any planning algorithm is to find this sequence of actions which would transpire an agent from an initial state to a goal state. Such a sequence of actions is also known as a plan. A planning problem consists of the current state, domain, and the final goal that has to be achieved.

Planning vs Acting

Planning is the process of deliberation over state-space, and any planning algorithm aims to reach a goal state. Searching for a plan always involves searching within predicted states and do not account for exigencies whereas the opposite is true for acting. Acting is the process of executing an action given the current state and activity. Acting makes use of an existing plan as a guide to move within the states.

However, acting can lead to unexpected contingencies. It is now up to the agent to perform a type of online learning to update the plan or restart a new plan from the current state. The process of decision making is to generate a plan while acting upon the decision is the actual acting on the plan. Re-planning could happen if constraints or goals changes during acting on a plan.

Model Representation in Planning

In planning problems, the real-world model is often represented as a toy model. Any model of such kind would consist of a domain, an initial state, and a goal. The domain is a written representation of the world using literals, variables, actions, preconditions, effects. The primary objective of any planning problem is to reach a goal state. We represent a state in the domain using fluents and predicates. These fluent themselves are a distinct representation (ground truths) of objects or group of objects that can change over time. Domain-specific planning only allows for generalized ground truths. An action schema constitutes of action fluent with its required precondition and its expected effect. Both the precondition and effect are a conjunction of states which can be either be positive or negative. An action is applicable if all of it's necessary preconditions are found to be true. After we have built our domain with the action schema, we can use either a forward search (progression) or backward search(regression) to find our requisite plan from the domain given an initial and a goal state.

PDDL - Planning Domain Definition language

To represent our model we make use of a representational language called PDDL (Planning Domain Definition Language) McDermott *et al.* (1998). PDDL makes use of factored representation. A PDDL domain file consists of a definition of the domain,

requirements satisfied by the domain, types of various objects in the domain, the property of a state as predicates and finally the action schemata. The action schemata contain the preconditions and effects of each action. A state can be any combination of one or more predicates. A PDDL problem file consists of an initial state and a goal state for the requisite domain. A Planner program like metric-FF(Hoffmann (2003)) finds out the sequence of actions(plans) which would lead to the goal state from the given initial state. A plan validator validates if a given plan is correct given an initial state, the goal state, and the goal. For our decision assistant, we will make use of numerical fluents to keep track of changes in time in the schedule which is a continuous resource.

2.1.2 Human-in-loop Planning and Scheduling

Human in loop planning or HILP in short as advocated by Kambhampati and Talamadupula (2015) and Chakraborty (2018) is required in many decision-making scenarios. As we have discussed earlier human decision maker's task can become quite complicated in real-world situations. At the same time leaving the entire decision making to automation could lead to inexplicable or undesirable plans. So we do need a planning mechanism which will enable humans and robots to team together.

Human-in-loop planning can happen in decision support at many different levels. One approach is to suggest a plan even before the robot or the AI system is asked about it. Another method would be to wait until the human makes a mistake and then suggest a change rectifying it. Other than this a decision support system can act a passive observer and can only come to help when asked by the human actor. The levels in which we can use automation is quite similar to levels of automation described by Parasuraman *et al.* (2000). The basic idea behind HILP in decision

making is, humans, do want assistance from the AI agent but at their terms. At the same time, the robots should only do what is expected of them, and they should assist as per the understanding between both the human and the robot. Keeping Human-in-the -loop makes the robot a better assistant in decision-making scenarios.

Just like for assistance in planning, decision assistance is necessitated while creating or preparing schedules. The AI system is regularly updated with control parameters and the mission goals of an industrial system like a spacecraft. While they can validate, create and assist any plan, keeping human in the loop is also important.

2.1.3 Explanations with Model Reconciliation

There have been some recent works in explanation generation for plans. Few of them has been mentioned in the next chapter. However, in our case, we have to consider that there is some amount of model asymmetry in between the human and the automated decision assistant. The reason for the presence of such a model asymmetry could be varied, but often it is the result of the limited capability of human actors to keep themselves updated with the latest changes at all times. It could be entirely possible that the decision assistant's model is outdated due to some external reasons. Whatever be the idea, we have to make sure that the human and the decision assistant reduce this asymmetry so that they can reach an agreement concerning the decision reached. This synergy is always a necessity in human-robot teaming domains. At the same time, We want the plan to be close to the human's model because it would make the plan appear more explicable to the human. We also always assume in decision support scenarios that the human to be a domain expert and has an internal model of the system. We expect the plan to close to the one expected by the human so that is is understandable to the human and there are no surprises for the human.

For generating such explanation which reduces the asymmetry, we will utilize the research on Model Reconciliation from existing literature of Chakraborti *et al.* (2017a). Model Reconciliation is an approach to provide a solution for the problem of model symmetry by bringing the human model closer to the model of the automation. The authors have formulated the Model Reconciliation Problem(MRP) in between a human model and a robot model, and the solution for such problems is termed a multi-model explanation. An MRP considers both the robot's model M_r and the human's approximation of it, i.e., M_h. Initially, no optimal plan was found in M_h using the same initial state and goal state because of Model differences. These model differences could be because of the absence of certain preconditions and effects or a gap in understanding of the current state. M_h is Model Reconciliation is supposed to happen when the incremental model $\widehat{M}$ can find an optimal plan for M_r. The explanation that is returned is the difference between the incremental model $\widehat{M}$ and human model M_h. One important property that is guaranteed here by the authors is minimality of the generated explanation. The explanation generated is always a minimal update to the human's model as there is no sequentiality to the search processes and all model differences are maintained in a set with precedence given to the low cost of the update.

2.2 Related Work

2.2.1 Related Work in Human -in-loop Planning

HILP as a subset of Human aware planning as suggested by Kambhampati and Talamadupula (2015) makes an effort to keep the human in the loop while acting on the environment. In case of decision support systems, the boundaries are defined by the system boundaries in which or for which the decision is being taken. Authors

in Kambhampati and Talamadupula (2015) have described four different areas for keeping the human in the loop by the robot. They call it the modalities. These are Cooperation Modality, Communication Modality, What is communicated and Knowledge Level. Accordingly, they have provided with many examples of domains with various levels of patterns can vary for Human-in-Loop-Planning or in short HILP. One important example here would be that of human-robot teaming. Here we can see that the Cooperation Modality is either teaming or collaboration, Communication could be a natural language, What is communicated could be goals, tasks or some other model information and Knowledge level could be something relevant to the domain not described by any other modalities. Keeping Human-in the-loop is vital because the human will not only make sure that the robot is working as per his wishes. The robot can learn from the environment but if it should act as per human's needs.

MAPGEN by Ai-Chang *et al.* (2004) is a practical use-case showing the importance of HILP. The primary user of MAPGEN is the MER mission tactical planners. Tactical planners will utilize MAPGEN most efficiently to reach their mission goals for the day. Keeping these planners(human)-in-loop can bear much better result for the day's scheduled goals in case the rover is faced with unexpected situations. Currently, the mixed-initiative planning for MAPGEN puts the automated planner in driver seat rather than the human-in-loop.

In the case of Mixed-Initiative Planning, humans are in the loop and can provide requirements to the planner for planning, but the cooperation is from human to the robot, with the human having to accept the plan the robot come up with. Advantage of such an approach is that the planner considers the human's objectives. The disadvantage here is that the human's aim may not always be reflected in the final plan as there are multiple stakeholders. Also, humans are not communicated as to why their changes was not considered for the final plan. In comparison, HILP has a varying

degree of modalities like Cooperation, Communication, What is communicated and Knowledge Level. Our automated decision assistance is also HILP, but it provides support for human planners keeping them in the driver's seat. It assists when asked for help by the human planner and not proactively. Plus it can also communicate an explanation back to human to reconcile any model differences.

Parasuraman *et al.* (2000) has also shown us by giving many examples of why it is essential that the human is kept in the loop. They have shown how many automation related incidents and accidents can be prevented if the human is aware of the internal state of the system. HILP becomes even more essential in decision-support because in case of decision support human does not only want to know the decision system has reached but would also like to learn why this decision has been reached? The idea of understandable plans is quite evident from the concept of explicability in Zhang *et al.* (2016). In HILP it is important that the conclusion reached must be agreeable and reasonable to humans.

Once we have kept a human in the loop, we want the domain model to be represented appropriately. Upon serving the problems appropriately, we want the system to assist the human in various ways. Sengupta *et al.* (2017) RADAR system does not only generate plans and thrust on the human but would help the human decision maker in providing validation for a hand-coded plan. It also assists the human planner to complete his/her plan from the partial plan and offer him or her with an explanation. RADAR uses methodologies for plan validation, plan explanation or plan generation developed in soliloquy. Our system CAP is quite similar to RADAR, but it is specially optimized for assisting mission planners in generating valid plans for crew scheduling Scenarios.

2.2.2 Related Work in Explainable Ai

In Parasuraman *et al.* (2000) authors have also pointed out that human operators must not be oblivious to the working of automation. They have provided historical data to support their claims. An analogy would be the case of an enterprise software solution. Through debugging, we can understand the internal states of variables in software. However, the same level of flexibility is not allowed or present in current AI systems. Explainable AI is the domain of research where the human-in-the-loop asks for explanation whenever she is unaware of the internal workings of the automation. Explainable AI or need for explanations in AI and automated systems is driving many research work in parallel. One seminal work is that of Miller (2018). The authors have discussed here the need for explanation and also highlighted what variety of explanations are necessitated for AI systems. They additionally have discussed kinds of "Explanations" for people like Constructive explanation or Attributive explanation. In their seminal work, they have also tried to explain a couple of Attributive explanations like a Social Attribution or a Causal Connection. Also according to them, explanations must give an idea about the causal history of a system. This piece of information is referred to as explanatory information. Explanations can be used to answer "What ?", "How?" and "Why ?" questions. All these questions give us appropriate reasoning based on factual assertion and causal history of an ongoing process. This process could be something like to reach a particular goal cum destination or perform a specific task or reconfigure to an appropriate configuration.

Intelligibility and Interpretability are related terms. As per Weld and Bansal (2018) Intelligibility can be introduced to the system either by exploiting the inherent interpretability of models like linear models. GAMs by Hastie (2017) and GA2Ms in Lou *et al.* (2013) are example of similar models. Linear models are easily interpretable

given they have a human-observable set of features. GA2MS were found to be more explainable then other linear models. Linear models cannot learn in research domains like image recognition or voice recognition. It is here we use an inscrutable system like Neural networks or random forests. In such cases, the models were tested by making them locally interpretable by using an algorithm called LIME. LIME by Ribeiro *et al.* (2016) only sees for local separation points or boundaries. The authors in Weld and Bansal (2018) have also suggested going for the interactive training of neural networks wherein the pictures and description can be trained together. So each picture will have its own explanation. However, in these cases, a further interactive explanation is not possible.

Verbalization by Rosenthal *et al.* (2016) is another way to represent the internal states of the system for the path or decisions they have taken. Authors have described three different properties for Verbalization which would shape a verbalization output and are dependent on the expectation of the human user. These are the abstraction, locality, and specificity. Abstraction represents the level of vocabulary used with the simplest being described only in points and the most complex one containing turns and semantic annotations. The locality describes the segment of the route a user may be interested. Specificity indicates the number of concepts or details needed to be discussed. The values are dependent on what a user expects from the robot. Moreover, a user can change the values to get a better understanding of the system.

Scheduling problems can also be represented as constraint programs. Many common scheduling problems like crew scheduling, factory scheduling has been described as constraint programs as shown by Pinedo (2016) and Dechter and Cohen (2003). There has been some work in generating explanations for constraint programs as well. The primary aim for such practices was better user experience. The user is thought to be an expert. Hence not much effort was given for model reconciliation. Earlier,

constraint programming systems did not provide much information back to the user when they were not able to find a solution. They would display some default message without actually explaining back to the user any more information about this error. The human then has to check the constraints themselves to find out what may be causing this error. Often this would require a lot of backtracking and elimination based reasoning by the human planner. The idea behind using explanations in constraint programming is to automate this process of reasoning of the user and then generate the user-oriented explanations.

Earlier work in explanations for constraint programming talks about constraints as the mode of communication between the human planner and the solver. Humans will either add/remove certain constraints and check how the solver responds. Some of the earlier works are Jussien (2001) and Jussien and Barichard (2000) . As per Jussien (2001) "an explanation is a set of constraints justifying the propagation events generated by a solver.". As per Jussien and Barichard (2000) "an explanation E for a piece of information I (current lower bound, current upper bound, value removal, ...) is a set of constraints such that its associated information remains valid as long as all the constraints in E are effectively active in the constraints system." PaLM in Jussien and Barichard (2000) has multiple examples of how explanation is being provided for a scheduling problem when either new constraints were added or deleted. The self-explain feature explains as to which constraints can change the final result variable.

2.2.3 Related Work in Automated Decision Assistance

Automated decision assistance can be seen as an area in the loop with Human-in-Loop Planning and Assistance. But many different kinds of research has been done for Automated decision assistance. The idea is to make the robot a subsididary

who either interacts or does not interacts with the human depending on human's requirement. Decision-Theoretic Model of Assistance by Fern *et al.* (2007) is one such example. The authors modeled a intelligent assistant system using an Assistant POMDP. They observed a goal-oriented agent, i.e., the human and assist it with an intelligent assistant so that assistive actions will decrease the global cost of reaching the goal. Cooperative Inverse Reinforcement Learning (CIRL) by Hadfield-Menell *et al.* (2016) also tried to model an autonomous system which would help the human in achieving its goal without actually causing any unintended consequences.

Mixed-initiative interfaces by Horvitz (1999) is another approach to decision support where the automated system must sense the humans need and can act accordingly. They have provided the following factors for successful integration of such systems

- Providing Automated services which are genuinely valuable regarding cost or time over direct manipulation.
- Uncertainty in a user's goal and focus must be modeled appropriately.
- Must make sure that user is not distracted when making decisions.
- The ideal action is inferred considering costs, benefits, and uncertainties.
- Dialog with the user to understand his focus.
- The user should be allowed to invoke automated services.
- Aim to minimize the cost of poor guesses.
- In case of too much uncertainty, the agents should gracefully degrade certain service.
- The user has the right to refine the analysis of agents.

- Socially appropriate behaviors for agent-user interaction.
- Maintain a trace or record of recent interactions.
- Learning by observation

The mentioned LookOut system infers about the User's goals by using a linear SVM classifier. Once the classification process is completed, the results from the classifier act as evidence for the uncertain purpose of the user. Given the four utilities based on action, dialog, and goal we can find an expected probability after which the intelligent agent can take action.

Mixed-Initiative interfaces by Horvitz (1999) and RADAR by Sengupta *et al.* (2017) are similar in the way they both provide proactive assistance, but RADAR also offers many other features which are absent in Mixed-Initiative interfaces. The features would include Plan Validation, Plan suggestion and Plan Explanation for the firefighting domain. Also, Mixed-initiative interface was an idea paper of how interfaces have a goal to proactively support humans should be designed whereas RADAR is an application built for proactive decision support.

