Taylor & Francis
Taylor & Francis Group

HUMAN–COMPUTER INTERACTION

Human–Computer Interaction

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/hhci20

RADAR: automated task planning for proactive decision support

Sachin Grover, Sailik Sengupta, Tathagata Chakraborti, Aditya Prasad Mishra & Subbarao Kambhampati

To cite this article: Sachin Grover, Sailik Sengupta, Tathagata Chakraborti, Aditya Prasad Mishra & Subbarao Kambhampati (2020) RADAR: automated task planning for proactive decision support, Human–Computer Interaction, 35:5-6, 387-412, DOI: 10.1080/07370024.2020.1726751

To link to this article: https://doi.org/10.1080/07370024.2020.1726751

HUMAN–COMPUTER INTERACTION
2020, VOL. 35, NOS. 5–6, 387–412
https://doi.org/10.1080/07370024.2020.1726751

Taylor & Francis
Taylor & Francis Group

RADAR: automated task planning for proactive decision support

Sachin Grover[a], Sailik Sengupta[a], Tathagata Chakraborti[b], Aditya Prasad Mishra[a], and Subbarao Kambhampati[a]

[a]Yochan Lab, Arizona State University, Tempe, AZ, USA; [b]IBM Research AI, AI interaction group

ABSTRACT

Proactive Decision Support aims at improving the decision making experience of human decision-makers by enhancing the quality of the decisions and the ease of making them. Given that AI techniques are efficient in searching over a potentially large solution space (of decision) and finding good solutions, it can be used for human-in-the-loop scenarios such as disaster response that demand naturalistic decision making. A human decision-maker, in such scenarios, may experience high-cognitive overload leading to a loss of situational awareness. In this paper, we propose the use of automated task-planning techniques coupled with design principles laid out in the Human-Computer Interaction (HCI) community for developing a proactive decision support system. To this extent, we highlight the capabilities of such a system RADAR and briefly, describe how automated planning techniques help us in providing the varying degrees of assistance. To evaluate the effectiveness of the different capabilities, we conduct ablation studies with human subjects on a synthetic environment for making an interactive plan of study. We found that planning techniques like plan validation and suggestions help to reduce planning time (objective metrics) and improves user satisfaction (subjective metrics) compared to expert human planners without any support.

ARTICLE HISTORY
Received 21 June 2019
Revised 4 February 2020
Accepted 4 February 2020

KEYWORDS
Proactive Decision Support; Automated Task Planning; HCI Design Theory

1. Introduction

Human-in-the-loop planning (HILP) (Kambhampati & Talamadupula, 2015) is a requirement in many present-day complex decision making and planning environments. In this paper, we consider a case of HILP where the human responsible for making the decisions in a complex scenario is supported by an automated planning system. High-level information fusion that characterizes complex long-term situations and supports the planning of effective responses is considered the greatest need in crisis-response situations (Laskey, Marques, & da Costa, 2016). Indeed, automated planning-based proactive support was preferred by humans involved in teaming with robots (Zhang, Narayanan, Chakraborti, & Kambhampati, 2015) where the cognitive load of the involved subjects involved was observed to have been reduced (Narayanan, Zhang, Mendoza, & Kambhampati, 2015).

Traditional planning techniques have focused on end-to-end plan generation rather than proactive support. Although there has been some work recently to make these techniques *human-aware* that try to account for human activities and intents while constructing the plans (Zhang et al., 2016), such as generating explicable (Zhang et al., 2016) or legible plans (Dragan, Lee, & Srinivasa, 2013), these works still focus on complete plan generation. Thus, none of these techniques can be directly adapted to providing decision support. In this work, we investigate the extent to which an automated planner can support the human's decision-making process, despite not having access to the complete domain and preference models, while the humans remain in charge of the process. This is appropriate in many cases, where the human-in-the-loop is ultimately held responsible for the plan execution and its results. This is in contrast to earlier work on systems such as TRAINS (Allen,

CONTACT Sachin Grover sachin.grover@asu.edu Yochan Lab, Arizona State University, Tempe, AZ, USA.

1994), MAPGEN (Ai-Chang et al., 2004) and (Kim, Banks, & Shah, 2017) where the planner is in the driver's seat, with the humans "advising" the planner. Thus, our work is distinct from them on *mixed-initiative* planning where humans enter the land of automated planners, manipulating their internal search process – here, the planners enter the land of humans.

An important complication arises because the planner and the human can have different (even complementary) models of the same domain or knowledge of the problem at hand (as shown in Figure 1). In particular, humans might have additional knowledge about the domain or the plan preferences that the automated planner is not privy to. This means that plan suggestions made by the automated planner may not always *make sense* to the human in the loop, i.e., appear as sub-optimal in their model. This is an ideal opportunity for the system to provide model updates as explanations (Chakraborti, Sreedharan, Zhang, & Kambhampati, 2017) and reconcile the models through iterative feedback from the human during the plan generation phase.

The extent to which a planner can be used for decision support is largely dependent on the nature of the model that is available. For example, if we have an incomplete model that often occurs in many mixed-initiative settings (Smith, 2012); then, an automated support component can use the incomplete model to complete or critique existing plans (Manikonda, Chakraborti, Talamadupula, & Kambhampati, 2017). Keeping this in mind, in the current paper we focus on scenarios which come with more well-defined protocols or domain models, and illustrate how off-the-shelf planning techniques may be leveraged to provide various degrees of decision support (as opposed to complete automation). We believe such technologies will be helpful in naturalistic decision making scenarios such as disaster response where the cognitive overload of the human can negatively affect the quality of decision making.

Human-Computer Interaction (HCI) is thought to have developed as a sub-field in three different areas – management information systems, computer science, and human factors (Grudin, 2011). While human factors have evolved to understand the behavioral effects of agents on different interfaces, management information systems, and computer science worked on various ways of

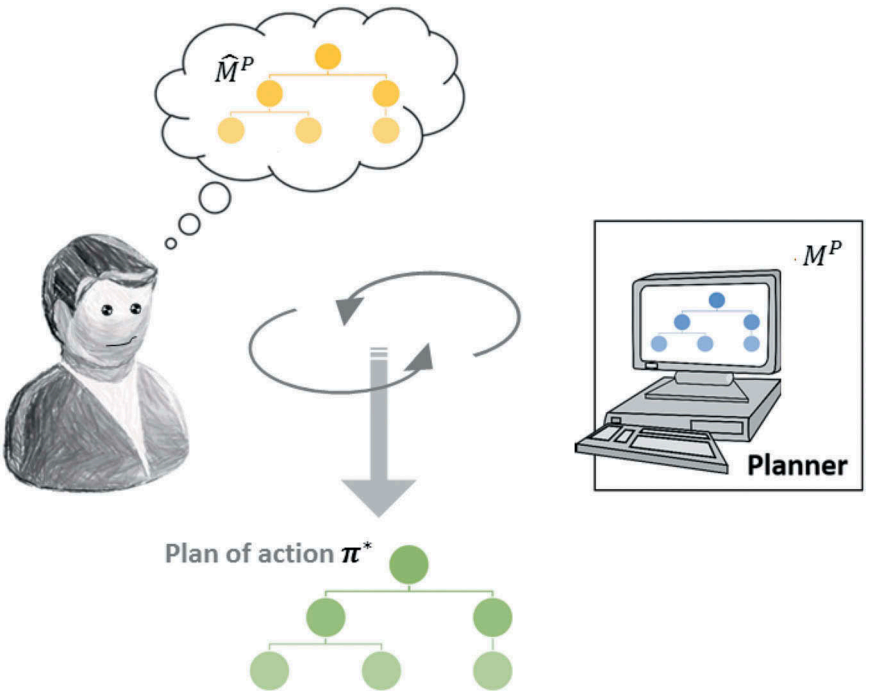


Figure 1. Planning for decision support must consider difference in models between the planner and the human.

designing these interactions. In the past, two of the most common methods of interactions were – direct manipulation and interface agents (Shneiderman & Maes, 1997). While direct manipulation occurs when the interface changes only based on the user's instructions, interface agents are assumed to possess more intelligence and adapt by themselves, behaving like a collaborator (Maes, Shneiderman, & Miller, 1997). For example, software for classifying news or e-mail as relevant or not, in the context of a specific user, can be thought of as an intelligent software agent. In this paper, we look at a specific intelligent software agent that provides decision support to a user and reduces their cognitive and information overload for sequential decision-making tasks.

Earlier works have applied the principles of Human-Human Interaction (HHI) for designing a collaborative disclosure interface (Lesh, 2004) rather than motivating the design of decision support software with principles in Human-Computer Interaction (HCI) directly. This work, to our knowledge, is the first to propose a proactive decision support (PDS) system RADAR following some of the design principles laid out in the literature in the (HCI) community. *Proactive decision support* can be described as the act of providing decision support to the user without waiting for an explicit request and proactively checking for decision failures due to various reasons like resource management. We demonstrate possible roles that existing automated planning technologies can play in the deliberative process of the human decision-maker in terms of the degree of automation of the planning process.

In the past, there have been parallels drawn between the work in HCI and AI where they were described as two fields divided by a common focus (Grudin, 2009). Since the introduction of the idea of intelligent software agents (Maes, 1995), the HCI community has used AI techniques for many applications, where *adaptive interfaces* connect directly to the notion of *adaptive agents* in the automated planning community. Furthermore, we believe that the notions of predictability of an adaptive interface (Gajos, Everitt, Tan, Czerwinski, & Weld, 2008) and explanations in the context of complex strategies for such interfaces (Rader, Cotter, & Cho, 2018) have connections to the identifiability and predictability of plans (Chakraborti, Kulkarni, Sreedharan, Smith, & Kambhampati, 2019) and explanations in automated planning literature (Chakraborti et al., 2017). Although works on the HCI side have shown how such interfaces affect user's behavior (Langley, 1999), their mental workload (Hancock & Chignell, 1988) or user satisfaction (Rader et al., 2018), works in automated planning, in the context of decision support, lack similar human studies. In this work, we seek to address this concern.

1.1. *Contributions*

The purpose of this paper is to showcase how the various planning technologies can be used to support human decision-makers. Thus, in this work, we –

- Show that state-of-the-art planning techniques can be adapted to design a Proactive Decision Support system, RADAR.
- Describe how the design decisions for RADAR are driven by the literature in the HCI community.
- Present user studies using iPass, a decision support system (similar to RADAR) designed for university students, to show the effectiveness of automated planning techniques for decision support.