Naturalistic Decision Making(NDM) is another area of interest for people working with decision Assistants. NDM has been at the forefront of analysis in large aeronautical, Mechanical and Electrical systems involving mission-critical decision making. These systems typically involve human supervisors to manage and maintain internal and external processes. These professionals usually have a predefined set of goals to be achieved. NDM studies the measure of effectiveness for the selections made underneath such high-stakes situations.

2.2.4 Related Work in Decision Assistance in Crew Scheduling

Smith *et al.* (2000) provides us with the necessary groundwork to represent a scheduling problem as a planning problem. Authors have also performed a comparison of various planning and search strategies for temporal planning problems. Crew scheduling have been studied by Freling *et al.* (2004) and Caprara *et al.* (1999) by using the operation research methodologies. NASA has previously, used scheduling methodologies for deep space missions for Hubble in Johnston (1990) and MARS rovers in Ai-Chang *et al.* (2004). An earth-based human planner uses these systems and provides a complete plan as output that often becomes difficult for a crew member in space to understand and edit. To address this, NASA has recently developed a tool called "Playbook" that is based on the principle of "Self-scheduling" by crew members as described in Marquez *et al.* (2017). Earth Analogs of the International Space Station (ISS) were utilized for the 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. To support such kind of decision making specific markers were provided by the system where the crew knew it could not schedule its task. Indeed, this method has its disadvantages as rightly pointed out by Marquez *et al.* (2017). These issues often happen due to limited domain knowledge of crew members, and often this knowledge is limited to their mission objectives. They do not have a complete understanding of the entire system. Also such kind of scheduling can create a conflict if a less critical task was scheduled before a high priority task. It would be challenging to resolve such conflicts as it would involve a lot of manual processes. In such situations, our tool would be of significant advantage as it would give a lot less cognitive load to the human decision maker and will always validate the schedule with the latest updates to the system. We have

Features	NASA's Playbook	CAP
Self – Scheduling	Supported	Not supported as we designed for Mission Planners.
Constraint Visualization	Go/No-Go visualization	Conflict Visualization
Plan Validation	Not Supported	Currently Supported
Automatic Plan Generation	Not Supported	Currently Supported
Partial Plan Generation	Not Supported	Currently Supported
Explanation	Not Supported	Currently Supported
Other Features	Notes, Scratch Pad	New Task Panel, Multi Crew Task

Figure 2.1: Comparison Between Nasa's Playbook and Our System CAP

made a contrast between both of these tools in figure 2.1. From the comparison, it is evident that the playbook does not possess many capabilities for decision support. Hence it's primary use is limited to only manual scheduling of activities. In contrast, our developed system `CAP` has support for automated decision assistance which can assist the crew in reaching an effective decision faster.

Chapter 3

CAP - DECISION SUPPORT FOR CREW SCHEDULING

In this section, we briefly describe `CAP` a decision support system for crew scheduling which was developed by integrating the capabilities of automated planning technologies like validating a plan, generating a plan or explaining a plan. Our system is a full stack software application with a web interface that helps a human decision maker to create, validate, complete a schedule for multiple crew members doing multiple tasks at multiple locations. We first described the planning domain followed by the user interface and the back-end technologies in detail. We have then described use-cases relevant to Crew scheduling for effective decision support.

3.1 Modelling the Crew Scheduling Problem

We have modeled the domain for CAP based on the NASA Crew Scheduling(NCS) problem. The primary objective of NCS has been to create schedules concerning the daily activities of crew members aboard a space station (such as ISS). Currently, the action of generating task schedules is a manual activity with hardly any support (in terms of decision-making) beyond interfacing elements Marquez *et al.* (2017). In this current section we will try to provide assistance to human planners in crew scheduling scenarios. We will formulate the problem of crew scheduling as a planning problem before utilizing the automated planning technologies to develop the decision support system for it.

3.1.1 Planning Formulation

`Crew Scheduling` in `CAP` is represented as a planning problem written in Planning Domain Definition Language (PDDL) McDermott *et al.* (1998) version 2.0. The planning problem given by Π would consist of the current state given by $\mathcal{I}$ (state in which the spacecraft is in), action model $\mathcal{M}$(which captures the actions and constraints on these actions which could be spatial, temporal or local in nature.), and the final goal state $\mathcal{G}$ (In our case it is a day's schedule). The planning problem $\Pi = \langle \mathcal{M}, \mathcal{I}, \mathcal{G} \rangle$

The solution to the planning problem Π is the schedule of activities planned for the day. The plan or schedule that needs to be made in order to solve the planning problem be denoted as π that is a concatenation of two smaller plans, i.e., $\pi = \langle \pi_p, \pi_f \rangle$ where π_p denotes the past, i.e. the partial plan that is already made and π_f denotes the future tasks or plan that needs to be made.

3.1.2 Detailed PDDL Based Model Description

The planning model developed for crew scheduling provides an ideal test-bed to illustrate the usefulness of decision support using the power of automated planning techniques. The model will include representation of both the planner and the internal human representation of it. Any validation, suggestion or explanation will be provided concerning the planner's model. Figure 3.1 shows a few instances of fluents of the domain in PDDL. The complete domain and an example problem file are copied at the end in the Appendix section. We will describe the various sections of the model next in this section.

- **Task and Actions**

  Multiple different kinds of actions are supported by our domain which are stand-in or toy actions for some of the real activities performed by crew members at

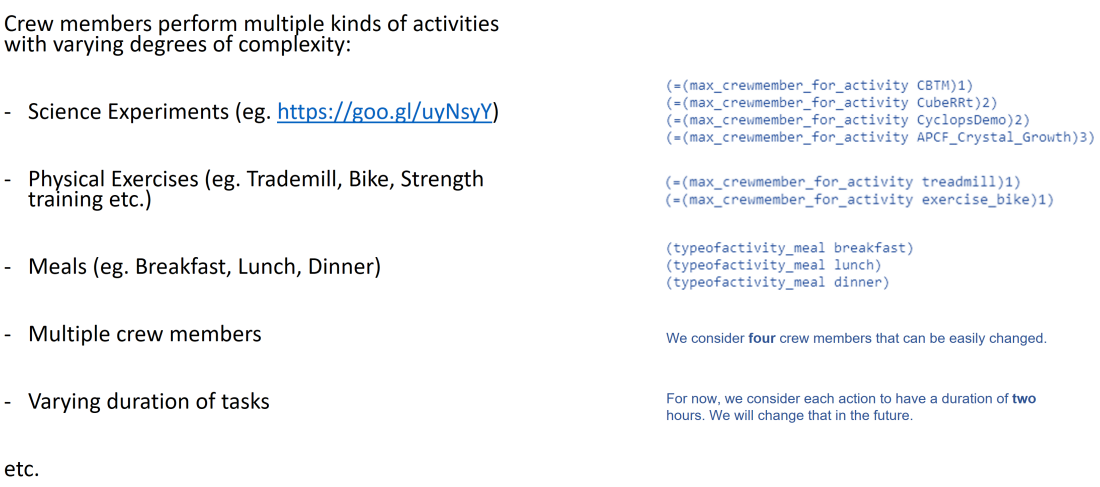


Figure 3.1: Front Panel

ISS. Some of these are described below–

Universal Actions are the tasks which are performed by all crew members as part of their daily routines. Some of these tasks are mandatory like *sleeping* or *breakfast* while most are optional like *exercise* or *daily meetings*. Often such a task could take up space and block calendars of crew members. As space and crew, both are constrained quantities so the planner must prioritize activities to arrive at a schedule maximizing mission objectives.

Science Experiments are performed by professional researchers, scientists, and experienced crew members. Again tasks of this kind take up space, and any crew member cannot deliver them. They need people with specialization. Therefore their requirement of personals is even more constrained than the Universal Actions. Examples include tasks requiring physical, biological or chemical study or tasks like *mass spectrometry*. Some of these tasks are needed to follow a precedence order.

Communication Tasks that send over the consequences of investigations

or gets information about new undertakings or information about updates to an existing venture from NASA's ground stations. These tasks often have temporal constraints. We may face a delay in completion due to communication or equipment failures. In such an occurrence a replanning with assistance is desired.

Maintenance Tasks involve repair and cleaning of equipment which are utilized in other activities. These may have constraints relating to individual members while ensuring that a single person is not always allocated maintenance tasks. Also, a precedence order is often observed in between such tasks and other tasks. One good example will be if an instrument is not repaired all pending experiments requiring that instrument would not be possible. Only a repair can generate the necessary effect to perform dependent tasks.

- **Goal State**

  The goal of a crew scheduling problem would be to arrive at a task schedule for the crew by maximally utilizing resources and abiding by constraints. The generated plan would be a crew schedule containing the list of actions with a time stamp and crew assignment. The goal of our decision support system is to assist the human decision maker in reaching his goal of building a task schedule.

- **Assumptions**

  Following assumptions were made while developing the domain model for `CAP`. It was done keeping in mind the computational abilities and the overall time complexity of the finding an optimal plan.

  1. The total number of hours a crew member can work in a day is ten.
  2. The maximum number of crew members is fixed at four.

3. The human model is available in PDDL to facilitate the generation of explanations.
4. A single crew member can perform only one task at a given instant.
5. The minimum unit of time to schedule a task is one hour.

3.2 System Architecture

In this section, we will describe the system architecture of our decision support system in detail. We will also describe all of the primary components of `CAP`.

3.2.1 Control Flow

Figure 3.2 shows the control flow of our application. The main component of our application are the following three components.

1. Interactive User Interface
2. Parsing and Back-end Services
3. Automated Planning Technology

The Interactive User Interface provides an intuitive and a domain optimized user interface for an end user to understand and participate in generating plans and schedules easily. The parsing and back-end services hosted on a different machine interact with the user interface through micro-services. This micro-service layer also act as a channel of communication to interact between the user interface and the automated planning technologies. Automated planning technology serves the role of the primary component of the decision support system that assists the human planner. Each of the components has been described in detail in the below sections.

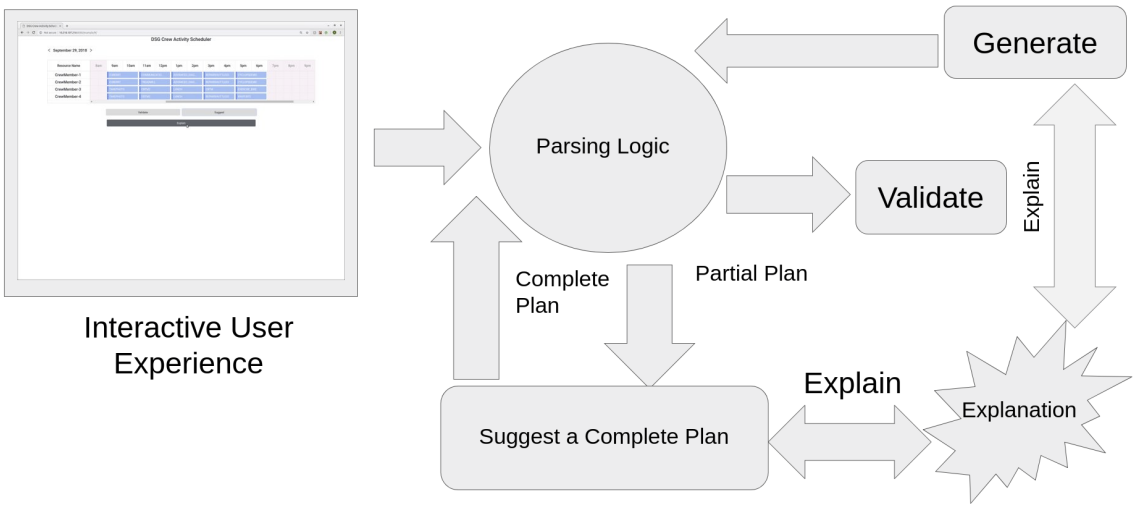


Figure 3.2: Control Flow Diagram of the CAP Interface.

3.2.2 User Interface

The user interface for CAP consist of multiple sections. Each of the section has a role to play in providing better user interface to the end user. Over ally, the intuitive UI design reduces cognitive load in the end user.

Front Panel

The front panel as illustrated in figure 3.3) consists of two sections one being the plan panel and the other being the button panel. It supports plan authoring by letting users drag their mouse over a time scale to create activities where they can specify the activity type and the astronauts who should be assigned to that activity. Other than creating a new task, a task (denoted by blue boxes in figure 3.3) can be moved around in a timeline along with extending the ends of the task. On hovering over a particular activity, it shows the details associated with the activity. Lastly, the button panel consists of three buttons– two of them (validate and suggest) show up by default and one (explain) that is displayed after the user asks for a suggestion to

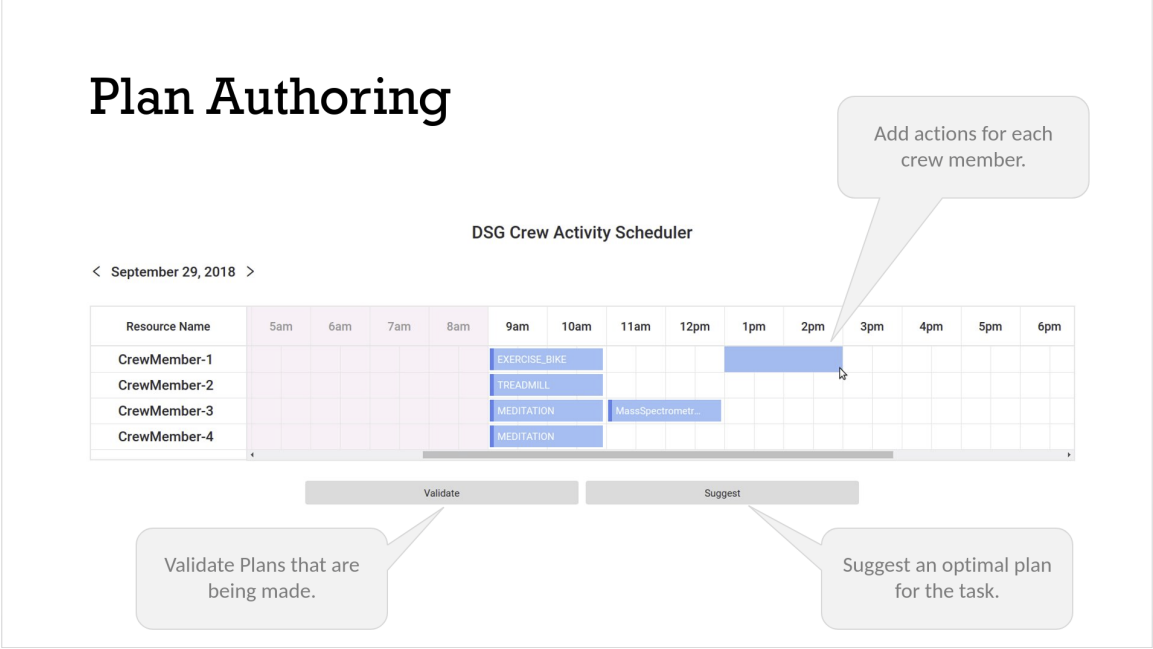


Figure 3.3: Front Panel

the system.

Task panel

The task panel in figure 3.4 is a form-based panel comes up when a user drags a section in a crew members timeline. Here the decision maker has to insert three other parameters including the name of the task, the type of the task and if it is a collaborative task he has to insert other crew members as well. Once inserted the task will appear on all of the selected crew members timelines. If there is any conflict, an error message would appear, and the task assignment would fail.

Dialogs

Dynamic UI dialog has been utilized to display appropriate status messages. In figure 3.10 below, we could see an explanation shown to the user. Dialogs are also used to show error messages or status messages.

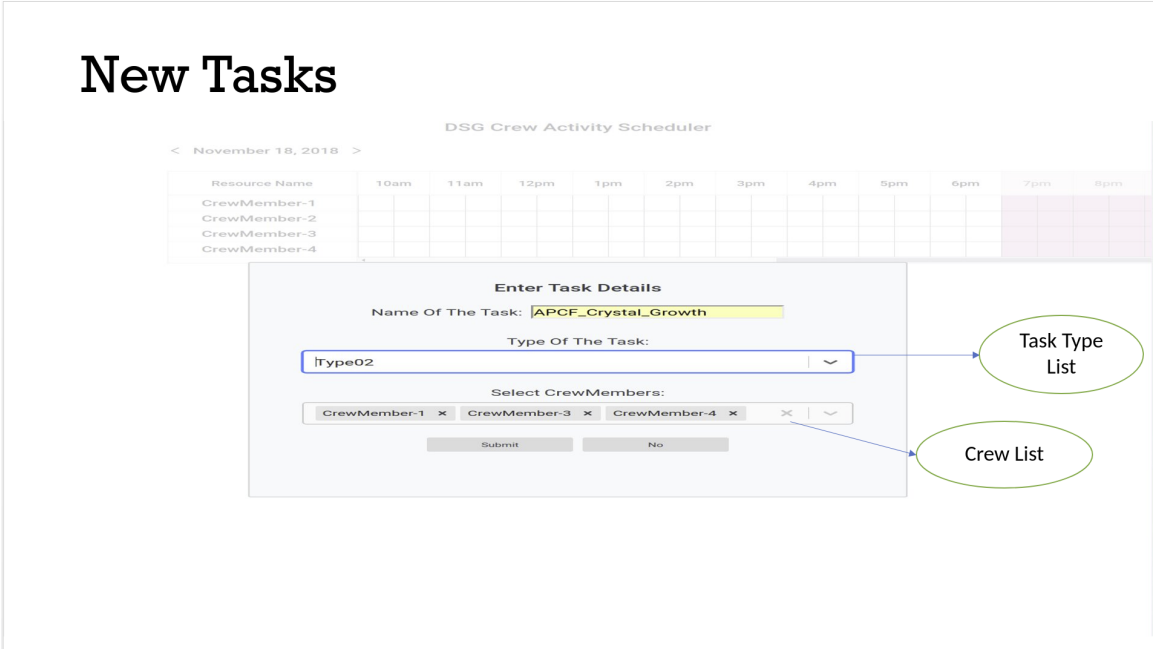


Figure 3.4: Task Panel

Interactive UI Components

User interface has been designed keeping in mind of design principles advocated in Shneiderman (2010). The UI also minimizes the chance of callback hell due to dangling JavaScript code, by using a variant of reactive programming paradigm as mentioned in Kambona *et al.* (2013). To give the user less cognitive load, we provide her with visual cues including highlighting the new section of plans once a partial plan has been updated, usage of visual cues is the use of a color-coded button in the dialog to emphasize upon the user the different situations emanating out of the assistant.

Software Stack - Frontend

For the development of user interface, we made use of following frameworks and programming languages:-

- HTML 5 and CSS - We used HTML 5 and CSS 3 to build the initial blueprint

of the system.

- REACT Framework - User interface logic was written in REACT using JSX based code. REACT was used for easy transferability of components across applications because of its original usage of the Component Application model.

- JavaScript - JavaScript was useful for events and state management. We did not implement a separate module for event and state management but rather used the internal functionalities of Javascript based DOM for event management.

- Web dev Server - Web dev server hosted the reactive front end after compiling down the ES6 version of code to ES5.

3.2.3 Parsing and Backend Architecture

Parsing layer acts as a bridge between the front end and back end by converting the UI objects to a PDDL plan or vice versa. For each functional use cases mentioned in section 5, we would utilize an associated backend component. The associated component would either validate, suggest or explain a plan to the user based on the user's request.

Software Stack - Backend

For the back-end and the parsing logic, we made use of following frameworks and programming languages:-

- Flask - Hosted on Flask Server which is a micro web framework written in Python. It does not require particular tools or libraries but rather use extensions to host services. We will host Python based micro services using JSON as our means of communication with the front end.

- Python - Python is used to develop the microservices, backend parsing logic and facilitating the Automated planning services.

As you can see separate servers are used for front-end and back-end to make sure that processing load can be distributed across multiple machines. The communication between front-end and back-end happened via micro-services through JSON.

3.2.4 Automated Planning Technology

Based on the planning formulation, we can now use off-the-shelf automated planning technologies to provide support during the Decision-making process.

- Plan Explanation Tool - Generate explanation by comparing the human and the robot's model from Chakraborti *et al.* (2017b). This tool considers two PDDL based models(one for the human planner, one for `CAP`). It then does an A-star search on the generated intermediate models, until it finds a plan optimal in an intermediate model. It then extracts the explanation as shown in Chakraborti *et al.* (2017b) and returns it to the end-user.

- Plan Recognition Tool - Plan Recognition Tool improvises the existing logic of the PR2 Plan by Ramírez and Geffner (2009) by including functionalities for numerical fluents for PDDL version 2.1 by Fox and Long (2003). Based on the existing plan supplied by the human planner, the tool will produce a domain where the supplied plan is already satisfied. It will then search for a plan in the corresponding domain model which will satisfy the end goal in the original model. The found plan is returned to be displayed to the human planner.

- Plan Validator Tool - Validation of a plan was done by utilizing the VAL tool by Howey *et al.* (2004a). This tool tries to satisfy the constraints in the PDDL

domain using the provided plan. Each action in the schedule is checked from the provided initial state for any constraint violation. If using the sequence of actions are valid, the plan validator returns success else it returns the violation.

- metric-FF Planner - To search for a new plan, we utilized the Metric-FF by Hoffmann (2003). Here, the planner will search for a schedule given the initial state, goal state, and the action model. This particular tool is useful when the user is unsure of how to create a schedule and asks CAP for a complete schedule.

3.3 USE CASES

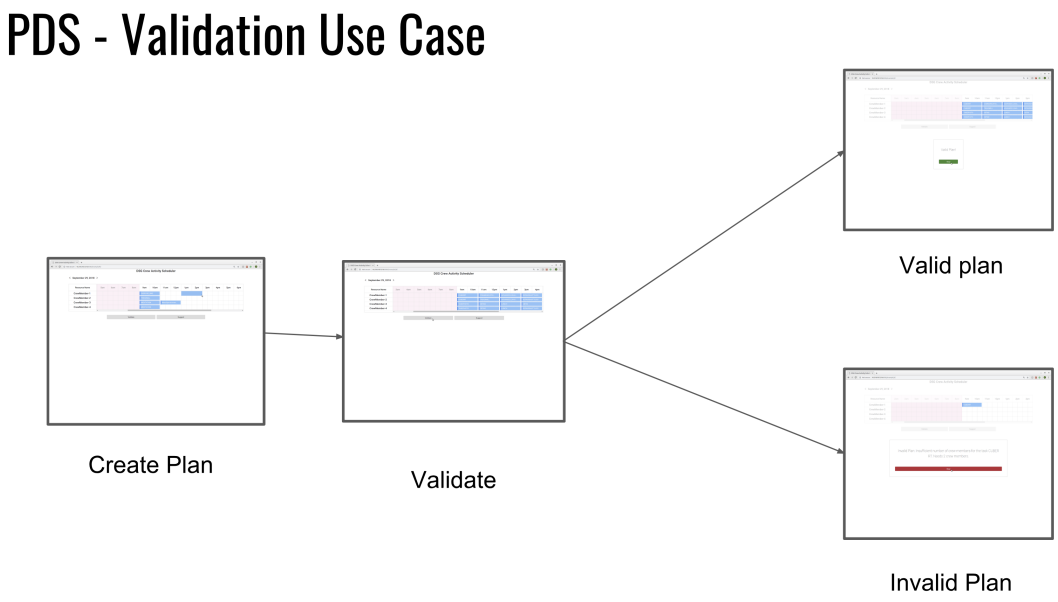


Figure 3.5: Control Flow for Validating a User's Partial Schedule and Reporting Constraint Violations.

In this section, we will try to demonstrate specific scenarios which a human planner may face while generating schedules for the crew member. Our use cases will highlight the working of CAP as a decision support system which can help the human planner in mission-critical scenarios.

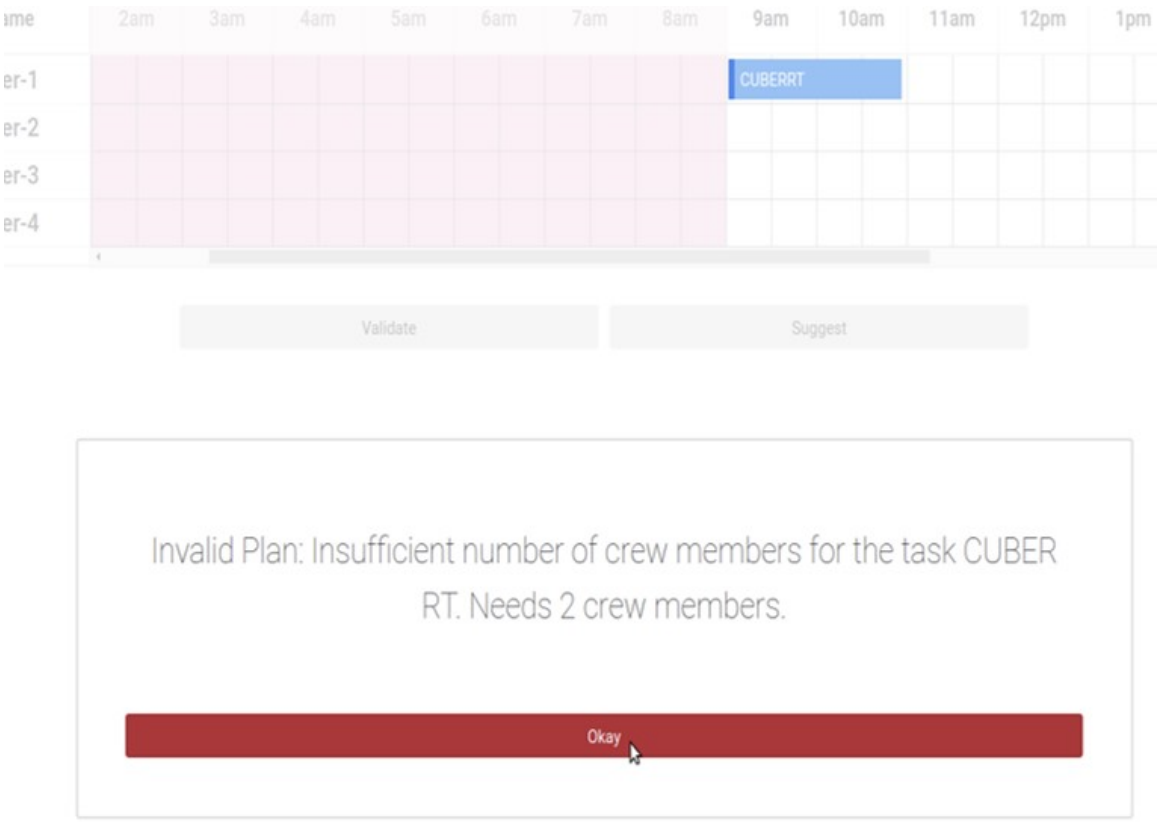


Figure 3.6: Validating a User's Partial Schedule and Reporting Constraint Violations.

3.3.1 Plan Validation

When a human makes a schedule by interacting with the user interface, they might not be aware of all of the constraints imposed by the domain or the individual astronauts. Thus, the partial plan π_p may not be realizable in practice. Plan validation using VAL Howey *et al.* (2004a) allows them to check if π_p is executable. If not, it can point out the constraints that are presently being violated, thereby helping the human on how to fix it.

The use case for plan validation has been shown in 3.5. In the initial screen, the human will make a plan using our hassle-free and interactive UI. Once done she will

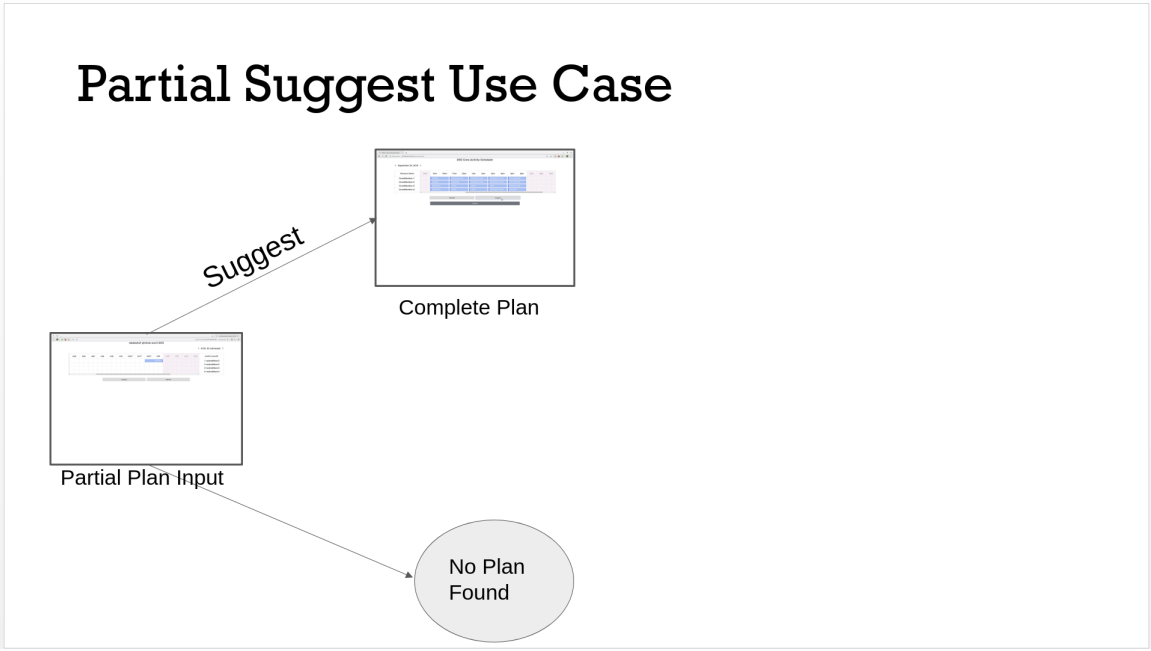


Figure 3.7: Control Flow for Suggesting the User a Completion for Their Partial Input Schedule.

ask the automated system to validate the schedule. The automated System validates the schedule for constraint violations and sends its output back to the home screen.

We can see in Figure 3.6, the user selects an action `CUBERRT` – this is as Science experiment which requires two crew members. The user having forgotten these constraints assigns the task to only a single crew member. Upon clicking 'validate', `CAP` reports this constraint violation.