Every domain comes with its nuances and thus, needs a personalized design for the decision support system to better support the use-cases desired by domain experts. RADAR showcases how a Fire Marshal may need decision support to handle emergencies in time-critical real-world scenarios. Similarly, iPass is used by graduate students to create their plan of study. Thus, we designed and implemented two different systems – one to show the applicability of human aware automated planning in real-world decision-making scenarios and the other to highlight the effectiveness of such

systems with users. Note that beyond the two domains highlighted in this paper, our methods for decision support can be leveraged, with minor changes to interfaces, across various domains.

2. Design principles

Before explaining how one can use planning technologies for decision support, we highlight the important features of the deliberative process that we have to ensure for seamless interaction with the planner.

2.1. *Naturalistic decision making*

The proposed proactive decision support system supports *naturalistic decision making* (NDM), which is a model that aims at formulating how humans make decisions in complex time-critical scenarios (Klein, 2008). It is acknowledged as a necessary element in PDS systems (Morrison, Feigh, Smallman, Burns, & Moore, 2013). Systems which do not support NDM have been found to have detrimental impact on work flow causing frustration to decision makers (Feigh, Pritchett, Denq, & Jacko, 2007). At the heart of this concept lies, as we discussed before, the requirement of letting the human to be in control. This motivates us to build a proactive decision support system, which focuses on aiding and alerting the human in the loop with his/her decisions rather than generating a static plan that may not work in the dynamic worlds that the plan has to execute in. In cases when the human wants the planner to generate complete plans, they still have the authority to ask RADAR to explain its plan when they finds it to be inexplicable (Chakraborti et al., 2017). We postulate that such a system must be augmentable, context sensitive, controllable, and adaptive to the human's decisions. Various elements of human-automation interaction such as adaptive nature and context sensitivity are necessary for the ease of usability (Sheridan & Parasuraman, 2005). It has been shown that vigilance requires hard mental work and is stressful via converging evidence from behavioral, neural, and subjective measures (Warm, Parasuraman, & Matthews, 2008). Our system may be considered as a part of such vigilance support thereby reducing the stress for the human.

2.2. *Degrees of automation*

One of the seminal works by Sheridan and Verplank builds a model that enumerates 10 levels of automation in software systems depending on the autonomy of the automated component (Sheridan & Verplank, 1978). Later, in the study of mental workload and situational awareness of humans performing alongside automation software, theoretical analysis separates the requirement for automation into four stages (Parasuraman, 2000) – Information Acquisition, Information Analysis, Decision Selection and Action Implementation (see Figure 2). We use this approach as an objective

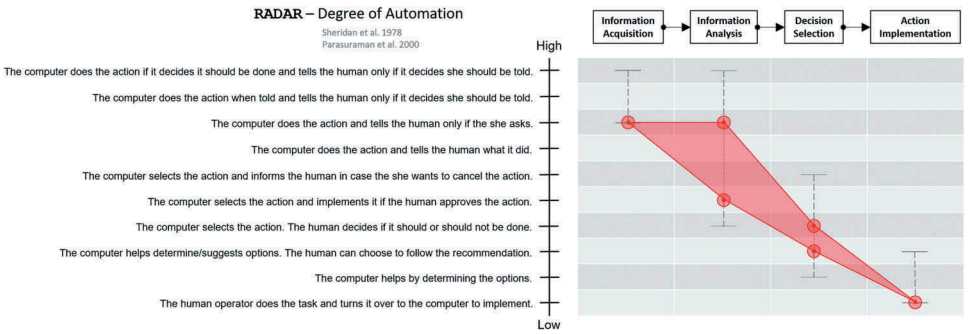


Figure 2. Degrees of automation of the various stages of decision support, and the role of RADAR in it.

basis for deciding which functions for our system should be automated and to what extent so as to reduce human's mental overload while supporting Naturalistic Decision making. Manzey shows that human use of automation may result in automation bias leading to omission and commission errors (Parasuraman & Manzey, 2010), which underlines the importance of reliability on the automation (Parasuraman & Riley, 1997). Indeed, it is well known that climbing the automation ladder (shown in Figure 2) might well improve operative performance but drastically reduce response quality when failures occur (Wickens, Li, Santamaria, Sebok, & Sarter, 2010). Hence, to meet the requirement of naturalistic decision making, we observe a downward trend in automation levels (in Figure 2) as we progress from data acquisition and analysis (which machines are traditionally better at) to decision making and execution.

2.3. *Interpretation & steering*

For the system to collaborate with the commanders effectively, in the context of a mixed-initiative setting, where the planner helps the human, it must have two broad capabilities – *Interpretation* and *Steering* (Manikonda et al., 2017). Interpretation means understanding the actions done by the commanders (eg. sub-goal extraction, plan, and goal recognition), while Steering involves helping the commanders to do their actions (e.g., action suggestion, plan critiques). The current system mainly addresses the decision making aspect, which requires the ability to both interpret and steer effectively, even as it situates itself in the level of automation it can provide in the context of naturalistic decision making.

2.4. *Human-AI interaction*

Recent work by Amershi et al. suggested guidelines based on traditional user-interface design techniques to support designers while creating AI agents (or software) (Amershi et al., 2019) that can be mapped to earlier work on principles of mixed-initiative user-interfaces (Horvitz, 1999). The former work elicitates 18 guidelines under four categories based on the different phases of interaction – (1) initially, (2) during the interaction, (3) when wrong, and (4) over time. We highlight, how these guidelines were are already considered in our interface design.

3. RADAR – decision support

This section presents automated planning techniques for decision support using the RADAR system (shown in Figure 3). RADAR system provides support to the fire-chief in the process of creating a plan to control fire in a building. During planning, the fire chief also needs to make a multitude of decisions regarding various resources such as the number of water-tanks to send, the number of ambulances to call, the areas to enclose, etc. We designed RADAR to help them in the planning process, as well as, highlight resource information. Figure 3, shows the interface for the system which is divided into four panels – (1) planning panel – for plan generation by the human, (2) goal selection panel – fire marshal can set the high-level goal, (3) map panel – shows geographical map for visual guidance and (4) resource panel – shows availability of resources like ambulance, etc.

3.1. *Different stages of decision support*

As discussed earlier, automation has four main stages, similarly automation for decision support will have three main stages: Information Acquisition, Information Analysis, and Decision Selection. Automation for decision support lacks the stage of action implementation, as the interface is just being used to help decide the best plan of actions for the given scenario. In this section, we discuss how each of these stages are supported in RADAR and the different guidelines that were useful in each stage with specific scenarios to help fire marshal handle different scenarios.

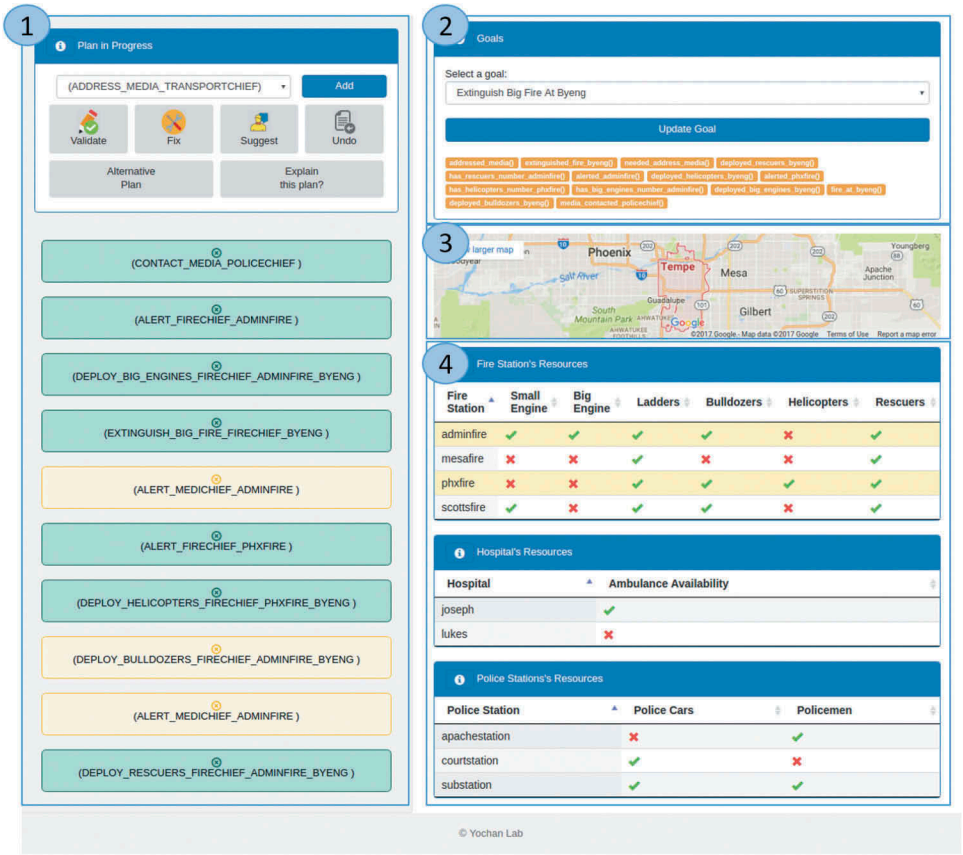


Figure 3. Illustration of the RADAR interface.

3.2. *Information Acquisition*

For effective decision support, the importance of data cannot be understated. While on one hand, it must support proactive data retrieval and integration capabilities, it must also have abilities to generate and recognize plans, and support the decision-making tasks of the commanders, with the help of this data. Thus, PDS can be seen to consist of two main capabilities, *data driven decision-making* and *decision driven data-gathering*. We call this the **Data-Decision Loop**.

In the current version, we assume that RADAR acquires relevant information regarding the availability of resources pertaining to the task at hand. We also assume that the system can keep track of drifting models in the background (Bryce, Benton, & Boldt, 2016), placing it in Degree 7 of automation. While we cannot expect the human to gather data for the system, designing a system that can choose to acquire and not display information (it thinks is irrelevant), climbing up to Degree 10, is contradictory to good design principles in automation agent design for Naturalistic Decision Making scenarios, as stated before. In the current version of our system, we do not integrate any data sources yet, but instead only focus on the decision making aspect.

Information acquired by the system is represented using the Planning Domain Definition Language (PDDL) (McDermott et al., 1998) and is assumed to be close, if not identical, to that of the expert in the loop. In PDDL, actions have pre-conditions (that need to be satisfied) and effects that change the state of the world (after execution). For example, an action such as "call an ambulance to site A" has preconditions such as (1) an ambulance is available at the hospital, (2) there is a connecting road between the hospital and the effect that an ambulance is, after action

execution, at site A. The scenario for RADAR plays out in a particular location (we use Tempe as a running example) that involves the local fire-fighting chief, in collaborating with the local police, the medical, and the transport authorities, trying to build a plan in response to a fire-scenario.