3.3.2 Plan Suggestion

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 a re-implementation of Ramírez and Geffner (2009) for numerical fluents. Originally this approach was used for plan recognition, but we find it an effective tool that can be used for plan completion as well. Fortunately, this completion method may

sometimes fix some of the validation errors that existed in the π_p made by the human.

Suggestion can be of two types depending on the size of the partial plan. If the partial plan is size zero, that is no conditions has been added to the schedule by the user; we search for a schedule without any partial schedule using a planner like metric-FF. If the user has provided a partial task schedule with multiple tasks, then the back end system considers this partial task schedule while generating the final schedule. The use case for such a scenario is shown in 3.7

In Figure 3.8, 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. Note that not only does the planner come up with the entire schedule where the actions added by `CAP` has a red border, but also combines appropriate actions before the blue actions to overcome constraint violations of the human's initial plan.

3.3.3 Plan Explanation

Often, a plan suggested by the system is inexplicable to the human in the loop. In such a case, we allow the human to ask the planner for explanations and provide explanations based on the model reconciliation technique Chakraborti *et al.* (2017a). It is done by assuming a predefined model of the user (i.e., a user who is familiar with some of the constraints in the domain) and then providing a *minimal* subset of those constraints that support the suggested plan.

in 3.11 we could see that the TAKE_PHOTO essentially has a dependency on the *latch*. The *latch* has to be open during taking the photo. While the task of COMMUNICATION essentially has the completely reverse effect. Now, if this condition is not known to a novice user, she may get confused as to why the schedule was wrong. An explanation using Model Reconciliation can solve this confusion.

In Figure 3.10, as the human is surprised as to why a particular photo taking

Figure 3.8: Suggesting the User a Completion for Their Partial Input Schedule. Tasks Added by `CAP` Have a Left Red Border.

task is scheduled before a daily activity when clearly in their mind the priority of the latter task is more than the first one. The planner points out a particular effect of the former action that enables the latter action; thus, justifying the ordering. Thus, while explanations provide details of the domain that support a plan, validation points out constraints that invalidate a plan.

3.4 Limitations of `CAP`

We see a few limitations with our system `CAP` and we will point them out in this section.

- The decision support system is not needed to work or has limited application when humans are fully aware of the changes in the domain at all time. Hu-

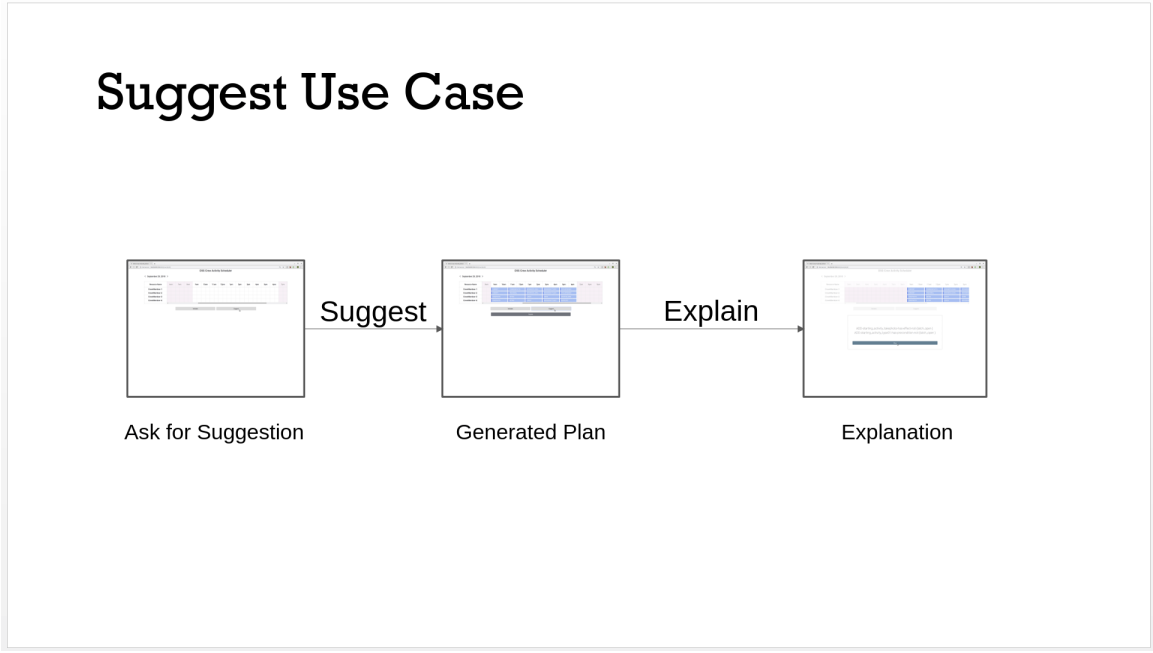


Figure 3.9: Control Flow for How Explanations Were Generated for a Particular Completion of a given Partial Plan.

mans will not seek assistance in such cases when they can come up with more straightforward plans. The humans may not utilize many of the features of the decision support system like explanation feature given that they may not either think of using it or forget about it. In that case, our system has no way to communicate back to the human to use these features

- The system is a simulated version with approximations. The real time system could possess more challenges.

.

3.5 Conclusion

In the current section, we have demonstrated `Crew Scheduling` which is an automated decision support system for crew scheduling. We have described all its features

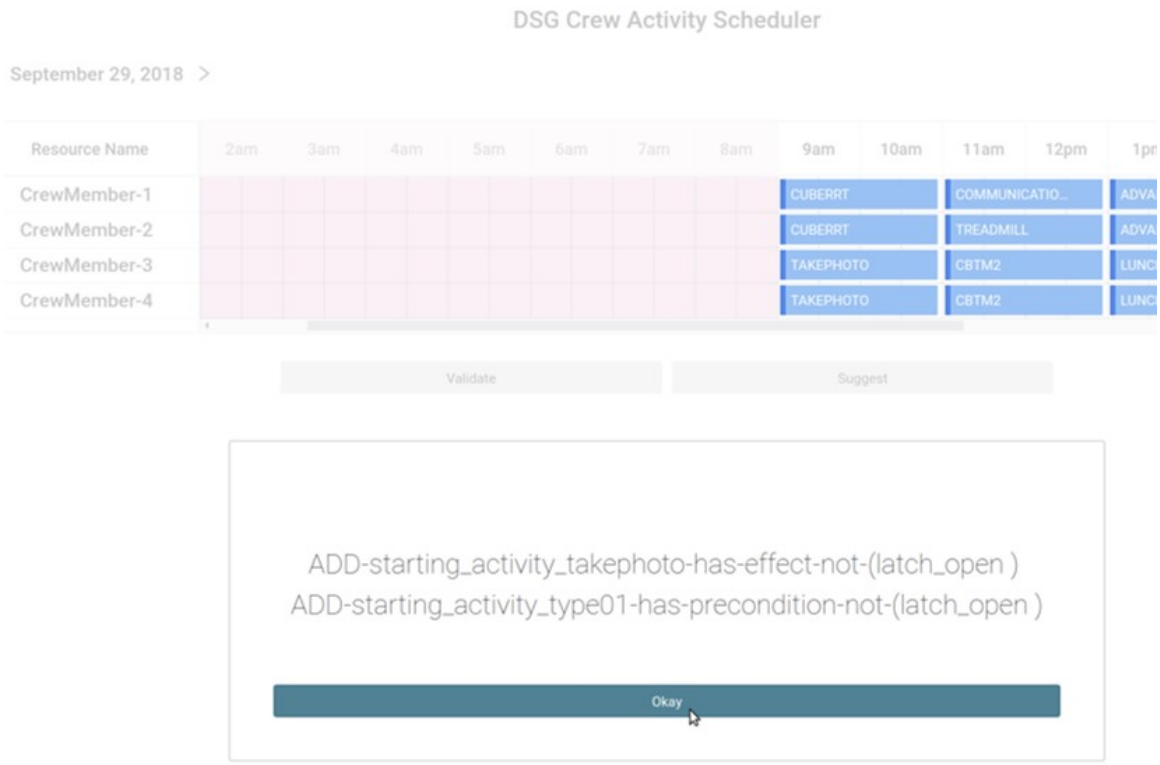


Figure 3.10: CAP Provides Explanations as to Why It Suggested a Particular Completion of a given Partial Plan.

and workings along with the software stack used for its development. Finally, we have also shown relevant use cases demonstrating scenarios which a mission planner may face during crew scheduling. We have demonstrated how our application can assist the human planner in such scenarios while adhering to principles of NDM and HILP.

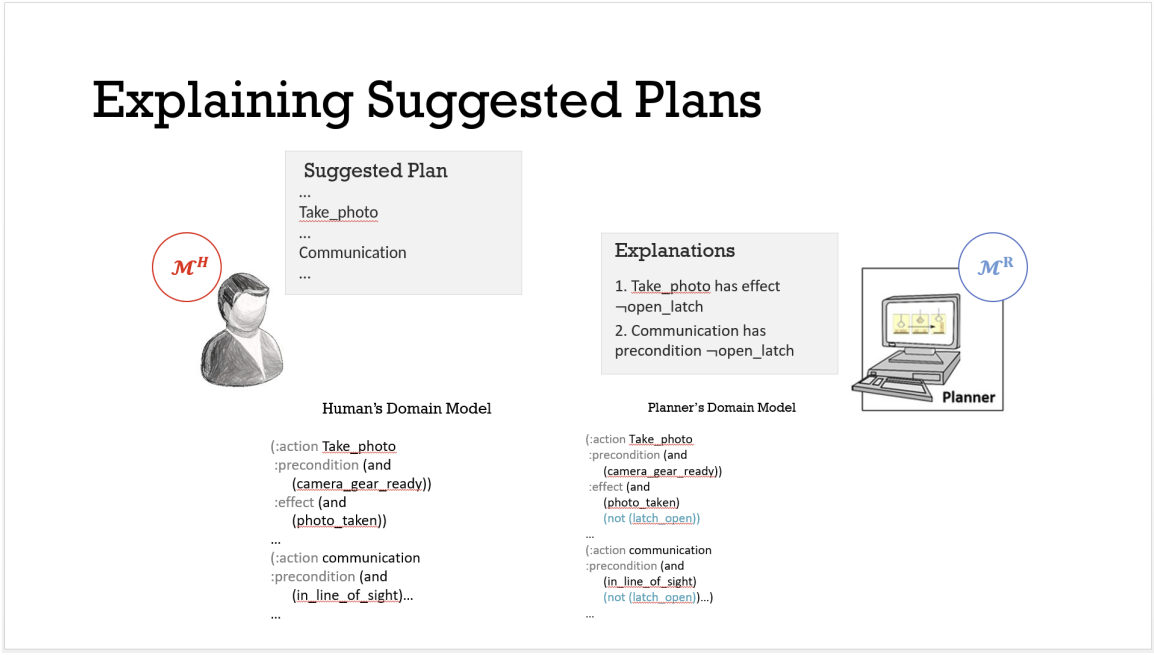


Figure 3.11: An Example Scenario Demonstrating Explanation Using Model Reconciliation.

Chapter 4

IPASS : EVALUATION OF THE EFFECTIVENESS OF AUTOMATED PLANNING FOR DECISION SUPPORT

One of the significant difficulties of the design of user studies in the area of automated decision support is access to domain experts who can verify the real usefulness of the decision support system. As stated earlier, It would have been nearly impossible to study the effectiveness of our framework built for crew scheduling because of unavailability of mission planners for a user study. In this section, we will try to construct a "plan of study" for graduate students of a university using the same principles of decision support highlighted in Chapter 3. The reason being we have easily accessible domain experts, i.e., graduate students for this evaluation. Moreover, this, in turn, allows us to perform a comprehensive study of key elements of decision support techniques using automated planning. The data gathered from these experiments were analyzed to determine to what extent automated task planning technologies proposed in the existing literature are useful as support systems for human-in-the-loop decision making.

4.1 Introduction

As mentioned in previous chapters, the theory of decision support is built around the idea of enabling human decision makers make decisions faster and more accurately with the added commitment to *never take the decision making away from the decision maker*. Although it is quite evident that the field of automated planning Ghallab *et al.* (2004) which aims to develop technologies that can compute a plan or a course of action given a problem description seems to be a perfect fit for this endeavor. However,

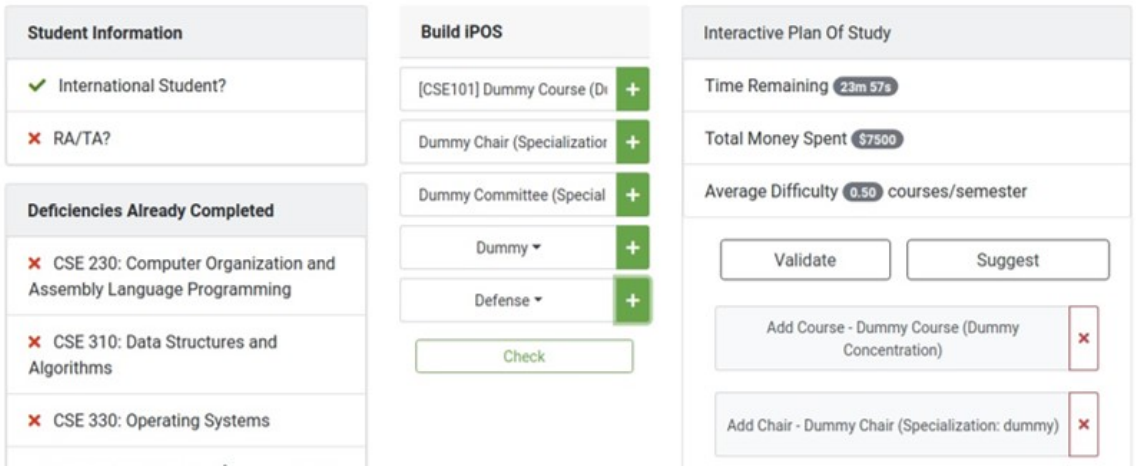


Figure 4.1: Illustration of the `Crew Scheduling` interface.

a user study evaluation can factually determine whether the synergy between the human planner and the automated planner has happened correctly or not. We can see whether the human can utilize the capacities of decision support to its fullest potential. The purpose of this paper is to do a case study of two key components of `RADAR` from Sengupta *et al.* (2017) and `CAP` from Chapter 3.

- the ability to validate a given plan for correctness.
- the ability to suggest a completion to a partial plan.
- demonstrate to what extent these components affect the the effectiveness of collaborative planning

4.2 `iPass` – System Overview

In this section, we describe the iPass planning domain and the components of the User Interface, which includes a feedback form to let the user answer subjective questions.

4.2.1 The iPass Domain and Interface

The task at hand to select subjects for ongoing and future semesters along with managing constraints of the domain is a scheduling problem. The domain was chosen because of the following reasons

- This task of course advising is challenging because we can see its presence in existing literature Khan *et al.* (2012) and also was recently presented as a competitive domain in the International Planning Competition Track (2018) as a benchmark domain;
- At the same time, we have domain experts, i.e., graduate students readily available for whom the domain is useful.

The interface (shown in Figure 4.1) has three panels –

- The panel on the left shows the relevant information of the student (e.g., information on deficiency courses, visa and residency status, research status, etc.)
- The central panel is utilized by the student to build the iPOS with various subpanels and forms to assist in building the IPOS. In this panel, a student can include adding courses, adding her specialization and her committee members.
- The panel on the right is an interactive interface to rework or rearrange the plan. It is often done to satisfy various constraints in the domain like the difficulty or an average number of courses a semester or total cost of tuition for the current plan.
- The last section is the decision support components. Decision support components like validate and suggest helps the user in coming up with a valid and desired plan. We will describe more about them in the next section.

4.2.2 Decision Support Components

The iPOS design problem is represented as a planning problem written in Planning Domain Definition Language (PDDL) by McDermott *et al.* (1998). It is quite similar to that of `CAP`. In this case, the planning problem would consist of the current state (which captures the student information), domain (which obtains the constraints of the area such as rules a student must follow), and the final goal that has to be achieved (a complete plan of study). The solution to a planning problem is the plan of study. Based on the planning formulation, we can now use off-the-shelf automated planning technologies to provide support during the planning process to the user who is constructing the plan of study. It is entirely analogous to the way we did with `CAP`.

- *Plan Validation* – Plan validation allows a student to ask the interface to check for correctness of a partially filled out iPOS. Again, We utilized VAL Howey *et al.* (2004b) to check if a sub-plan is executable in the compiled planning domain. For example, if a user attempts to violate rules or constraints of the domain like adding a normal course before completing deficiencies or adding a chair who is outside of the student's specialization area, the VAL will highlight this violation in a similar way it did in `CAP` for task assignments.

- *Plan Suggestion* – We will make use of an existing compilation from Ramírez and Geffner (2009) to complete a plan. The compiler here takes in the plan already constructed by the student, turns them into observations that must be produced in a compiled version of the original planning problem, and then solves it to ensure that parts of the iPOS already specified by the student are respected in the suggestions it is coming up with. One example would be If a student chooses their specialization and ask for suggestions that complete the rest of the course requirements and possible committee chair selection that

satisfies that specialization.

- *Plan Explanations* – The planner also provides explanations of its suggestions, only if requested by the user, using the technique of model reconciliation introduced in existing literature Chakraborti *et al.* (2017a).

4.2.3 Comparison with `CAP`

As mentioned earlier, we are evaluating `iPass` because it was not possible to do a user study on astronauts who are domain experts for `CAP` . We chose `iPass` because the nature of the problem for which assistance was required was similar. In both cases, we had an assignment problem. `CAP` had a task assignment problem for the crew where the human planner assigns various tasks for the crew considering various constraints while `iPass` had a course assignment problem for the students where the students themselves assign multiple subjects as per initial constraints. The background automated planning technologies we are using to assist is also the same. This way we will test the effectiveness of the same automated decision support which was being used in `CAP` . At the same time, the degree of assistance from the background decision assistant was also identical. Although, it is not a foolproof evaluation and the best evaluation can only happen only with a user study on astronauts, but our user study evaluated the necessary features for the effectiveness of automated decision support.

4.3 Aim of the Study

To determine the individual as well as the cumulative impact of the two decision support components, validation, and suggestion – we evaluated our interface in the below four conditions –

C_0 Both validation and suggestion capabilities are absent in the interface. The users do have to pass correctness bypassing all constraints of the domain themselves before they can submit.

C_1 Only validation capability is enabled in the interface.

C_2 Only suggestion capability is enabled in the interface.

C_3 Both validation and suggestion options are available in the interface.

Further, each participant, assigned to one of the study conditions C_i, performed the iPOS planning task twice. The student information was generated randomly in each case. So have two sub-conditions, C_i^1 and C_i^2 for each study condition C_i. We will judge our study based on the following parameters.

1. Planning performance for each study condition $P \rightarrow$ Planning performance is how the plan has performed relatively concerning various parameters like Time of completion for each study condition or the satisfaction of the final plan or interface by the end user. It is necessarily the performance of how the system behaved during an isolated evaluation.

2. The difference in time to completion between C_i^1 and C_i^2 is $\rightarrow \Delta T(C_i)$ is the difference in times of the two sub-conditions of the study for the end-user. The study of this parameter will help us to understand whether the user performed better with the knowledge of the interface, domain and hence using this information, whether the user can now do a better job at creating a plan, i.e., an IPOSS.

Given the above four conditions and two parameters, we are put forwarding the following hypothesizes concerning the parameters. These hypotheses are in sync with

the idea that the user can arrive at a better decision with active and consistent support.

H1. We hypothesize an increasing order of planning performance P as shown below –

$$P(C_0) < P(C_1), P(C_2) < P(C_3)$$

Note that, it is not expected that validation or suggestion functionalities by themselves are more useful than the other. 'Performance' can be interpreted in various forms as shown below:

H1a. The time to completion $T(C_i)$, $i = [0, 3]$ will follow the same order as above, e.g. $T(C_0) > T(C_1), T(C_2) > T(C_3)$.

H1b. The satisfaction with the final plan of study constructed will follow the **same** order.

H1c. The satisfaction with the feedback from the interface will follow the same order.

Note: The satisfaction measure in H1b and H1c checks how much the user is satisfied with the plan. If the user was pleased with the final plan of study, it means that the user was able to create an IPOS which according to them was excellent. If the user was satisfied with the feedback from the interface, it says that the user's experience from the interface was satisfactory. Now, the reason we think this improved planning performance is because we will ask the user to explain the IPOS at the end. So we expect them to produce a valid IPOS and not a namesake. Given that they have to complete the task in a time-constrained manner so if they are satisfied by the end product, we assume they had a better planning performance. As decision support's objective is to

assist human planners, so a measure showing human satisfaction is required to show how well our decision support system worked.

H2. The time to completion will reduce in all four conditions, however the reduction $\Delta T(C_i) = T(C_i^1) - T(C_i^2)$ will also follow the same order, i.e. –

$$\Delta T(C_0) < \Delta T(C_1) < \Delta T(C_2) < \Delta T(C_3)$$

We expect this to happen because,

1. In the later condition, users are provided relevant details of the domain as they construct a plan, and are thus expected to become more familiar with the domain.
2. Also, the user becomes more used to the interface the second time. So, the ability to navigate across the interface should improve.
3. This effect should be much more visible in C_2 and C_3 which provides explanations specifically for purposes of model reconciliation.

H3. The effects of support components on performance will be more pronounced and visible for subjects with less expertise, e.g., students who had not previously completed their iPOS.

4.4 Experimental Results

4.4.1 User Study - Evaluation Process

The user study was conducted in the following steps.

1. The study was conducted on the university premises.
2. Each subject was given $\$15$ for an hour of study when they used `iPass` software to make two IPOSs for the two conditions discussed above.

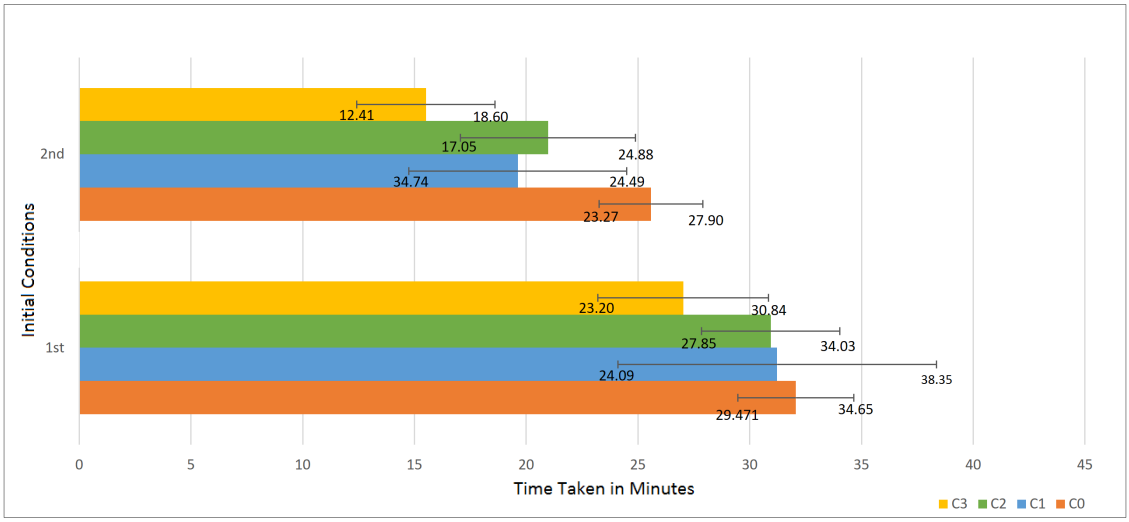


Figure 4.2: Average Time Taken (along with Confidence Interval of 95%) by a Participant to Complete the Two Parts of the Study for Each Condition c_i^1 and c_i^2.

3. At the start of the study, participants were informed that they would be asked to explain each IPOS with the hope that it will help them be more invested in the task as suggested by Mercier and Sperber (2011).

4. Then they were given a document explaining the planning domain and another document explaining the functionality of the elements in the interface.

5. Lastly, they were given 20 minutes to make each IPOS, after which they were presented with a feedback form.

4.4.2 User Study - Information about Participants

Out of a total of 56 participants, six were undergraduates, and the rest 50 were graduate students. We also had a group of experienced 18 participants who had submitted an IPOS before. The participants were distributed evenly among the four study conditions.

4.4.3 Hypothesis H1

H1a. The average time a participant took to complete the first and the second IPOS and submit their feedback [1] is shown in Figure 4.2. The data shows a significant improvement in performance with regards to time as one goes from C_0 to C_3 showing that the automated planning technologies have helped in improving the efficiency of the decision making process. We conducted t-tests to show the statistical significance of the results (i.e the effect on time to create the IPOS) for each of the IPOS when we move from C_0 to C_3. For the first IPOS, we see the effect of decision support on the time to create the IPOS in initial condition C_3 ($M = 27.02, SD = 7.29, N = 14$) from initial condition C_0 ($M = 32.06, SD = 4.95, N = 14$) which had no decision support. This effect is profound as shown from the results of t-test for a confidence interval of 95% and $d = 0.8$ is $t(23) = 2.17, p < 0.05$. For the second IPOS, we see the effect of decision support on the time to create the IPOS in initial condition C_3 ($M = 15.51, SD = 5.90, N = 14$) from initial condition C_0 ($M = 25.58, SD = 4.42, N = 14$) which had no decision support even more significantly than first IPOS. This effect is even more profound then the first IPOS as shown from the results of t-test for a confidence interval of 95% and $d = 1.93$ is $t(24) = 5.153, p < 0.001$.

Unfortunately, At the same time no measurable improvement in performance was found from (1) C_0 to C_1 or C_2 and (2) C_1 or C_2 to C_3 was observed. Thus, *hypothesis H1a was found to be only partly true.* Frequency of different functionalities that were used on the interface by the participants like the number of times the end users checked their solution for submission, and the number of times they rearranged, added or deleted actions in the plan are analyzed to study the behavior of the participants.

[1] Since the feedback was part of all the conditions, this is indicative of, even though not the actual, planning time.

This is shown in Figure 4.3. Our observations are stated below –

1. The average number of checks called was the highest in the case C_0. this was expected as it did not have any plan validation or suggestion support.

2. The average number of checks value is considerably less for the cases C_3 and C_1 which had validation feature.

3. Considering that the number of times a user validated their plan in conditions C_1 and C_3 (shown in Fig. 4.4), the use of check did not significantly have an impact on the time taken by the user to finish the IPOS.

4. The average number of times users rearranged actions is almost similar for all the conditions.

5. The average number of times a user clicked delete in the conditions C_2 and C_3, indicates that even though they clicked suggest four times in average in both of these two conditions (shown in Fig. 4.5), they were not satisfied with the plan generated by the automated planning system and hence edited (added and deleted) many actions.

H1b. In Figure 4.6, the answers of the users to the subjective statement *Q3: I am happy with the final Plan of Study* was plotted on the Likert Scale for all of the four conditions. It is observed that case C_0 has the least number of users who agreed (either agreed or strongly agreed) with the statement across all the four requirements. It is an expected scenario as many users were not even able to build a valid plan of study without any decision support in C_0. For C_1, six participants said they agreed with the statement Q3, and For C_2 and C_3, half of the participants were happy (i.e., either agreed or strongly agreed) with their plan of study, which is the highest across all the four conditions. However, in C_2 there was one participant who strongly

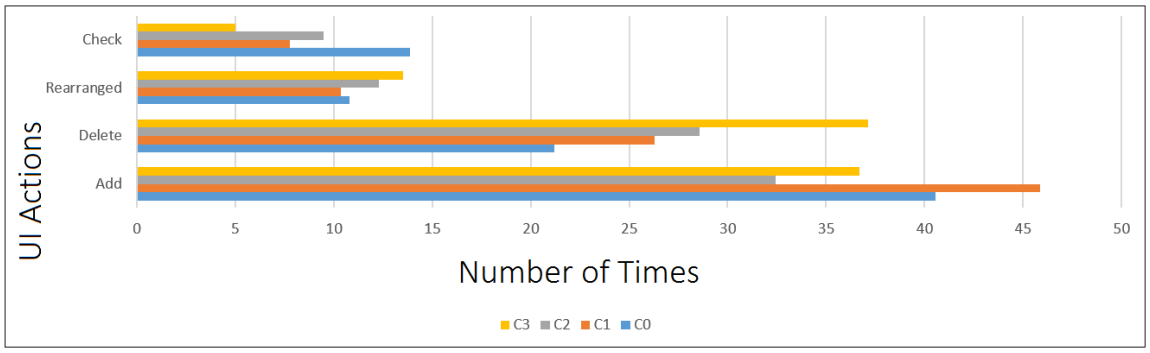


Figure 4.3: Average Number of times Participants Added, Deleted, Rearranged Courses or Clicked 'check' While Making an Ipos for All the Conditions c_i^1.