During Information Acquisition, the interface initially shows an empty planning panel, a goal selection dropdown, a map of the city, and the resource panel. There are different ways to interact with the system to determine the current status of resources and pathways (eg. blocked roads, nearest fire station to a particular location, etc.) in the map. These interactions follow the 'guidelines to show the updated information' specified in (Amershi et al., 2019).

3.3. *Information analysis*

The proposed system can leverage planning technologies to provide relevant suggestions and alerts to the human decision maker with regard to the *information needed to solve the problem*. The planning problem itself is given by $\Pi = \langle M, \mathcal{I}, \mathcal{G} \rangle$ where M is the action model, and $\mathcal{I}, \mathcal{G}$ are the current and goal states representing the current context and task description, respectively. Finally, the plan $\pi = \pi_e \circ \pi_h \circ \pi_s$ is the solution to the planning problem, which is represented as the concatenation of three sub-plans – π_e is the plan fragment that the commander has already deployed for execution, and π_h is the set of actions being proposed going forward. Of course, these two parts might not achieve the goal, and this is the role of the plan suffix π_s that is yet to be decided upon.

In RADAR, information analysis pertains to presenting the information about the initial state, the goal state and plan fragments π_e, π_h to the fire-chief who can perform various activities in this phase. For example, chose the goal from a list of goals, add actions to the planning panel, check different landmarks that need to be achieved to reach the goal, or validate a sequence of actions that are being constructed. While designing these capabilities for the interface, several guidelines for efficient human-AI interaction were kept in mind – presenting 'contextually relevant information' (depending on the goal of the problem), supporting 'efficient dismissal' (the fire-chief can overlook the landmarks and continue to create their plan), and 'making clear why the system did what it did' (provide explanations to the fire-chief) (Amershi et al., 2019). Now, we will demonstrate how planning techniques can be adapted to help with the various plan fragments highlighted above.

3.31. *Model updates*

As an augmentable system, the system must support update to the rules that govern its decision support capabilities, as required by the user, or by itself as it interacts with the environment. Of course, such models may also be learned (Zhuo, Nguyen, & Kambhampati, 2013) or updated (Bryce et al., 2016) on the fly in case of failures during execution of π_h or actions of the human in response to excuses generated from the system, or to account for model divergence due to slowly evolving conditions in the environment. Further, the system should be, if possible, act in a fashion that is easily understandable to the human in the loop (Zhang, Sreedharan, Kulkarni, Chakraborti, & Hankz Hankui Zhuo, 2017), or be able to explain the rationale behind its suggestions if required (Kambhampati, 1990; Sohrabi, Baier, & McIlraith, 2011) in a fashion easily understood by the human user (Perera, Selvaraj, Rosenthal, & Veloso, 2016).

Often a key factor in these settings is the difference in the planner's model of the domain and the human's expectation of it. Thus, a valid or satisfactory explanation corresponding to a suggestion may require a *model reconciliation process* where the human model needs to be updated, as shown in Figure 4. Chakraborti, Sreedharan, et al. have discussed the technical details for model reconciliation process, but we only highlight the four kinds of explanation (Chakraborti et al., 2019) that we have used in our systems –

- Model Patch Explanations (MPE) – all the model differences (pertaining to the missing preconditions and effects of an action in the human's model) are presented to the user. Note that some of these differences may not be relevant to the problem at hand and further, will result in information overload for the fire marshall.

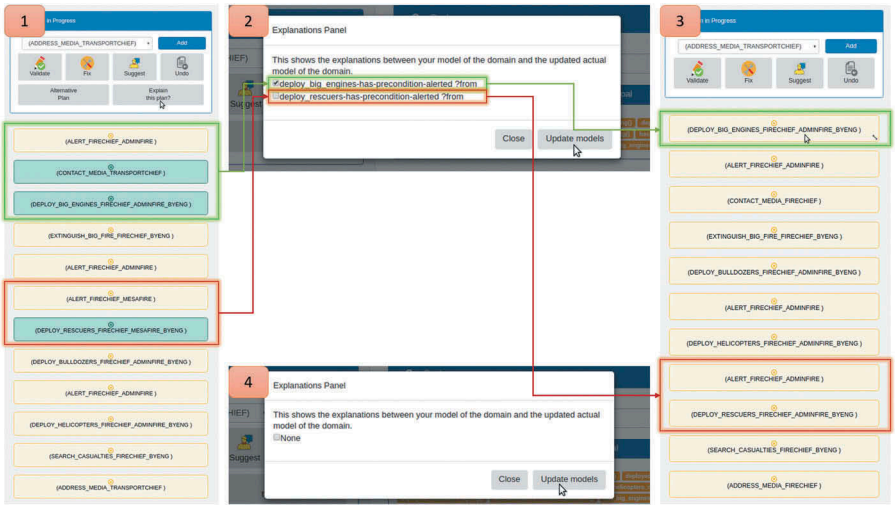


Figure 4. (1) RADAR knows that in the environment, the commander needs to inform the Fire Station's Fire chief before deploying big engines and rescuers. In green, Admin's Fire Chief is alerted to deploy big engines from Admin Fire Station. In red, Mesa fire stations' Fire Chief is alerted to deploy rescuers from Mesa Fire Station. (2) The human's model believes that there is no need to inform Fire Chiefs and questions RADAR to explain his plan. RADAR finds these differences in the domain model and reports them to the human. The human acknowledges that before deploying rescuers one might need to alert the Fire Chief and rejects the update the Fire Chief needs to be alerted before deploying big engines. (3) In the alternative plan suggested by RADAR, it takes into account the humans knowledge and plans with the updated model. (4) Clicking on 'Explain This Plan' generates no explanations as there are none (with respect to the current plan) after the models were updated.

- Plan Patch Explanations (PPE) – model differences are given in regards to only the actions that are a part of the plan that is to be explained. This brings down the size of the explanations considerably and can be computed fast (as shown later, we will use this to provide real-time explanations to users in the context of iPass).
- Minimally Complete Explanations (MCE) – a minimal set of model differences that can explain the plan suggested to the fire-chief. Although MCE reduces the amount of irrelevant information provided as an explanation, thus decreasing the human's cognitive overload, but it is time-consuming to compute.
- Minimally Monotonic Explanations (MME) – Note that MCE explanations offered at a point in time may appear incomplete when the human learns about other model differences at a later point in time. To tackle this shortcoming, MMEs are model differences that are minimal and complete (unlike MCEs).

RADAR performs model-space search to come up with *minimally complete explanations*, that strike a balance between computation time and overloading the human-in-the-loop, to explain the plan suggested by it. This design follows the guidance of presenting minimal and contextually relevant information to the user (Amershi et al., 2019). Note that here the human has the power to veto the model update if they believe that the planner's model is the one which is faulty, by choosing to approve or not approve individual parts of the explanation. Thus, the system here displays Degree 5 of automation.

3.3.2. *Plan summarization*

As we mentioned before, when a task or high level goal is selected by the human, RADAR automatically generates the corresponding planning problem in the background, analyses the possible solution to it, and highlights resources required for it to give the human an early heads-up. It can, however, do even more by using landmark analysis of the task at hand to find bottlenecks

Figure 5. Once a goal is selected, the problem file is generated and the landmarks are computed to help the commander be on track to achieve the goal.

in the future. Briefly, landmarks (Hoffmann, Porteous, & Sebastia, 2004) are (partial) *states* such that *all* plans that can accomplish the tasks from the current state must go through it during their execution, or *actions* that *must* be executed in order to reach the goal. These are referred to as state landmarks and action landmarks, respectively. Clearly, this can be a valuable source of guidance in terms of figuring out what resources and actions would be required in future, and may be used to increase the decision maker's situational awareness by summarizing the task at hand and possible solutions to it in terms of these landmarks. In the current system, we use the approach of (Zhu & Givan, 2003) for this purpose. Figure 5 illustrates one such use case, where the system automatically computes and displays the landmarks after the human selects the goal, thus exhibiting characteristics of Degree 7 automation of information analysis.

3.3.3. *Plan validation*

Plan failure occurs when the plan fragment π_e that has already been dispatched for execution and/or the sub-plan π_h currently under construction are not valid plans, i.e. $\delta(\mathcal{I}, \pi_e \circ \pi_h) \models \perp$. From the point of view of planning, this can occur due to several reasons, ranging from unsatisfied preconditions to incorrect parameters, to the model itself being incorrect or incomplete. Errors made in π_h that can be explained by the model can be easily identified using plan validation technologies like VAL (Fox, Howey, & Long, 2005), while errors in π_e should be used as feedback (context-sensitive) so that the system, in looking forward, may have to re-plan (adaptive) from a state $s \neq \delta(\mathcal{I}, \pi_e)$. VAL also validates that whether a particular action can be executed in the current state, i.e., whether all the conditions to execute an action are met or not.

Of course, the goal may be unreachable given the current state (for example, due to insufficient resources). This can be readily detected via *reachability analysis* using *planning graph* techniques. This is supported by most planners, including Fast-Downward (Helmert, 2006). Once the system detects a state with no solution to the planning problem, apart from alerting the human to this situation itself, it can choose to suggest an alternative state $\mathcal{I}^*$ where a solution does exist, i.e. $\exists \pi$ s.t. $\delta(\mathcal{I}^*, \pi) \models \mathcal{G}$. This can provide guidance to the human in how to fix the problem in situations beyond the system's control/knowledge, and may be achieved using *excuse generation* techniques (Göbelbecker, Keller, Eyerich, Brenner, & Nebel, 2010) and *plan revision* problems (Herzig, Menezes, de Barros, & Wassermann, 2014). We achieved this using a slightly modified version of the model-space search technique introduced by (Chakraborti et al., 2017) – where we create a new model with an initial state that has all the resources available and then find the minimum set of changes in our (current) faulty model which are consistent with the new model to guarantee feasibility.

3.4. Decision selection

The decision selection process is perhaps closest to home for the planning community. The fire-chief decides which actions to chose in the context of the plan fragments π_h (currently being discussed) and π_s (actions to do in the future). RADAR provides support by (1) correcting or repairing the existing plan and (2) suggesting new actions for π_s. Given the system can monitor plan generation, we follow the guidelines of 'overtime interactions' (Amershi et al., 2019). Referring back to our discussion on naturalistic decision making and the need for on-demand support, we note that our system is restricted to Degree 3 and 4 of automation with respect to decision selection.

3.4.1. Plan correction or repair

In the event π_h is invalid and may be repaired with additional actions, we can leverage the compilation pr2plan from (Ramírez & Geffner, 2010) for a slightly different outcome. The compilation, originally used for plan recognition, updates the current planning problem Π to $\Pi^* = \langle M^*, \mathcal{I}^*, \mathcal{G}^* \rangle$ using π_h as a set of *observations* such that $\forall a \in \pi_h$ is *preserved in order* in the (optimal) solution π of Π^*. The actions that occur in between such actions in the solution π to the compilation may then be used as suggestions to the user to fix the currently proposed plan π_h. Figure 6 illustrates one such use case, demonstrating Degree 3 of automation – i.e., the system only complements the decision process when asked, and provides the human an option to undo these fixes at all times. Note that since the deployed actions are required to be preserved (and the suggested actions preferably so) when looking ahead in the plan generation process, we will use Π^* for all purposes going forward.

3.4.2. Action suggestions

The most basic mode of action suggestion would be to solve the current planning problem Π^* using an optimal planner such as Fast-Downward (Helmert, 2006) and suggest the plan suffix π_s as the best course of action. Of course, the actions suggested by the commander in π_h may themselves be part of a sub-optimal plan and may thus be improved upon. Here we again use an existing compilation from (Ramírez & Geffner, 2010) for a slightly different purpose than originally intended. Given a goal, we find out if the choice $a \in \pi_h$ is sub-optimal using the difference in cost $\Delta = C(\hat{\pi}) - C(\pi)$ where $\hat{\pi}$ is the solution to the planning problem $\langle M^*, \mathcal{I}^*, \mathcal{G}^* + a \rangle$ as given by pr2plan. This is again shown in Figure 6.

3.4.3. Monitoring plan generation

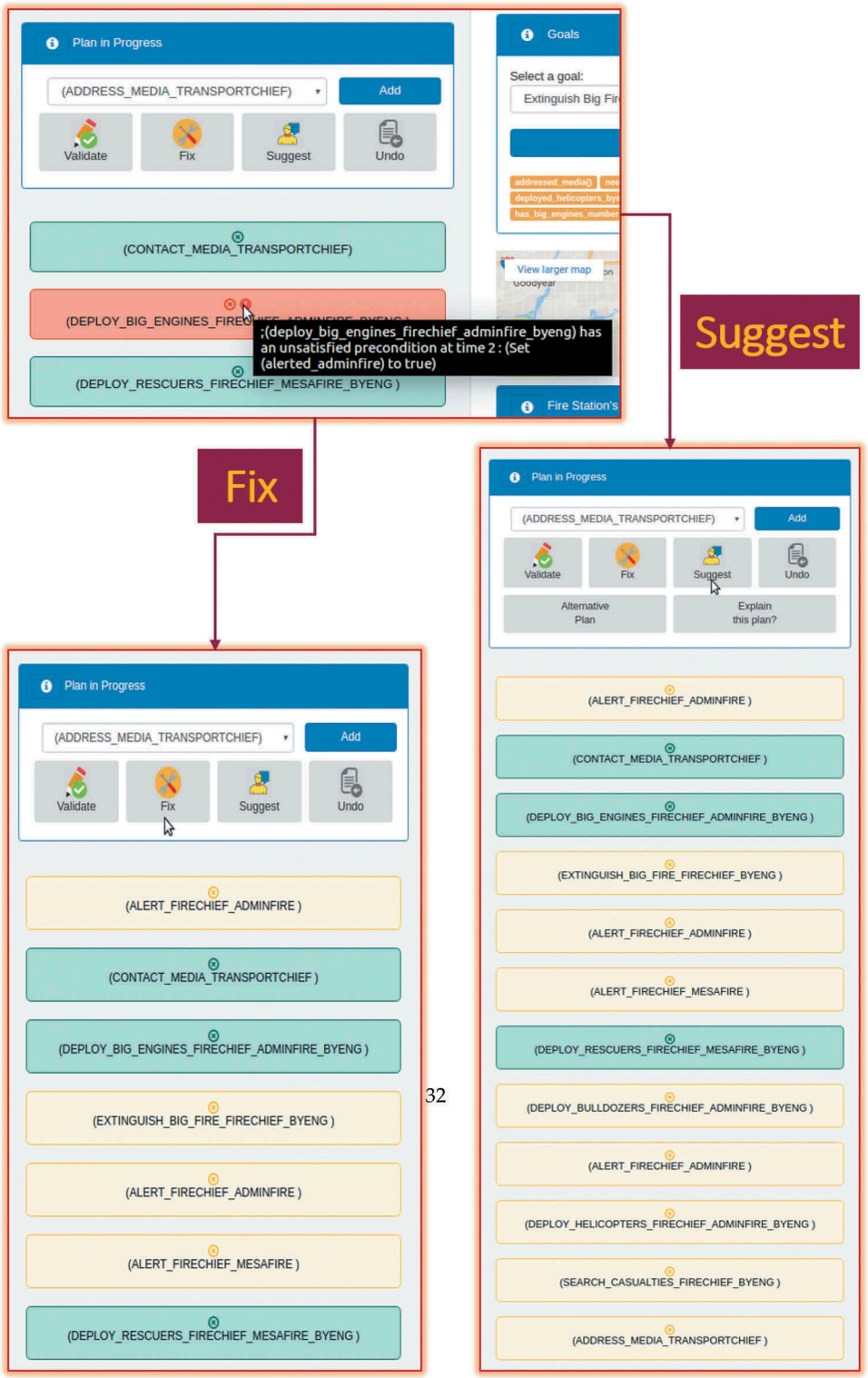
In cases where there are multiple ways to achieve the goal, and the system is not aware of the user's implicit preferences $\mathcal{P}$, we can compile the goal into $\mathcal{G}^* \leftarrow \mathcal{G} + \mathcal{P}$ and check for correctness or likelihood of $P(\mathcal{G}^* | \pi_e \circ \pi_h)$, the current hypothesis (Ramírez & Geffner, 2010). This is used by RADAR in determining the response to suggest or fix any hypothesis.

3.4.4. Plan suggestions

One useful way of increasing the situational awareness of the human decision maker is to make him/her aware of the different, often diverse, choices available. Currently, when asked for alternative plans, RADAR provides an optimal plan as a suggestion. This may not be always desired. Moreover, if landmarks are *disjunctive*, just alerting the commander of these landmarks may not be enough to tell how they contribute to the planning choices. In such cases, the concept of *diverse* (Nguyen et al., 2012) and *top-K plans* (Riabov, Sohrabi, & Udrea, 2014) become useful. We are exploring avenues of integrating these techniques into our current system.

3.5. Action implementation

The current system does not provide any endpoints to external facilities and thus lies at Degree 1 of automation in the Action Implementation phase. Some of these tasks can however be automated – e.g., in our fire-fighting domain the human can delegate the tasks for alerting police-stations and



32

Figure 6. RADAR 's 'Fix' button does plan correction, providing action suggestions. The 'Suggest' provides actions and plan suggestions to help achieve the goal.

fire-stations to be auto-completed. Thus, RADAR can potentially range from Degrees 1 to 6 in this phase. However, given how such systems are known for failing to capture the complexity of these scenarios, including some of the mixed initiative schedulers from NASA, the execution phase is often just left to the human operators completely, or firmly at the lower spectrum of the automation scale. Recent attempts (Chakraborti, Talamadupula, Fadnis, Campbell, & Kambhampati, 2017; Gombolay, Gutierrez, Clarke, Sturla, & Shah, 2015) at *learning* such action models and preferences in mixed-initiative schedulers and automated technical supports settings might provide interesting insights into climbing the automation levels at the final stage of decision support for planning, without significant loss of control.

Until now we have presented different ways to integrate automated planning techniques to perform useful portions in decision support using RADAR. But it is essential to validate the usefulness of such a system through user study. Thus in the next section, we look at iPass interface and discuss how some of the components of RADAR has been integrated to provide decision support to students.

4. iPass – system overview

The RADAR system is useful to provide support to the fire chief marshal, but to be certain about the usefulness we need to perform a user study. It is hard to find domain experts like the fire chief marshal and thus we created another system called iPass to help university students make their *plan of study* (iPOS), as their are plenty of domain experts in the university. We begin with a brief description of the domain for iPOS and the iPass interface and its decision support components.

4.1. *The iPOS domain and interface*

One of the major difficulties of designing user studies in the decision support paradigm is access to *domain experts* who can verify the real usefulness of the decision support for sequential decision making. Thus, earlier works that propose software to help the human in their decision making process (Sengupta, Chakraborti, & Kambhampati, 2018; Sengupta, Chakraborti, Sreedharan, & Kambhampati, 2017) are unable to provide any evidence as to how effective they are in practice. Keeping this in mind, we situate our study in a domain for constructing an *"interactive Plan of Study"* (iPOS) at Arizona State University. This has two implications. On the one hand, this task is known to be challenging for any student as per (1) evidence in existing literature (Khan, Poupart, & Black, 2012), and (2) its use in the International Planning Competition (Track, 2018) as a benchmark domain. On the other hand, this is a domain that graduate students, who are easily accessible in the academic setting for large-scale user studies, are already familiar with because they have to build and maintain an iPOS for themselves as per university requirements. Note that, in RADAR, we showcase the techniques that can be leveraged for decision support in real-world settings and evaluate the effectiveness of these techniques through iPass. This should give the reader an idea that beyond cosmetic changes needed for the user-interface, the decision support methods can be used out-of-the-box for other domains. Important rules that a student needs to remember while constructing an iPOS are:

- Complete 30 credits and where every course is 3 credits
- There are three required courses (these courses are to be taken that are pre-requisites from under graduate classes and a student who has not taken them) that are to be finished before any normal course and they do not count toward 30 credits.
- Define area of specialization.
- Complete 3 specialization courses.
- Choose a chair and two other committee members.
- Chair should be from same area of specialization.
- Complete 2 research courses – CSE599A & 599B.

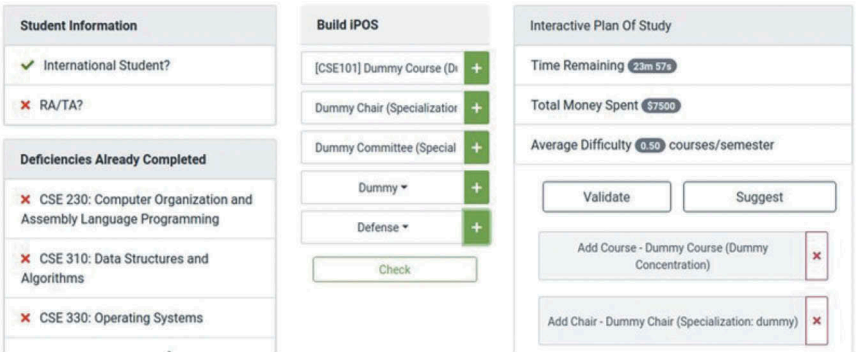


Figure 7. Illustration of the iPass interface.

- Defense is to be scheduled in the last semester.