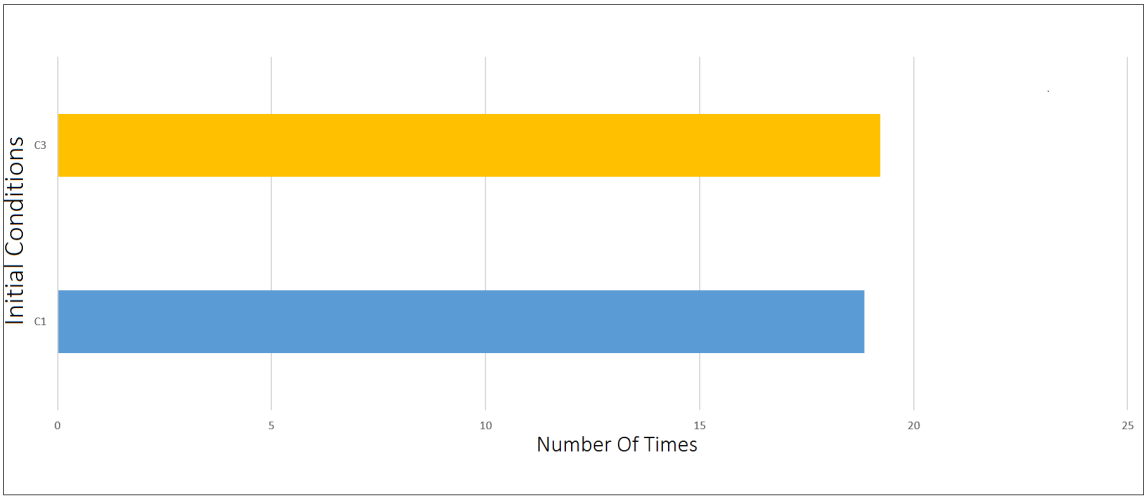


Figure 4.4: Average Number of times 'validate' Was Clicked in Condition c_1^1 and c_3^1.

disagreed with the statement, while for C_3 there were none. Thus, *the hypothesis H1b holds.* It was mentioned earlier that the participants erased and included more activities for the conditions C_2 and C_3 that can give plan recommendations. In the light of answers to the question Q3, it is intriguing to notice that even though the participants altered the proposed plan, having a ready-made plan accessible to them to bootstrap on for altering made them increasingly proficient, and also expanded

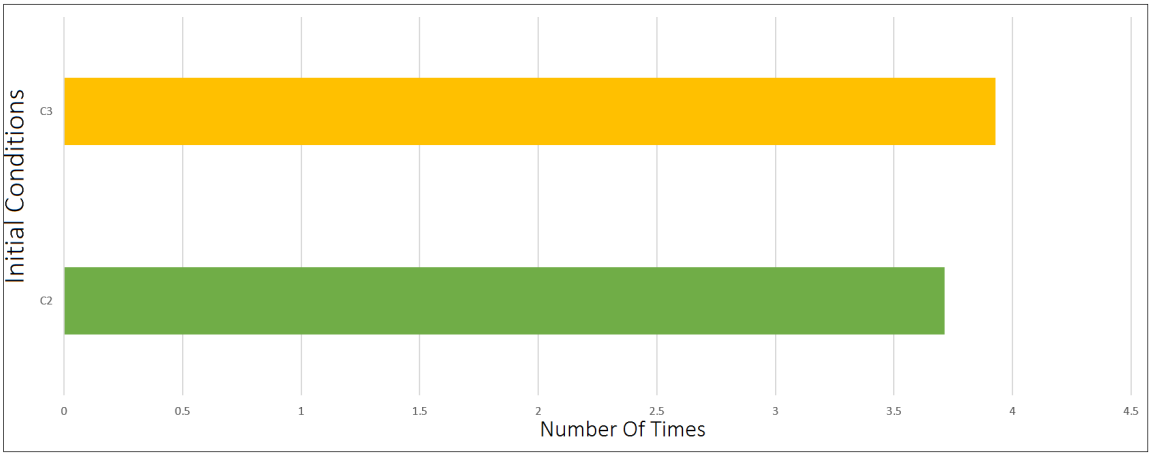


Figure 4.5: Average Number of Times 'suggest' Was Clicked in Conditions c_2^1 and c_3^1.

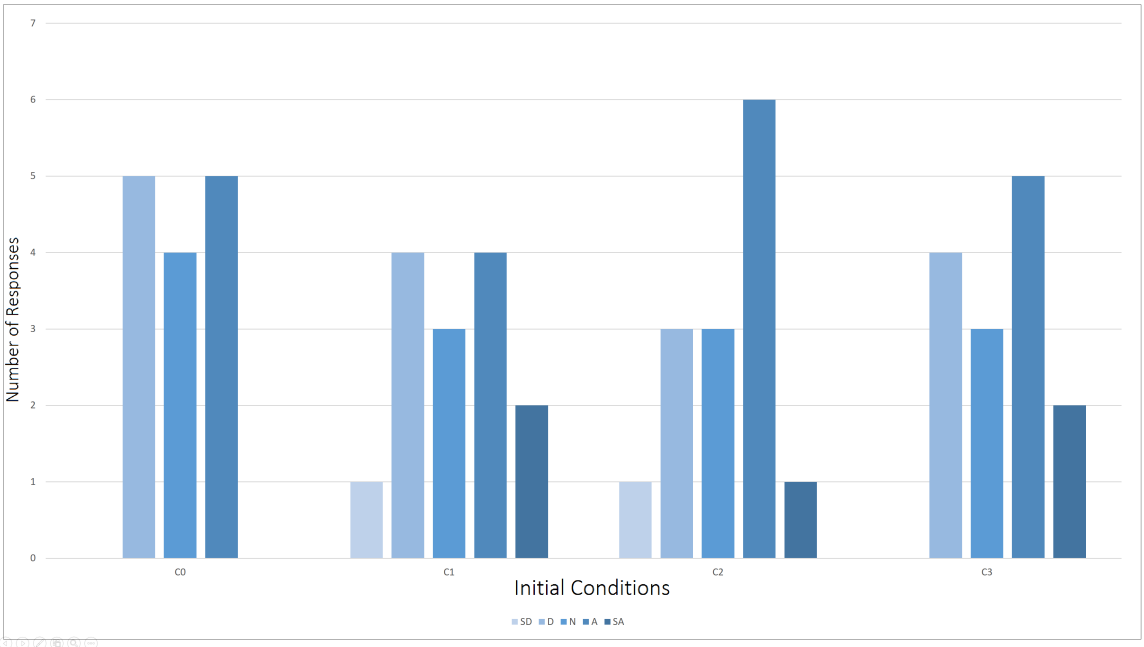


Figure 4.6: Average Score for Subjective Q3 for Conditions c_i^1.

their fulfillment.

H1c. Let n_{C_i} denote the number of participants who either agree or strongly agree with the statement *Q2: The feedback from the interface helped the iPOS making process.*, then from the figure 4.7 it is shown that the relation $n_{C_0} < n_{C_1}, n_{C_2} \leq n_{C_3}$

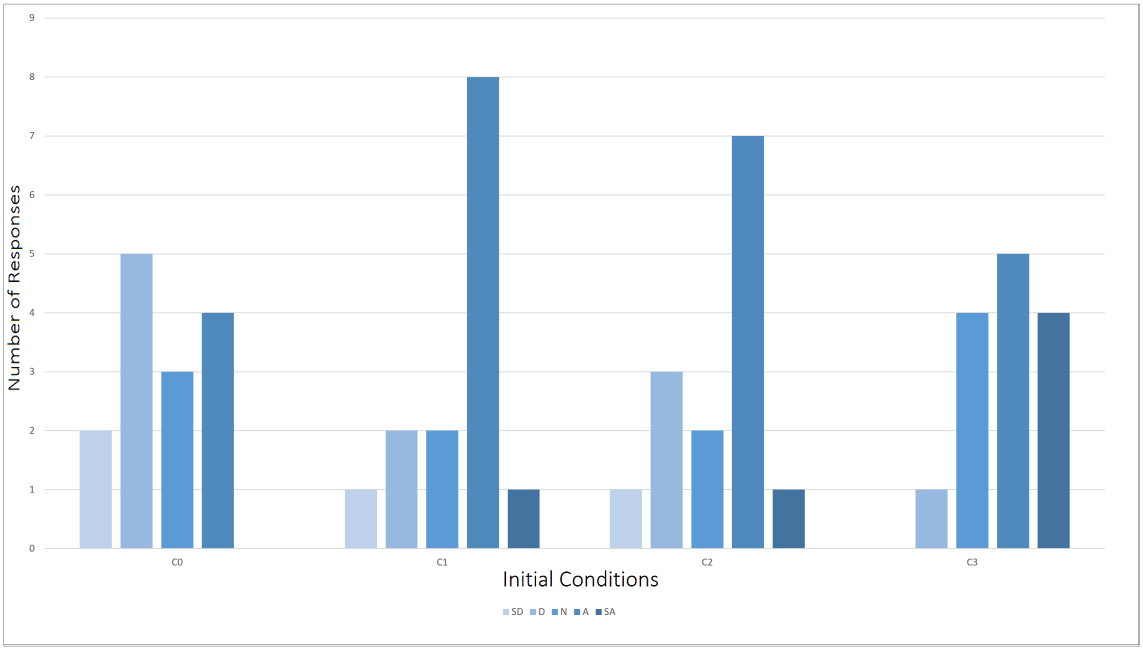


Figure 4.7: User Agreement Metrics for the Statement 'q2: The Feedback from the Interface Helped the Ipos Making Process' for Each Condition c_i^1.

holds. Although the equality holds for n_{C_1} and n_{C_3}. The number of people who strongly agreed to the statement was, by far, the highest for C_3. Thus, it is proven that *the hypothesis H1c holds.*

4.4.4 Hypothesis H2

The mean reduction in time has been plotted in completing the second IPOS after doing the first IPOS with `iPass` for all the four study conditions in Figure 4.8. We can deduce that

1. The lowest reduction in time for C_0 was expected as it shows that feedback given to the user by the decision support system helps them learn more about the domain model, thereby improving their performance in making the second IPOS.

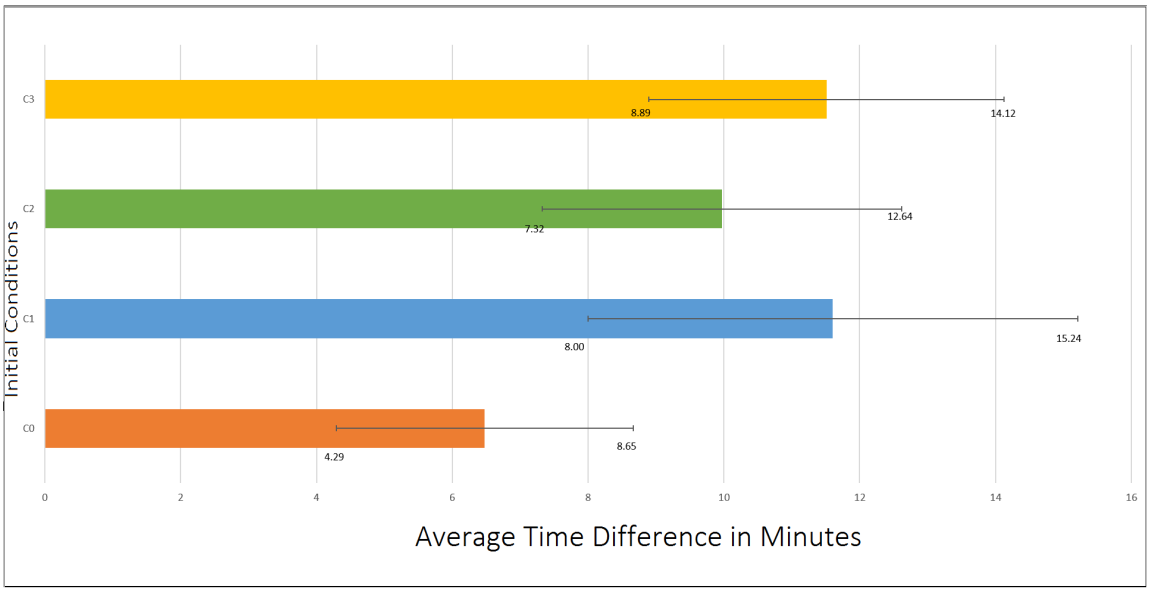


Figure 4.8: Time Difference $\delta T(C_i)$ Between Two Tasks c_i^1 and c_i^2 of ipos Planning for Every Condition c_i (along with Confidence Interval of 90%).

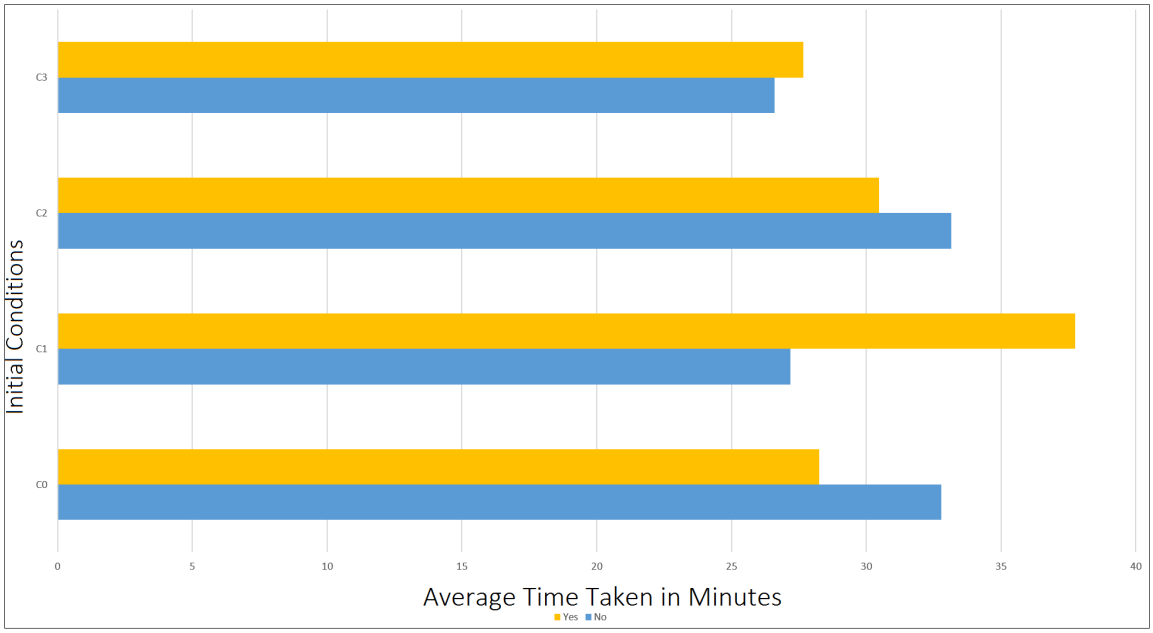


Figure 4.9: Time Taken by Experienced (in Yellow) and Non-experienced (in Blue) Users to Make the First Ipos (c_i^1).

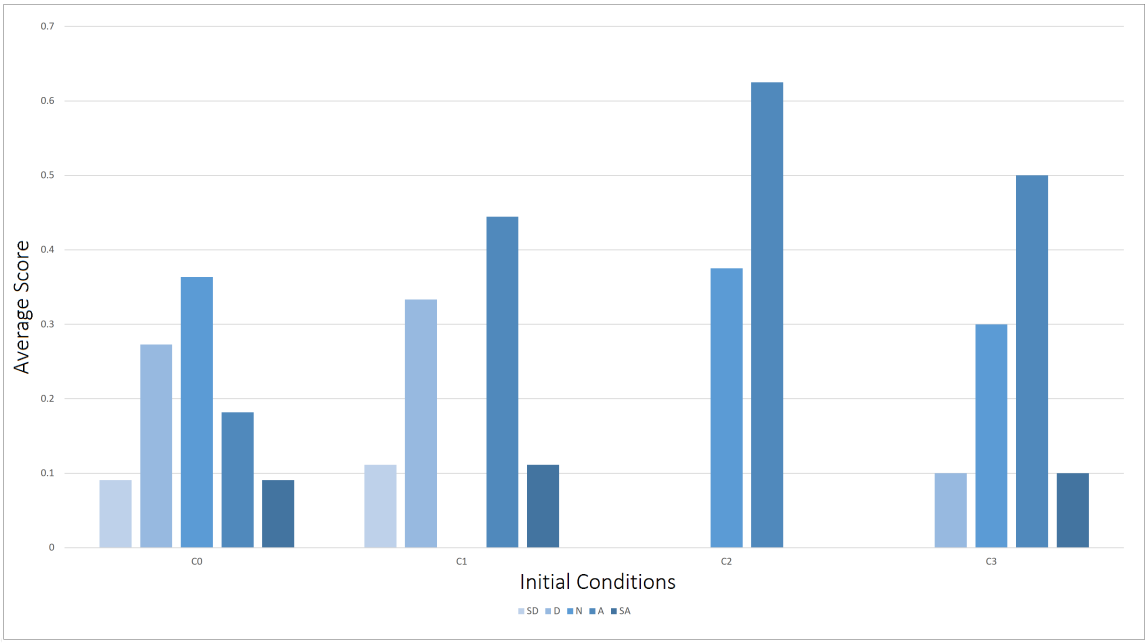


Figure 4.10: Feedback of Non-experienced Users about the Statement 'q1: The Planning Task Was Pretty Simple for Me'.