The interface (shown in Figure 7) has three panels – (1) The panel on the left shows the relevant information of the student (e.g., what required courses they have, whether they are an international student, if they are research or teaching assistants, etc.); (2) The central panel provides the student with options to build the iPOS for the given student information. Actions in this panel can include adding a course, specialization, committee members, etc.; (3) The panel on the right provides an interactive interface to work on the plan (such as rearranging or deletion of action) along with relevant information about the plan (e.g., difficulty or average number of courses a semester, total cost of tuition for the current plan, etc.). This panel also houses the decision support components that, if available, let the user ask for validation of the current plan or suggestions to complete it. The technical details for iPass were presented in the previous section, and now we will discuss specific modules that are provided to students for the user study.

4.2. *iPass – decision support components*

Although the technical details for all the decision support components have been provided in the previous sections (for RADAR), there are certain differences that we describe in this section. In iPass, we consider $\pi_e = \pi_h$ because the human-in-the-loop is, at the start of the study, given a randomly allocated initial state and asked to make an iPOS from scratch. We note that due to constraints on the response time, the decision support components of iPass differ from RADAR, in some cases, to make them more scalable. Moreover, due to the availability of additional resources such as the user's manual, providing landmarks on the interface becomes redundant.

Plan Checking is shown as the check button in Figure 7. Given a plan of study π_h generated by the student, it checks whether $\delta(\mathcal{I}, \pi_h) \models \mathrm{T}$ i.e., the student has a valid iPOS that fulfills all the requirements necessary for graduation. The feedback generated is binary indicating whether the submitted iPOS is valid or not.

Plan Validation as mentioned before, is used to validate the student's plan π_h. For this purpose, we use VAL (Fox et al., 2005), which validates whether all actions $\in \pi_h$ can be executed (i.e., all pre-conditions are satisfied). When an action cannot be executed, a message is provided to the student explaining why the iPOS is invalid. For example, Figure 8 shows that a student, upon validation, is informed that they need to complete the course on Computer Organization (a requirement) before enrolling in the course on Artificial Intelligence (which is a graduate-level course).

Action Suggestion The goal of action suggestion is to generate π_s given a partially constructed plan π_h by the student. In order to achieve this, we use the pr2plan compilation (Ramírez & Geffner,

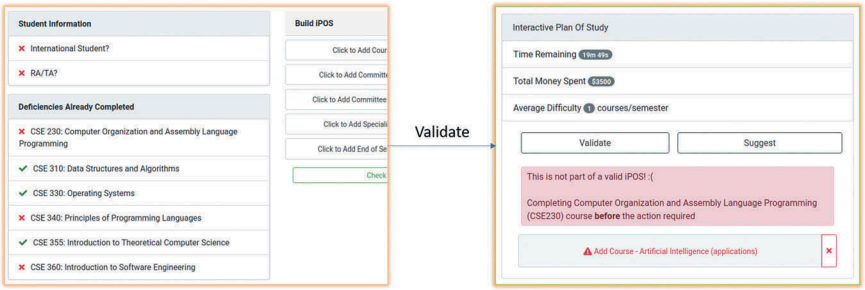


Figure 8. Illustration of Plan validation, where a student adds a course and checks whether the course can be taken at the beginning of the first semester. VAL provides feedback to the user, whether taking the action in a particular is possible or not.

2010) described above. In order to help the user distinguish between the suggested actions $\in \pi_s$ and the existing actions $\in \pi_h$, we highlight them on the interface. In the scenario shown in Figure 9, a student chooses their specialization (on Artificial Intelligence shown in black) and asks for suggestions that completes the iPOS. In this case, iPass adds the required amount of courses constrained based on the specialization, selects a graduate committee, and ensures that a dissertation is done. There may be scenarios where the input partial plan π_h cannot be completed in any way to come up with a valid iPOS. In such cases, the user is notified that the added actions cannot lead to a valid iPOS. Note that there might be various suggestions for π_s. Some of them might be preferred by a particular student. Although we do not consider this setting, an explicable (Kulkarni et al., 2016) plan completion algorithm might be useful.

Plan Explanations In order to provide meaningful explanations based on model reconciliation, we expect to have an idea of the student's understanding of an iPOS. Given that the graduate study domain made by us is significantly different from the university rules, we assumed an empty model of the student (i.e., they are not familiar with any of the constraints while constructing the iPOS). We then provide Plan Patch Explanations (PPE) that can explain the suggested plan. For example, as shown in Figure 9, the need for completing a particular course (Artificial Intelligence) is deemed to be necessary for specialization in the selected topic (AI). Note that while explanations provide details of the domain that support a plan, validation points out constraints that invalidate a plan. Thus, these functionalities are complimentary in the context of a sequential decision support system. As stated at the start of the section, due to the lesser complexity of the iPOS design task in comparison to the fire scenario and the need for quick

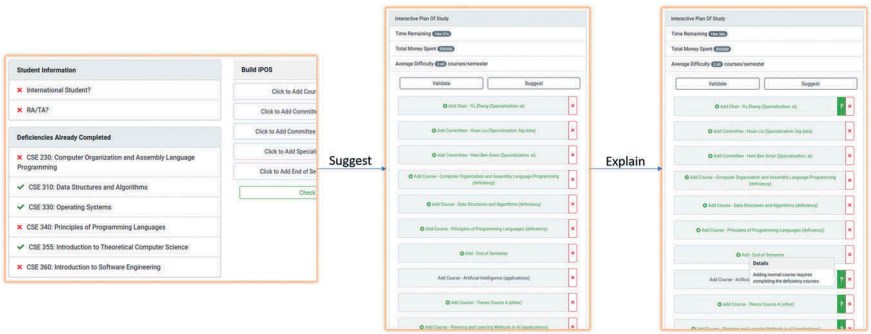


Figure 9. Illustration of Plan Suggestion and Explanation. Actions in green have been added by the planner and actions in black were added by the user. It follows from Validation scenario where the user first added the *Artificial Intelligence* course and then asks for suggestion of a complete plan with it. Explanation is shown using the box.

Stage	Support Component	RADAR	iPass
Information Acquisition	Data Decision Loop	✓	✓
Information Analysis	Plan Summarization	✓	✗
	Model Updates	✓	✓
	Plan Validation	✓	✓
Decision Selection	Plan Correction	✓	✓
	Action Suggestions	✓	✓
	Optimal Plan Suggestions	✓	✗
	Monitoring Plan Generation	✓	✗
Action Implementation		✗	✗

Figure 10. Comparison of different decision components present in RADAR and iPass system where ✓ means that the support component is part of the system and ✗ means that the support component is not part of the current version of the system.

response time, some of the decision support aspects were modified. We highlight the differences in the decision support components present in RADAR and iPass in table shown in Figure 10.

5. Aim of the study

In this section, we present the user study using iPass to evaluate the effectiveness of the decision support components present in the system. To determine the individual as well as the cumulative impact of the decision support components, we evaluated our interface in four conditions –

$C_{Control}$ Both validation and suggestion capabilities are absent. The users do have to pass correctness before they can submit.

C_1 Only validation capability is enabled.

C_2 Only suggestion capability is enabled.

C_3 Both validation and suggestion options are available.

Furthermore, each participant is assigned to one of the study conditions C_i performed the iPOS planning task twice (with different, randomly generated initial states, i.e., student portfolio). We thus, have two sub-conditions (denoted using the super-script) C_i^1 and C_i^2 for each study condition. Given these four conditions, we hypothesize that–

H1. Planning performance P will be in increasing order of –

$$P(C_{control}) < P(C_1), P(C_2) < P(C_3)$$

Note that we do not expect validation or suggestion by themselves to be more useful than the other. "Performance" here can manifest itself in different forms –

H1a. The time to completion $T(C_i)$, $i = \{Control, 1, 2, 3\}$ will follow the same order, e.g. $T(C_{control}) > T(C_1), T(C_2) > T(C_3)$.

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.

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. –

$$\Delta T(C_{control}) < \Delta T(C_1) < \Delta T(C_2) < \Delta T(C_3)$$

We expect this to happen because, in the later conditions, users are provided relevant details of the domain as they construct a plan, and are thus expected to become more familiar with the domain. We expect this effect to be more pronounced 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 for subjects with less expertise, e.g., students who had not previously completed their own iPOS.

6. Experimental results

The study was conducted on the university premises. Each subject was given $15 for an hour of study where they used the iPass software to make two iPOS s. 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 (Mercier & Sperber, 2011). Then they were given a document explaining the planning domain and another document explaining the functionality of the elements in the interface. Lastly, they were given 20 minutes to make each iPOS in order to simulate the time-critical nature of the environment. In the end, they were presented with a feedback form.

After performing pilot studies with two participants, we sent out a department-wide advertisement asking interested participants to apply for an hour's slot. Specifically, they were asked to fill a form and choose multiple time slots indicating their availability. We did not have any specific criteria to choose the participants for the study, beyond the notion of first-come-first-serve. The study was conducted over a period of 5 days in each hour, we had four students be present at the lab to take part in the study. For each participant, the specific system condition was allocated in a round-robin fashion based on their arrival time (i.e., first participant got C1, second got C2, etc.). We obtained data from 59 participants, of whom three were faced with a run-time error. Thus, we ended up with data from 56 participants (13–15 in each condition) of which six were undergraduates and others were graduate students. Out of the 56 participants, a total of 18 students had submitted an iPOS before.

Now we will present, the detailed results from the study. A part of these results was presented in the earlier version (Grover, Sengupta, Chakraborti, Mishra, & Kambhampati, 2019).

6.1. *Hypothesis H1: time to complete an iPOS and satisfaction about finalized plan follow a particular order*

H1a. We show the average time a participant took to complete the first and the second iPOS and submit their feedback[1] in Figure 11. The data shows a significant improvement in performance with

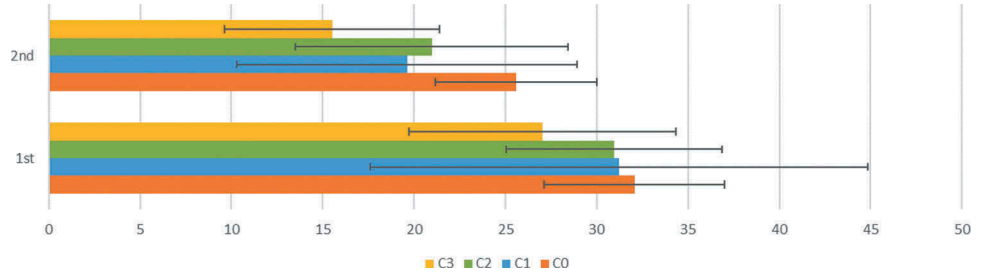


Figure 11. Average time taken (along with the standard deviation) by a participant to complete the two parts of the study for each condition C_i^1 and C_i^2.

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

regards to time as one goes from $C_{control}$ to C_3 ($p < 0.05$ for the first and $p < 0.01$ for the second iPOS) showing that the automated planning technologies in conjunction helped in improving the efficiency of the decision making process. Unfortunately, there was no significant improvement seen in performance from (1) $C_{control}$ to C_1 or C_2 and (2) C_1 or C_2 to C_3. Thus, *hypothesis H1a was found to be partially true*, thereby showing that all the planning technologies and not a subset of them were necessary to improve the planning performance for the expert in the loop.

In order to analyze the behavior of the subjects in the different study conditions, we now look at the frequency with which they used different functionalities on the interface – i.e., number of times they checked their solution for submission, and number of times they rearranged, added or deleted actions in the plan, as shown in Figure 12. As expected, the average number of checks called in the case $C_{control}$, which has no plan validation or suggestion support, is the highest, while this value is significantly less for the cases C_3 and C_1 which had validation. Considering that the number of times a user validated their plan in conditions C_1 and C_3 (shown in Figure 13), the use of check did not significantly have an impact on the time taken by the user to finish the iPOS. Also, the average number of times users rearranged actions is similar for all the conditions. Interestingly, the average number of times a user clicked delete in the conditions C_2 and C_3, indicates that although they clicked 'suggest' approximately four times in these two conditions (shown in Figure 14), they were not pleased with the plan returned by the automated planning system and ended up deleting (and adding) a lot of actions. This behavior is indicative that the planner failed to capture unspecified 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.

H1b. In Figure 15, we show the answers of the users to the subjective statement *Q3: I am happy with the final Plan of Study* on the Likert Scale for all the four conditions. In $C_{control}$, we noticed that the least number of users agreed (either agreed or strongly agreed) with the statement across all the four conditions. This is not surprising because many users were not even able to come up with a valid plan of study without any planning support in $C_{control}$. For C_1, six participants said they were in agreement with the statement Q3, and for C_2 and C_3, half of the participants were happier (i.e., either agreed or strongly agreed) with their plan of study, which is the highest across all the four conditions. But, in C_2 there was one participant who strongly disagreed with the statement, while for C_3 there were none. Thus, *the hypothesis H1b holds*.

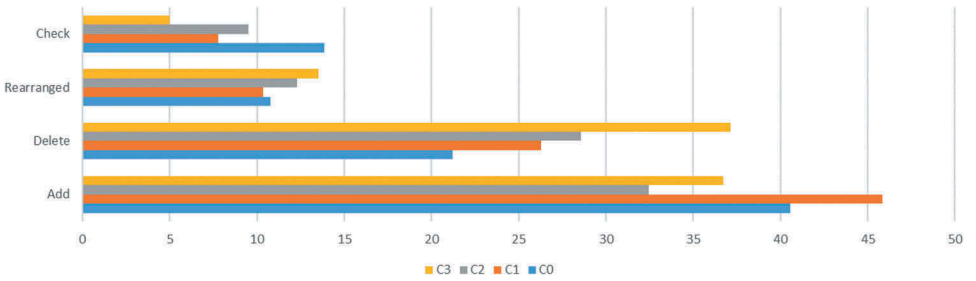


Figure 12. Average number of times participants added, deleted, rearranged courses or clicked 'check' while making an iPOS for all the conditions C_i^1.

Figure 13. Average number of times 'validate' was clicked in condition C_1^1 and C_3^1.

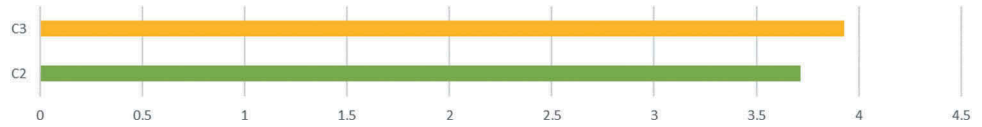


Figure 14. Average number of times 'suggest' was clicked in conditions C_2^1 and C_3^1.

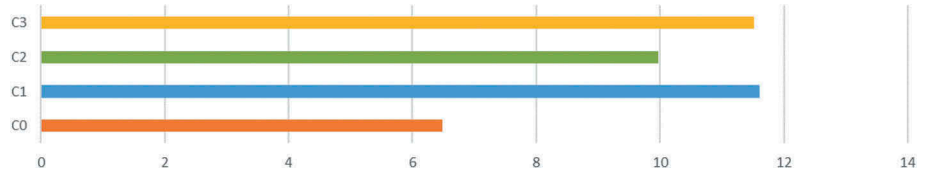


Figure 15. Average score for subjective 'Q3: I am happy with the final iPOS ' for conditions C_i^1.

Note that we mentioned earlier that the users deleted and added more actions for the conditions C_2 and C_3 that can provide action suggestions. In the light of answers to the statement Q3, we find it interesting that although the users had to edit the suggested plan, having a plan available to them to bootstrap for editing not only made them more efficient, but also increased their satisfaction.

H1c. In Figure 16, we show the number of users who agreed with the ratings on the Likert Scale for the statement *Q2: The feedback from the interface helped the iPOS making process*. If we let n_{C_i} denote the number of participants who either agree or strongly agree with the statement, then the following relation holds, $n_{C_{control}} < n_{C_1}, n_{C_2} \leq n_{C_3}$. Although the equality holds 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, we infer that *the hypothesis H1c holds*.

6.2. *Hypothesis H2: time to complete the plan will reduce at the second attempt*

We plot the average decrease in time in completing the second iPOS after doing the first iPOS with iPass for all the four study conditions in Figure 17. The lowest reduction in time for $C_{control}$ 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. We also saw that the highest reduction in time occurred for the conditions C_1 ($p < 0.1$) and C_3 ($p < 0.01$). We feel that the presence of plan validation in both these conditions informed users about the reason behind each error they made while constructing the first iPOS that was effective in teaching users about the actual

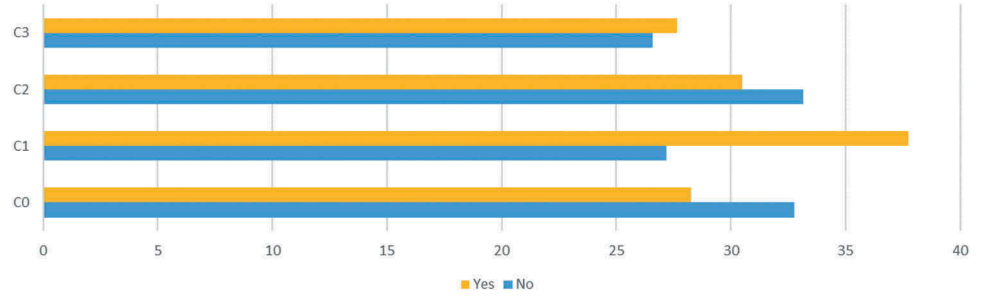


Figure 16. User agreement metrics for the statement 'Q2: The feedback from the interface helped the iPOS making process' for each condition C_i^1.

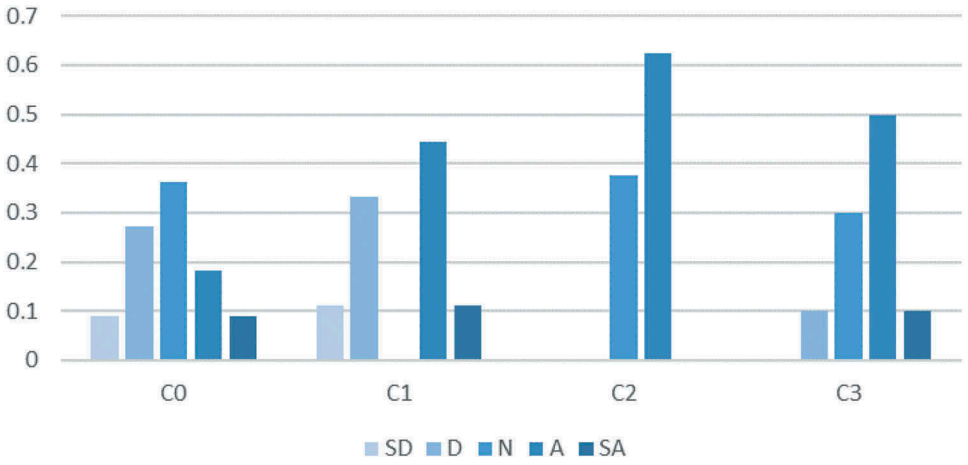


Figure 17. Time difference $\Delta T(C_i)$ between two tasks C_i^1 and C_i^2 of iPOS planning for every condition C_i.

domain. For a similar reason, we had 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 about the domain, thus reconciling the models. Unfortunately, this functionality was used very rarely (0.14 and 0.91 average number of times for C_2 and C_3) and thus, improvement in performance was not observed. Hence, *H2 was only found to be partially true*, supporting the claim that use of automated planning C_3 for decision support improved the efficiency of the human thereby reduced the time for making the second iPOS.

6.3. *Hypothesis H3: less expert users benefit more from decision support components*

We 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 18). Although the experienced participants did perform slightly better in $C_{control}$, C_1 and C_3, to our surprise, we noticed that for C_2, the users who had no prior experience performed better. This might be because the latter 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

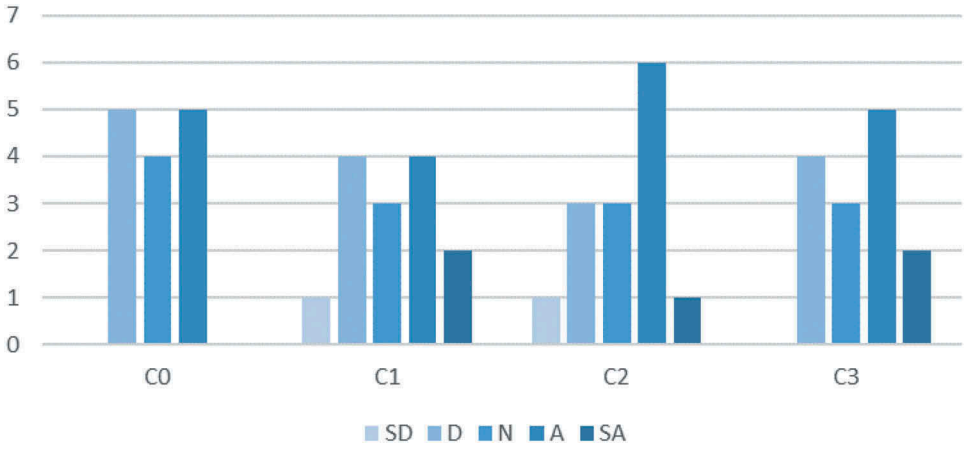


Figure 18. Time taken by experienced (in yellow) and non-experienced (in blue) users to make the first iPOS (C_i^1).