2. The highest decrease in time occurred for the conditions C_1 and C_3. We think that this decrease in time could have happened because of the presence of plan validation in both of these conditions. It informed the users about the reason behind each error they made while constructing the first IPOS. Hence it was effective in teaching the users about the actual domain. We conducted t-tests to show the statistical significance of the results (i.e the higher decrease in time while creating IPOS for the second time) for each of the IPOS when we move from C_0 to C_1 and C_0 to C_3. While moving from C_0 to C_1, we see the effect of decision support having an effect on the decrease in time while creating the second IPOS in initial condition C_1 $(M = 11.603, SD = 8.203, N = 14)$ from initial condition C_0 $(M = 6.47, SD = 4.97, N = 14)$ which had no decision support. This effect is profound as shown from the results of t-test for a confidence interval of 90% and $d = 0.76$ is $t(21) = -2.00, p < 0.1$.

While moving from C_0 to C_3, we see the effect of decision support having an effect on the decrease in time even more significantly while creating the second IPOS in initial condition C_3 ($M = 11.51, SD = 5.95, N = 14$) from initial condition C_0 ($M = 6.47, SD = 4.97, N = 14$) which had no decision support even more significantly. This effect is even more profound as shown from the results of t-test for a confidence interval of 90% and $d = 0.95$ is $t(25) = -2.42, p < 0.05$.

3. It was also hypothesized that the presence of plan explanations in C_2 and C_3 will reduce the time significantly because these explanations will teach the user more about the domain, thus reconciling their model differences. However, due to less usage of the Explanation features, we cannot come to a strong conclusion.

Hence, *H2 was also only found to be only partially true.* It supports the theory that the use of automated planning C_3 for decision support improves the efficiency of the human planner thereby reducing the time for making the second IPOS.

4.4.5 Hypothesis H3

It was noticed that the performance (time) was *not significantly better* for participants who had filled an IPOS before when compared to participants with no experience (Figure 4.9). Although the experienced participants did perform slightly better at C_0, C_1 and C_3. It was also noticed that for C_2, the users who had no prior experience performed better which was surprising. The reason we thought could be because the non-experienced group had prior conceptions about the rules of making an IPOS and thus, spent time making plans that appeared valid *in their model*, but were invalid in the `iPass` domain. With the presence of 'validate' in C_1, they might have ended up having to correct their partial plans multiple times, resulting in a long

time and worse performance.

The response of non-experienced users to the subjective question *Q1: The planning task was pretty simple for me* was plotted in Figure 4.10. Interestingly, the non-experienced users seemed to agree (or strongly agree) more with the statement in C_3 compared to C_0, which indicates that support features have contributed to a decrease in perceived difficulty of the task.

4.5 Limitations

Although our User Study was fairly successful in proving the importance of decision support. It too has its limitations. Larger sample size with diverse subjects other than computer science could have probably given us a better idea on some of our hypothesis. I am also mentioning a few of the limitation of our evaluation process itself.

1. The behavior mentioned in hypothesis H1 whereby we see a high number of adds and deletes is indicative that the planner decisively failed to capture general user preferences and we believe that the work on building explicable plans Zhang *et al.* (2016) will help improve the performance further for the cases C_2 and C_3.

2. While generating our hypothesis, we did not think of the scenario where the users will not make use of the explanation feature. Hence, we could not evaluate this feature properly as the users did not make use of it that much. If we had hypothesized about it and then made some suggestions accordingly during the user study in one set of experiments, we could have studied the effect of explanation on decision making to a greater extent.

3. The use of satisfaction measure for hypothesis H1 does not consider the bias

the user may be subjected to due to various psychological reason while filling the survey. Are they filling the survey accurately or are they just doing it for namesake? Also how much presence of a person, who was overlooking the test affects the user's answers. We did not consider these factors while considering the satisfaction measure as a parameter for planning performance.

4. H3 failed as users with no experience in generating IPOSs were also able to make IPOSs very efficiently. It could have happened because of their prior conceptions which led to valid plans in their model. If we had thought about it before the user study, we could have made changes to the domain model to make it unfamiliar yet recognizable to a set of participants for at least a set of experiments. We think that because of familiarity of the domain, people without any experience of building IPOS could still use it.

4.6 Conclusion

In conclusion, we found that two key decision support components – validation and suggestion – for human-in-the-loop planning tasks were, in general, helpful in improving the performance or satisfaction of the human decision-maker. From the written feedback, we noticed that 11 people asked for more feedback from the interface in C_0 (3 of whom mentioned suggestion feedback and 5 mentioned validation feedback) thus highlighting the role of the evaluated support components in the normative expectations of the user. Multiple users asked for computer generated suggestions in C_1, and for modeling of preferences on top of constraints in C_2 and C_3 thus corroborating patterns observed in data, and underlying their need in the future design of `CAP iPass` and decision support in general.

Chapter 5

CONCLUSION AND FUTURE WORK

In conclusion, we can say that we have successfully demonstrated the capabilities of an automated decision support system using the principles of Naturalized Decision Making(NDM), Human-In-The-Loop decision assistance(HILP) and Explainable AI based assistance. We set up two automated decision support systems `CAP` and `iPass` utilizing the basic tenets from all of the three disciplines. Both of our tools supported use cases which required decision support in their particular domains. Our objective of creating an interface to support decision making in crew scheduling domains was also achieved. We also did a user survey with `iPass` and presented its results suggesting the usefulness of automated decision support. We can say that with automated decision support especially with its validation and suggestion functionalities, the user experience was improved while making a decision. The integration of explanation generation through Model Reconciliation, in general, helped the end user to better gauze the system. For using explanation feature, the system does not assume anything and would leave it to the human planner to decide when it would need explanation. Even though the system can compute an explanation, it will only compute it when the human asks for it. We have also tested the performance of humans without explanation feature in initial condition C_0 and C_1 and initial condition C_2 and C_3 with the explanation feature. We do not see any heavy usage of the explanation feature for C_2 and C_3, so we cannot conclusively say that explanations were always helpful.

We did not do a deep dive into how much over-reliance can happen with our system. Our primary objective was to make sure that humans are assisted while they

have to decide on a high stake scenario. We also ensured that the humans are in the driver's seat throughout the decision-making process. As our decision support system is a factored decision support system with multiple varying features like plan validation, plan suggestion or explanation, we expect the user to interact more with the system. Also, we don't provide the end solution instead provide an assistance mechanism which can be accessed interactively by the human planner to reach to a solution. To develop a feature for studying over-reliance, we must consider diverse demography of domain experts. The reason being multiple social and cultural factors can affect the decision of a human to over-rely on automation. It is a separate problem of engendering trust with automation. Only with a use-case that targets to resolve this particular question on a user study can adequately provide evidence for over-reliance. This issue of over-reliance on automation was not investigated in our user study as the demography was not diverse enough. Though one crucial point was observed from the study, and that is even though explanation feature was available for IPOS we did not see a lot of its usage in the 2nd run which means the human-decision makers did not over-rely on it.

Although, in our current work we have performed Model Reconciliation and accordingly generated an explanation. A separate work of ours Sreedharan *et al.* (2019) does not even require a particular model of a human for reconciliation but instead, try to learn from the generated and presented explanations which has been better understood by the human actor. In the future, we can try to combine this particular approach of Model Reconciliation with `CAP` like system. Moreover, instead of searching over the model space for reconciliation, we can learn an explanatory model which works for a category of users.

In future, We will try to represent the problem of crew scheduling as a constraint program which will be challenging given occurrence of various tasks in the schedule

based on effect of certain other tasks. Other ideas to improve the system includes fact and foil based explanations and use of intelligible models to represent human preferences.

REFERENCES

Ai-Chang, M., J. Bresina, L. Charest, A. Chase, J.-J. Hsu, A. Jonsson, B. Kanefsky, P. Morris, K. Rajan, J. Yglesias *et al.*, "MAPGEN: Mixed-Initiative Planning and Scheduling for the Mars Exploration Rover Mission", IEEE Intelligent Systems (2004).

Caprara, A., M. Fischetti, P. L. Guida, P. Toth and D. Vigo, "Solution of large-scale railway crew planning problems: The italian experience", in "Computer-aided transit scheduling", pp. 1–18 (Springer, 1999).

Chakraborti, T., S. Sreedharan, Y. Zhang and S. Kambhampati, "Plan Explanations as Model Reconciliation: Moving Beyond Explanation as Soliloquy", in "IJCAI", (2017a).

Chakraborti, T., S. Sreedharan, Y. Zhang and S. Kambhampati, "Plan explanations as model reconciliation: Moving beyond explanation as soliloquy", arXiv preprint arXiv:1701.08317 (2017b).

Chakraborty, T., "Foundations of human-aware planning", URL `http://tchakra2.com/assets/files/dissertation.pdf` (2018).

Dechter, R. and D. Cohen, *Constraint processing* (Morgan Kaufmann, 2003).

Fern, A., S. Natarajan, K. Judah and P. Tadepalli, "A decision-theoretic model of assistance.", in "IJCAI", pp. 1879–1884 (2007).

Fox, M. and D. Long, "Pddl2. 1: An extension to pddl for expressing temporal planning domains", Journal of artificial intelligence research **20**, 61–124 (2003).

Freling, R., R. M. Lentink and A. P. 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).

Ghallab, M., D. Nau and P. Traverso, *Automated Planning: Theory and Practice* (Elsevier, 2004).

Hadfield-Menell, D., S. J. Russell, P. Abbeel and A. Dragan, "Cooperative inverse reinforcement learning", in "NIPS", (2016).

Hastie, T. J., "Generalized additive models", in "Statistical models in S", pp. 249–307 (Routledge, 2017).

Hoffmann, J., "The metric-ff planning system: Translating"ignoring delete lists"to numeric state variables", Journal of Artificial Intelligence Research **20**, 291–341 (2003).

Horvitz, E., "Principles of mixed-initiative user interfaces", in "Proceedings of the SIGCHI conference on Human Factors in Computing Systems", pp. 159–166 (ACM, 1999).

Howey, R., 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)", pp. 294–301 (2004a).

Howey, R., 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)", pp. 294–301 (2004b), URL `http://strathprints.strath.ac.uk/2550/`.

Johnston, M. D., "Spike: Ai scheduling for nasa", in "Sixth Conference on Artificial Intelligence for Applications", pp. 184–190 (IEEE, 1990).

Jussien, N., "e-constraints: Explanation-based constraint programming", in "CP01 Workshop on User-Interaction in Constraint Satisfaction", (2001).

Jussien, N. and V. Barichard, "The palm system: explanation-based constraint programming", in "Proceedings of TRICS: Techniques foR Implementing Constraint programming Systems, a post-conference workshop of CP", vol. 2000, pp. 118–133 (2000).

Kambhampati, S. and K. Talamadupula, "Human-in-the-loop planning and decision support", AAAI Tutorial (2015).

Kambona, K., E. G. Boix and W. De Meuter, "An evaluation of reactive programming and promises for structuring collaborative web applications", in "Proceedings of the 7th Workshop on Dynamic Languages and Applications", p. 3 (ACM, 2013).

Khan, O. Z., P. Poupart and J. P. Black, "Automatically Generated Explanations for Markov Decision Processes", in "Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions", pp. 144–163 (IGI Global, 2012).

Klein, G., "Naturalistic decision making", Human factors **50**, 3, 456–460 (2008).

Lou, Y., R. Caruana, J. Gehrke and G. Hooker, "Accurate intelligible models with pairwise interactions", in "Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining", pp. 623–631 (ACM, 2013).

Marquez, J. J., S. Hillenius, B. Kanefsky, J. Zheng, I. Deliz and M. Reagan, "Increasing crew autonomy for long duration exploration missions: self-scheduling", in "Aerospace Conference, 2017 IEEE", pp. 1–10 (IEEE, 2017).

Marquez, J. J., G. Pyrzak, S. Hashemi, K. McMillin and J. Medwid, "Supporting real-time operations and execution through timeline and scheduling aids", in "43rd International Conference on Environmental Systems", p. 3519 (2013).

McDermott, D., M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld and D. Wilkins, "Pddl-the planning domain definition language", (1998).

Mercier, H. and D. Sperber, "Why do humans reason? arguments for an argumentative theory", Behavioral and brain sciences **34**, 2, 57–74 (2011).

Miller, T., "Explanation in artificial intelligence: insights from the social sciences", arXiv preprint arXiv:1706.07269 (2017).

Miller, T., "Explanation in artificial intelligence: Insights from the social sciences", Artificial Intelligence (2018).

Parasuraman, R., T. B. Sheridan and C. 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).

Pinedo, M. L., *Scheduling: theory, algorithms, and systems* (Springer, 2016).

Ramírez, M. and H. Geffner, "Plan recognition as planning", in "IJCAI", (2009).

Ribeiro, M. T., S. Singh and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier", in "Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining", pp. 1135–1144 (ACM, 2016).

Rosenthal, S., S. P. Selvaraj and M. M. Veloso, "Verbalization: Narration of autonomous robot experience.", in "IJCAI", pp. 862–868 (2016).

Russell, S. and P. Norvig, *Artificial intelligence: a modern approach* (Prentice Hall, 2003).

Sengupta, S., T. Chakraborti, S. Sreedharan, S. G. Vadlamudi and S. Kambhampati, "Radar—a proactive decision support system for human-in-the-loop planning", in "2017 AAAI Fall Symposium Series", (2017).

Shneiderman, B., *Designing the user interface: strategies for effective human-computer interaction* (Pearson Education India, 2010).

Smith, D. E., "Planning as an iterative process.", (2012).

Smith, D. E., J. Frank and A. K. Jónsson, "Bridging the gap between planning and scheduling", The Knowledge Engineering Review **15**, 1, 47–83 (2000).

Sreedharan, S., A. Olmo, A. P. Mishra and S. Kambhampati, "Model-free model reconciliation", arXiv preprint arXiv:1903.07198 (2019).

Track, I. P. C. I. P., "Academic Advising Domain", `https://ipc2018-probabilistic.bitbucket.io/#domains` (2018).

Weld, D. and G. Bansal, "The challenge of crafting intelligible intelligence", Communications of ACM (2018).