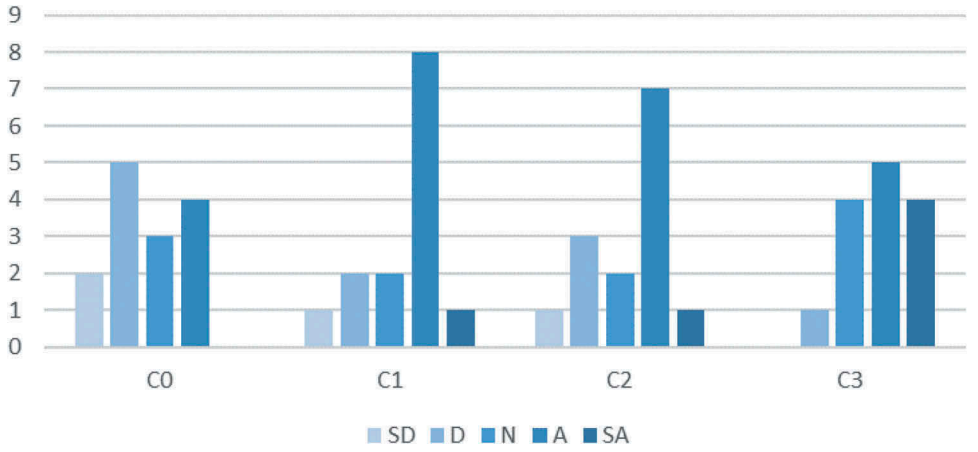


Figure 19. Feedback of non-experienced users about the statement 'Q1: The planning task was pretty simple for me' for each condition C_i^1.

'validate' in C_1, they might have ended up having to correct their partial plans multiple times, resulting in a longer time and worse performance.

We plot the response of non-experienced users to the subjective question *Q1: The planning task was pretty simple for me* in Figure 19. Interestingly, the non-experienced users seemed to agree (or strongly agree) more with the statement in C_3 compared to $C_{control}$, indicating that support features have contributed to decrease in perceived difficulty of the task.

Table shown in Figure 20, shows the summary of the results for every hypothesis, and compares it to the actual outcome based on the statistical significance results.

6.4. *Qualitative results*

We asked the users three qualitative questions –

Q1. Describe in detail at least 5 things you liked about the Plan of Study you came up with.

Q2. Describe in detail at least 5 things you did NOT like about the Plan of Study you came up with.

Q3. Describe in detail what other features of the interface you would like to have.

	Measure	Expected	Outcome
H1a.	Time taken	$T(C_{control}) > T(C_1) > T(C_2) > T(C_3)$	$T(C_{control}) > T(C_1) \approx T(C_2) > T(C_3)$
H1b.	Satisfaction iPOS	$n_{C_{control}} < n_{C_1} < n_{C_2} < n_{C_3}$	$n_{C_{control}} < n_{C_1} < n_{C_2} < n_{C_3}$
H1c.	Satisfaction interface	$n_{C_{control}} < n_{C_1} < n_{C_2} < n_{C_3}$	$n_{C_{control}} < n_{C_1} < n_{C_2} < n_{C_3}$
H2.	Time difference	$\Delta T(C_{control}) < \Delta T(C_1) < \Delta T(C_2) < \Delta T(C_3)$	$\Delta T(C_{control}) < \Delta T(C_1) \approx \Delta T(C_2) \approx \Delta T(C_3)$
H3.	Time taken less experienced	$T(C_{control}) > T(C_1) > T(C_2) > T(C_3)$	$T(C_{control}) \approx T(C_1) \approx T(C_3) \approx T(C_2)$

Figure 20. Summary of Results.

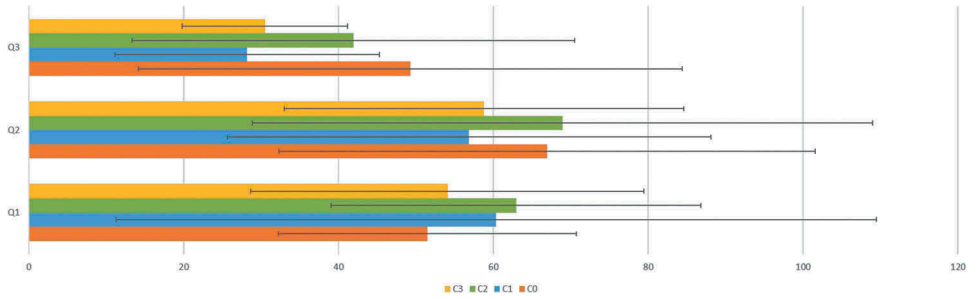


Figure 21. Average word count for every feedback question, with error bars showing ± 1 standard deviation for the word count. "Liked" is 5 things you liked about your iPOS, "Didn't Like" is 5 things you didn't like about your iPOS and "What more" is what other features of the interface you would like to have.

Figure 21 shows the average word count for the questions with error bar is ± 1 standard deviation. There were three cases where the word count difference was statistically significant (< 0.05). First, for Q3, the word count was lower for C_3 compared to $C_{control}$. This implied that users in $C_{control}$ requested many features while, provided with the added functionalities of validate and suggest, their enumeration of *what more* came down significantly. Second, we had a similar conclusions in the case of C_1 and $C_{control}$. This result raises an interesting question – if we were able to significantly reduce the user's demand for more features with only the validate functionality in C_1, what added purpose did the suggest and explain functionality in C_3 serve? Third, lower numbers of participants liked the system (Q1) compared to the ones who disliked it (Q2) for the condition $C_{control}$.

Having read all this feedback, we found that participants preferred the "Validate" functionality compared to the "Suggest" functionality because it pointed out specific errors when they were stuck (although this was not statistically significant). For Q3 in $C_{control}$, participants said that they wanted specific errors that would show why a submitted iPOS check fails; in other words, they felt that plan validation functionality would have been helpful. In condition C_3, feedback for Q3 was more in regard to personal preferences, showcasing that users started caring about plan quality when, due to decision support functionalities, coming up with a valid iPOS was no longer a challenging task. We further highlight some of our interesting feedback as a part of the future research directions.

7. Discussion and future work

As mentioned above, the decision support described in this work uses a set of domain-independent techniques in automated planning that provide the back-end functionalities of plan validation, summarization, suggestion, explanation, etc. While these can be used for plug-and-play, there are some aspects of the decision support system that need to be catered to the specific domain for it to be effective. For example, landmarks which provide relevant information to keep a commander aware of their goal in RADAR ceases to be important for iPass where subjects have access to an iPOS handbook. Furthermore, it also helps us to identify some shortcomings of current planning technologies necessary to make the decision support effective in real-time. Before ending the section, we briefly talk about the suggestions provided in the subjective user feedback, highlighting directions for possible future research. We finish the section by highlighting the connections between HCI and AI.

7.1. *Domain-specific designs*

There are various components of the decision support system that need careful attention when it is used in the context of a specific domain. The foremost among these is the user-interface. For

example, while a resource panel was useful in the context of a fire-fighting scenario (in RADAR), it was replaced by a panel that showcases the student information for the iPass domain. In domains that are close to scheduling problems, it is often necessary to revamp the entire user-interface design (Mishra, Sengupta, Sreedharan, Chakraborti, & Kambhampati, 2019). The use of domain-independent technology in the back-end ensures that beyond cosmetic changes to the front-end, all functionalities can be provided with little effort.

7.2. *Scalability of back-end technologies*

Many of the back-end technologies suffer from scalability issues. When domains become complex, finding optimal plans within a reasonable amount of time becomes difficult. This leads to longer wait times when the plan or the action suggestion modules are called.

A scalability *vs*. verbosity trade-off exists in the case of generating explanations based on model reconciliations. While minimally complete explanations help the human understand the validity (and optimality) of the suggested plan, time taken to compute makes them unusable in the context of real-world settings like iPass. Furthermore, explanations that can be computed faster (e.g., plan patch explanations), are often verbose, adding to the existing cognitive overload of the human-in-the-loop. Furthermore, the, explanations provided should ideally be the start to a conversation that helps users elicit either their preferences or (expert) knowledge about the domain. Thus, research that facilitates the two-way communication between the decision support and the human-in-the-loop, thereby learning from one another, could be an interesting future work.

7.3. *User feedback*

Depending on the condition assigned to a particular user, their feedback varied considerably. While users in $C_{control}$ asked for features like validation, users in C_3 expected the system to provide suggestions that are more personalized for them. Given that different subjects, with different student information assigned to them, belong in a spectrum of preferences, solutions that generate explicable plans (Kulkarni et al., 2016) cannot be simply used out-of-the-box. The reason being that they assume all human models come from the same distribution.

7.3.1. *HCI and AI*

Maes discussed her vision of intelligent software agents that would know the user's interests and act autonomously on their behalf. She divided the task of creating such agents into – (1) knowledge gathering or learning models from the data, and (2) then utilizing them to support the users (Maes, 1995; Maes et al., 1997). In this paper, we presented an end-to-end software agent which assumes knowledge about a user's capabilities and collaborates with them by providing support for sequential decision making. We used ideas from both HCI and AI communities to make this software, such as, 'design principles' for the interface (Amershi et al., 2019; Parasuraman, Sheridan, & Wickens, 2000), ideas from 'automation' to understand the modules to be automated and the degree of automation (Parasuraman et al., 2000; Sheridan & Verplank, 1978) and 'automated planning techniques' to implement the system (Chakraborti et al., 2019; Ramírez & Geffner, 2009; Sheridan & Verplank, 1978).

In the past, there have been suggestions that HCI and AI are two different communities with a common focus (Grudin, 2011). Early work related to integrating smaller components to the system, such as integrating e-mail classifier to the e-mail management software (Horvitz, 1999). Combining these components in the software created difficulties in their own right, for example, to ensure that the user may not miss an important e-mail. Through RADAR we have not just designed an intelligent component for a system, but rather created an intelligent software agent bringing these fields together.

8. Conclusion

In this article, we presented a decision support system that uses automated planning techniques to support sequential planning problems for a human-in-the-loop. We first introduced RADAR and described how different planning technologies such as validation, plan recognition, landmarks, model-reconciliation based explanations can be used to aid a human commander in a time-critical domain. We then situate the various capabilities of RADAR on the automation hierarchy, carefully describing the design choices we deliberately make. Unfortunately, testing the effectiveness of such systems became challenging given the lack of experts. To address this challenge, we designed a test-bed system, iPass that enables support for a domain in which university students (our subjects for the study) are already experts.