Zhang, Y., S. Sreedharan, A. Kulkarni, T. Chakraborti, H. H. Zhuo and S. Kambhampati, "Plan Explicability for Robot Task Planning", in "RSS Workshop on Planning for Human-Robot Interaction: Shared Autonomy and Collaborative Robotics", (2016).

APPENDIX A

RAW DATA

```
(define (domain Nasa_strips_advanced_model_withSoftConstraints)
;; ===== ;; REQUIREMENTS ;; =====
(:requirements :strips :typing :fluents :negative-preconditions
:disjunctive-preconditions :equality :existential-preconditions
:quantified-preconditions :conditional-effects :adl)
;; ===== ;; TYPES ;; =====
(:types crew activity location - objects )
;; ===== ;; PREDICATES ;; =====
(:predicates
(daystarted)
(daycompleted)
(assign_crewmember ?crmem - crew ?wrt - activity)
(deactivatingactivityforcrew ?wrt - activity)
(typeofactivitynormal ?actvar - activity)
(typeofactivitytype01 ?actvar - activity)
(typeofactivitytype02 ?actvar - activity)
(typeofactivitytakephoto ?actvar - activity)
(inordercrew ?crew1 - crew ?crew2 - crew)
(currentcrewmember ?crew - crew)
(cannotassigncrew ?wrt - activity)
(busy_crewmember ?crew1 - crew)
(blocked_location ?loc - location)
(changelevel ?crew1 - crew)
(latch_open)
(activated_activity_forloc ?wrt - activity ?loc - location)
(activated_activity_forcrew ?wrt - activity)
(activitycompleted ?wrt - activity)
(activityinprogress)
(recentlyused ?crmem - crew)
(useonlyonceforcleanup)
(useforincreasingthecbustvalue)
)
;; ===== ;; FUNCTIONS ;; =====
(:functions
(rem_time_today_forall)
(rem_time_today ?crmem - crew)
(number_of_crew_members ?wrt - activity)
(max_crewmember_for_activity ?wrt - activity)
(decreaseintime)
(cannotbeusedtill)
(revecountcannotbeusedtill)
)
;; ===== ;; ACTIONS ;; =====
(:action starting_day
:parameters ()
```

```
:precondition (and(not(daystarted)))
:effect (and(daystarted))
) ;; (:action close_latch ;; :parameters ()
;; :precondition (and(not (daystarted)))
;; :effect (and (not(latch_open))) ;; )
;;(:action cleanrrecentlyusedtaskone
;; :parameters(?wrt - activity)
;; :precondition(and(¿(cannotbeusedtill)0)
;; (not(useonlyonceforcleanup))
;; (deactivatingactivityforcrew ?wrt))
;; :effect(and(useonlyonceforcleanup)
(decrease(cannotbeusedtill)(max_crewmember_for_activity ?wrt)))
;;)
(:action cleanrrecentlyusedtaskone
:parameters(?crmem - crew)
:precondition(and(=(revecountcannotbeusedtill)4))
:effect(and(not(recentlyused ?crmem)))
)
(:action cleanrrecentlyusedtasktwo
:parameters()
:precondition(and(=(revecountcannotbeusedtill)4))
:effect(and(decrease(revecountcannotbeusedtill)4))
)
(:action starting_activity_normal
:parameters (?wrt - activity ?loc - location )
:precondition(and
(daystarted)
(not(activitycompleted ?wrt))
;;(useonlyonceforcleanup)
(not(activityinprogress))
(typeofactivitynormal ?wrt)
(not(blocked_location ?loc)))
:effect(and (blocked_location ?loc)
(activityinprogress)
;;(not(useonlyonceforcleanup))
(activated_activity_forloc ?wrt ?loc)
(activated_activity_forcrew ?wrt))
)
(:action starting_activity_type01
:parameters (?wrt - activity ?loc - location )
:precondition(and
(daystarted)
;;(useonlyonceforcleanup)
(not(activitycompleted ?wrt))
(not(activityinprogress))
(typeofactivitytype01 ?wrt)
(not(blocked_location ?loc))
```

```
(not(latch_open)))
:effect(and (activityinprogress)
(blocked_location ?loc)
;;(not(useonlyonceforcleanup))
(activated_activity_forloc ?wrt ?loc)
(activated_activity_forcrew ?wrt))
)
(:action starting_activity_type02
:parameters (?wrt - activity ?loc - location )
:precondition(and
(daystarted)
;;(useonlyonceforcleanup)
(not(activitycompleted ?wrt))
(not(activityinprogress))
(typeofactivitytype02 ?wrt)
(not(blocked_location ?loc))
(not(latch_open)))
:effect(and
(activityinprogress)
(blocked_location ?loc)
;;(not(useonlyonceforcleanup))
(activated_activity_forloc ?wrt ?loc)
(activated_activity_forcrew ?wrt)) )
(:action starting_activity_takephoto
:parameters (?wrt - activity ?loc - location )
:precondition(and
(daystarted)
;;(useonlyonceforcleanup)
(not(activitycompleted ?wrt))
(not(activityinprogress))
(typeofactivitytakephoto ?wrt)
(not(blocked_location ?loc))
(latch_open))
:effect(and
(activityinprogress)
(blocked_location ?loc)
(not(latch_open))
;;(not(useonlyonceforcleanup))
(activated_activity_forloc ?wrt ?loc)
(activated_activity_forcrew ?wrt))
)
(:action assigning_current_crew_member
:parameters(?crmem - crew ?crmem1 - crew)
:precondition(and(currentcrewmember ?crmem)
(busy_crewmember ?crmem)
(inordercrew ?crmem ?crmem1)
) :effect(and
```

(not(currentcrewmember ?crmem))
(currentcrewmember ?crmem1))
)
(:action assigning_crew_members_activity
:parameters (?wrt - activity ?crmem - crew)
:precondition(and(activated_activity_forcrew ?wrt) (not(recentlyused ?crmem))
(not(busy_crewmember ?crmem))
(not(cannotassigncrew ?wrt))
(¿(rem_time_today ?crmem)0)
(¡(number_of_crew_members ?wrt)(max_crewmember_for_activity ?wrt)))
:effect(and (assign_crewmember ?crmem ?wrt)
(busy_crewmember ?crmem)
(decrease(rem_time_today ?crmem)(decreaseintime))
(decrease(rem_time_today_forall)(decreaseintime))
(increase(number_of_crew_members ?wrt)1)
))
;;Checks if all crewmember are assigned and frees the location.
(:action free_location_after_assignment
:parameters (?wrt - activity ?loc - location)
:precondition (and(activated_activity_forcrew ?wrt)
(=(number_of_crew_members ?wrt)(max_crewmember_for_activity ?wrt))
(activated_activity_forloc ?wrt ?loc)
(blocked_location ?loc))
:effect (and
(cannotassigncrew ?wrt)
(not(blocked_location ?loc))
(deactivatingactivityforcrew ?wrt)
(not(activated_activity_forloc ?wrt ?loc))
))
;;Loop over crew member(s)
and frees all of them one by one (:action free_individual_crew_members
:parameters (?wrt - activity ?crmem - crew)
:precondition (and(deactivatingactivityforcrew ?wrt)
(assign_crewmember ?crmem ?wrt)
(¿(number_of_crew_members ?wrt)0))
:effect (and (not(assign_crewmember ?crmem ?wrt))
(not(busy_crewmember ?crmem))
(increase(revecountcannotbeusedtill)1)
(decrease(number_of_crew_members ?wrt)1)
(recentlyused ?crmem)
))
(:action complete_activity
:parameters(?wrt - activity)
:precondition(and(deactivatingactivityforcrew ?wrt)
(activated_activity_forcrew ?wrt)
(=(number_of_crew_members ?wrt)0)
)

```
:effect(and
(not(activityinprogress))
(activitycompleted ?wrt)
(not(activated_activity_forcrew ?wrt))
(not(deactivatingactivityforcrew ?wrt))
)
)
(:action complete_day
:parameters()
:precondition(and(daystarted)
(not(activityinprogress))
(¡=(rem_time_today_forall)0)
)
:effect(and(not(daystarted))
(daycompleted)
) )
)
```

A.2 A Daily Plan Generated for CAP

```
(STARTING_DAY)
(STARTING_ACTIVITY_TYPE01 MASSSPECTROMETRYCAB02 LOCF)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
MASSSPECTROMETRYCAB02 CREWMEMBER-1)
(FREE_LOCATION_AFTER_ASSIGNMENT
MASSSPECTROMETRYCAB02 LOCF)
(FREE_INDIVIDUAL_CREW_MEMBERS
MASSSPECTROMETRYCAB02 CREWMEMBER-1)
(COMPLETE_ACTIVITY MASSSPECTROMETRYCAB02)
(STARTING_ACTIVITY_TYPE01
MASSSPECTROMETRYCAB03 LOCA)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
MASSSPECTROMETRYCAB03 CREWMEMBER-2)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
MASSSPECTROMETRYCAB03 CREWMEMBER-3)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
MASSSPECTROMETRYCAB03 CREWMEMBER-4)
(FREE_LOCATION_AFTER_ASSIGNMENT
MASSSPECTROMETRYCAB03 LOCA)
(FREE_INDIVIDUAL_CREW_MEMBERS
MASSSPECTROMETRYCAB03 CREWMEMBER-3)
(FREE_INDIVIDUAL_CREW_MEMBERS
MASSSPECTROMETRYCAB03 CREWMEMBER-4)
(FREE_INDIVIDUAL_CREW_MEMBERS
MASSSPECTROMETRYCAB03 CREWMEMBER-2)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-1)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-2)
```

(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-4)
(COMPLETE_ACTIVITY MASSSPECTROMETRYCAB03)
(STARTING_ACTIVITY_NORMAL XRAYEXP2 LOCA)
(CLEANRRECENTLYUSEDTASKTWO)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
XRAYEXP2 CREWMEMBER-1)
(FREE_LOCATION_AFTER_ASSIGNMENT XRAYEXP2 LOCA)
(FREE_INDIVIDUAL_CREW_MEMBERS XRAYEXP2 CREWMEMBER-1)
(COMPLETE_ACTIVITY XRAYEXP2)
(STARTING_ACTIVITY_NORMAL XRAYEXP LOCF)
(ASSIGNING_CREW_MEMBERS_ACTIVITY XRAYEXP CREWMEMBER-2)
(ASSIGNING_CREW_MEMBERS_ACTIVITY XRAYEXP CREWMEMBER-3)
(ASSIGNING_CREW_MEMBERS_ACTIVITY XRAYEXP CREWMEMBER-4)
(FREE_LOCATION_AFTER_ASSIGNMENT XRAYEXP LOCF)
(FREE_INDIVIDUAL_CREW_MEMBERS XRAYEXP CREWMEMBER-3)
(FREE_INDIVIDUAL_CREW_MEMBERS XRAYEXP CREWMEMBER-4)
(FREE_INDIVIDUAL_CREW_MEMBERS XRAYEXP CREWMEMBER-2)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-1)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-2)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-4)
(COMPLETE_ACTIVITY XRAYEXP)
(CLEANRRECENTLYUSEDTASKTWO)
(STARTING_ACTIVITY_TYPE01
COMMUNICATION2 LOCA)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
COMMUNICATION2 CREWMEMBER-2)
(FREE_LOCATION_AFTER_ASSIGNMENT COMMUNICATION2 LOCA)
(FREE_INDIVIDUAL_CREW_MEMBERS
COMMUNICATION2 CREWMEMBER-2)
(COMPLETE_ACTIVITY COMMUNICATION2)
(STARTING_ACTIVITY_TYPE02
REPAIRSHUTTLE03 LOCF) (ASSIGNING_CREW_MEMBERS_ACTIVITY
REPAIRSHUTTLE03 CREWMEMBER-1)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
REPAIRSHUTTLE03 CREWMEMBER-3)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
REPAIRSHUTTLE03 CREWMEMBER-4)
(FREE_LOCATION_AFTER_ASSIGNMENT
REPAIRSHUTTLE03 LOCF)
(FREE_INDIVIDUAL_CREW_MEMBERS
REPAIRSHUTTLE03 CREWMEMBER-4)
(FREE_INDIVIDUAL_CREW_MEMBERS
REPAIRSHUTTLE03 CREWMEMBER-3)
(FREE_INDIVIDUAL_CREW_MEMBERS
REPAIRSHUTTLE03 CREWMEMBER-1)

(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-1)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-2)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-4)
(COMPLETE_ACTIVITY REPAIRSHUTTLE03)
(CLEANRRECENTLYUSEDTASKTWO)
(STARTING_ACTIVITY_TYPE02 REPAIRSHUTTLE3 LOCA)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
REPAIRSHUTTLE3 CREWMEMBER-2)
(FREE_LOCATION_AFTER_ASSIGNMENT
REPAIRSHUTTLE3 LOCA)
(FREE_INDIVIDUAL_CREW_MEMBERS
REPAIRSHUTTLE3 CREWMEMBER-2)
(COMPLETE_ACTIVITY REPAIRSHUTTLE3)
(STARTING_ACTIVITY_TYPE01 COMMUNICATION LOCF)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
COMMUNICATION CREWMEMBER-3)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
COMMUNICATION CREWMEMBER-4)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
COMMUNICATION CREWMEMBER-1)
(FREE_LOCATION_AFTER_ASSIGNMENT
COMMUNICATION LOCF)
(FREE_INDIVIDUAL_CREW_MEMBERS
COMMUNICATION CREWMEMBER-4)
(FREE_INDIVIDUAL_CREW_MEMBERS
COMMUNICATION CREWMEMBER-1)
(FREE_INDIVIDUAL_CREW_MEMBERS
COMMUNICATION CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-1)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-2)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-4)
(COMPLETE_ACTIVITY COMMUNICATION)
(CLEANRRECENTLYUSEDTASKTWO)
(STARTING_ACTIVITY_TYPE02 BREAKFAST2 LOCA)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
BREAKFAST2 CREWMEMBER-1)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
BREAKFAST2 CREWMEMBER-4)
(FREE_LOCATION_AFTER_ASSIGNMENT BREAKFAST2 LOCA)
(FREE_INDIVIDUAL_CREW_MEMBERS BREAKFAST2 CREWMEMBER-1)
(FREE_INDIVIDUAL_CREW_MEMBERS BREAKFAST2 CREWMEMBER-4)
(COMPLETE_ACTIVITY BREAKFAST2)
(STARTING_ACTIVITY_NORMAL REPAIRSHUTTLE01 LOCA)
(ASSIGNING_CREW_MEMBERS_ACTIVITY
REPAIRSHUTTLE01 CREWMEMBER-2)

(ASSIGNING_CREW_MEMBERS_ACTIVITY
REPAIRSHUTTLE01 CREWMEMBER-3)
(FREE_LOCATION_AFTER_ASSIGNMENT REPAIRSHUTTLE01 LOCA)
(FREE_INDIVIDUAL_CREW_MEMBERS
REPAIRSHUTTLE01 CREWMEMBER-2)
(FREE_INDIVIDUAL_CREW_MEMBERS
REPAIRSHUTTLE01 CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-1)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-2)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-3)
(CLEANRRECENTLYUSEDTASKONE CREWMEMBER-4)
(COMPLETE_ACTIVITY REPAIRSHUTTLE01)
(CLEANRRECENTLYUSEDTASKTWO)
(COMPLETE_DAY)