The effectiveness of the system was measured by creating different study groups using different subset of capabilities of the system *vs.* a control group. The evaluation was based on their (1) time taken to complete the planning task, (2) time taken to perform similar tasks across multiple trials and (3) the effect of their expertise level. In summary, we found that two key decision support components – validation and suggestion – for human-in-the-loop planning tasks were useful in improving the performance and/or satisfaction of the decision-maker. From subjective feedback, we found that 11 students asked for more feedback from the interface in $C_{control}$ (3 of whom mentioned feedback that can suggest new courses and 5 mentioned validation kind of feedback) thus, highlighting the role of the support components for the normative expectations of the user. We also believed that providing explanations to users will have a positive impact on decision support but after the study, we realized that we need to learn accurate human models to provide personalized decision support and explanations.

Background

Initial part of the results shown in the paper was presented at – Grover, S., Sengupta, S., Chakraborti, T., Mishra, A. P., & Kambhampati, S. (2019). iPass: A Case Study of the Effectiveness of Automated Planning for Decision Support.

Acknowledgments

We acknowledge the help of Dr. Satya Gautam Vadlamudi in the initial design of the RADAR framework and Sarath Sreedharan for sharing his code and engaging in useful discussions in regards to generating explanations.

Funding

This research is supported in part by ONR grants N00014-16-1-2892, N00014-18-1-2442, N00014-18-1-2840, N00014-9-1-2119, AFOSR grant FA9550-18-1-0067, DARPA SAIL-ON grant W911NF-19-2-0006, NSF grants 1936997 (C-ACCEL), 1844325, NASA grant NNX17AD06G, and a JP Morgan AI Faculty Research grant;Air Force Office of Scientific Research [FA9550-18-1-0067];Defense Advanced Research Projects Agency [W911NF-19-2-0006];National Aeronautics and Space Administration [NNX17AD06G];National Science Foundation [1844325,1936997];Office of Naval Research [N00014-9-1-2119, N00014-16-1-2892, N00014-18-1-2442, N00014-18-1-2840];JP Morgan AI Faculty Research grant;

HCI Editorial Record

First received on *June 15th, 2019*. Revisions received on *November 14th, 2019*. Accepted by *Dr. Munmun De Choudhury*. Final manuscript received on *January 31st, 2020*.

Notes on contributors

Sachin Grover (sachin.grover@asu.eduhttps://sites.google.com/site/sachingrover211/) is a *Doctoral student in Computer Science* with an interest in *Human Aware Artificial Intelligence and multi-agent systems*; he is a *graduate research scientist* in the *Yochan Lab* of *Arizona State University*.

Sailik Sengupta (sailiks@asu.eduhttps://sailik1991.github.io/) works in *multi-agent systems* with an interest in *proactive cyber defense and automated decision support*; he is a *Doctoral student* in the *department* of *Computer Science* at *Arizona State University*.

Tathagata Chakraborti (tchakra2@ibm.comtchakra2.com) works in *Artificial Intelligence* with an interest in *automated planning*; he is a *Research Staff Member* in the *AI Interaction Group* of *IBM Research AI (Cambridge, USA)*.

Aditya Prasad Mishra (amishr28@asu.edu) works as a *Software Developer* in *Epitech, Michigan*.

Subbarao Kambhampati (rao@asu.eduhttp://rakaposhi.eas.asu.edu) works in *Artificial Intelligence* with a focus on *Human-Aware AI Systems*; he is a *Professor* in the *Computer Science Department* of *Arizona State University* and a Fellow of AAAI, AAAS and ACM.

References

Ai-Chang, M., Bresina, J., Charest, L., Chase, A., Hsu, -J.-J., & Jonsson, A. (2004). Mapgen: Mixed-initiative planning and scheduling for the mars exploration rover mission. *IEEE Intelligent Systems*. doi:10.1109/MIS.2004.1265878

Allen, J. F. (1994). Mixed initiative planning: Position paper. In *Proceedings of Arpa/rome labs planning initiative workshop*, Tuscon, AZ, USA. Morgan Kaufman, Palo Alto.

Amershi, S., Weld, D., Vorvoreanu, M., Fourney, A., Nushi, B., & Collisson, P., 2019). Guidelines for human-ai interaction. In *Proceedings of the 2019 chi conference on human factors in computing systems* (pp. 1–13). Glasgow, Scotland.

Bryce, D., Benton, J., & Boldt, M. W. (2016). Maintaining evolving domain models. In *Proceedings of the twenty-fifth international joint conference on artificial intelligence* (pp. 3053–3059). Palo Alto, CA: AAAI Press/International Joint Conferences on Artificial Intelligence.

Chakraborti, T., Kulkarni, A., Sreedharan, S., Smith, D. E., & Kambhampati, S. (2019). Explicability? legibility? predictability? transparency? privacy? security? the emerging landscape of interpretable agent behavior. In *Proceedings of the international conference on automated planning and scheduling* (Vol.29, pp. 86–96), Berkley, California, USA.

Chakraborti, T., Sreedharan, S., Grover, S., & Kambhampati, S. (2019). Plan explanations as model reconciliation: An empirical study. In Proceedings of the 14th ACM/IEEE International Conference on Human-Robot Interaction, Daegu, South Korea (pp. 258–266). doi:10.1109/HRI42470.2019

Chakraborti, T., Sreedharan, S., Zhang, Y., & Kambhampati, S. (2017). Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (pp. 156–163).

Chakraborti, T., Talamadupula, K., Fadnis, K., Campbell, M., & Kambhampati, S. (2017). *UbuntuWorld 1.0 LTS – A platform for automated problem solving & troubleshooting in the Ubuntu OS*. In *Twenty-ninth IAAI Conference*, San Fransisco, CA.

Dragan, A. D., Lee, K. C., & Srinivasa, S. S. (2013). Legibility and predictability of robot motion. In *Proceedings of the 8th acm/ieee international conference on human-robot interaction* (pp. 301–308), Tokyo, Japan.

Feigh, K. M., Pritchett, A. R., Denq, T. W., & Jacko, J. A. (2007). Contextual control modes during an airline rescheduling task. *Journal of Cognitive Engineering and Decision Making*, *1*(2), 169–185. doi:10.1518/155534307X232839

Fox, M., Howey, R., & Long, D. (2005). Validating plans in the context of processes and exogenous events. In *Proceedings of the Twentieth AAAI Conference on Artificial Intelligence*, Pittsburgh, PA (Vol. 5, pp. 1151–1156).

Gajos, K. Z., Everitt, K., Tan, D. S., Czerwinski, M., & Weld, D. S. (2008). Predictability and accuracy in adaptive user interfaces. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 1271–1274), Florence, Italy.

Göbelbecker, M., Keller, T., Eyerich, P., Brenner, M., & Nebel, B. (2010). Coming up with good excuses: What to do when no plan can be found. In *Proceedings of the twentieth international conference on international conference on automated planning and scheduling*, Toronto, Canada (pp.81–88).

Gombolay, M., Gutierrez, R., Clarke, S., Sturla, G., & Shah, J. (2015). Decision-making authority, team efficiency & human worker satisfaction in mixed human-robot teams. *Autonomous Robots*. doi:10.1007/s10514-015-9457-9

Grover, S., Sengupta, S., Chakraborti, T., Mishra, A. P., & Kambhampati, S. (2019). iPass: A case study of the effectiveness of automated planning for decision support. *Naturalistic Decision Making*.

Grudin, J. (2009). Ai and hci: Two fields divided by a common focus. *Ai Magazine*, *30*(4), 48. doi:10.1609/aimag.v30i4.2271

Grudin, J. (2011). Human-computer interaction. *Annual Review of Information Science and Technology*, *45*(1), 367–430. doi:10.1002/aris.144.v45:1

Hancock, P. A., & Chignell, M. H. (1988). Mental workload dynamics in adaptive interface design. *IEEE Transactions on Systems, Man, and Cybernetics*, *18*(4), 647–658. doi:10.1109/21.17382

Helmert, M. (2006). The fast downward planning system. *JAIR*, *26*, 191–246. doi:10.1613/jair.1705

Herzig, A., Menezes, V., de Barros, L. N., & Wassermann, R. (2014). On the revision of planning tasks. In *Proceedings of the Twenty-First European Conference on Artificial Intelligence*, Prague, Checz Republic (pp. 435–440).

Hoffmann, J., Porteous, J., & Sebastia, L. (2004). Ordered landmarks in planning. *JAIR*, *22*, 215–278. Retrieved from http://dl.acm.org/citation.cfm?id=1622487.1622495

Horvitz, E. (1999). Principles of mixed-initiative user interfaces. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 159–166), Pittsburgh, PA. doi:10.1016/s1095-0397(99)00016-3

Kambhampati, S. (1990). A classification of plan modification strategies based on coverage and information requirements. In *Aaai spring symposium on case based reasoning*.

Kambhampati, S., & Talamadupula, K. (2015). Human-in-the-loop planning and decision support. In *AAAI tutorial*.

Khan, O. Z., Poupart, P., & Black, J. P. (2012). Automatically generated explanations for markov decision processes. In *Decision theory models for applications in artificial intelligence: Concepts and solutions* (pp. 144–163). Hershey, Pennsylvania: IGI Global.

Kim, J., Banks, C. J., & Shah, J. A. (2017). Collaborative planning with encoding of users' high-level strategies. In *Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence* (pp.955–961).

Klein, G. (2008). Naturalistic decision making. *The Journal of the Human Factors and Ergonomics Society*, *50*, 456–460. doi:10.1518/001872008X288385

Kulkarni, A., Chakraborti, T., Zha, Y., Vadlamudi, S. G., Zhang, Y., & Kambhampati, S. (2016). Explicable robot planning as minimizing distance from expected behavior. *CoRR, abs/1611.05497*. Retrieved from. http://arxiv.org/abs/1611.05497

Langley, P. (1999). User modeling in adaptive interface. In *Proceedings of the seventh international conference on user modeling* (pp. 357–370) Banff, Canada. Springer.

Laskey, K. B., Marques, H. C., & da Costa, P. C. G. (2016). High-level fusion for crisis response planning. In *Fusion Methodologies in Crisis Management (pp. 257–285). Springer.*

Lesh, A. G. N. (2004). *Applying collaborative discourse theory to human-computer interaction.*

Maes, P. (1995). Intelligent software. *Scientific American*, *273*(3), 84–86.

Maes, P., Shneiderman, B., & Miller, J. (1997). Intelligent software agents vs. user-controlled direct manipulation: A debate. In *Chi'97 Extended Abstracts on Human Factors in Computing Systems* (pp. 105–106).

Manikonda, L., Chakraborti, T., Talamadupula, K., & Kambhampati, S. (2017). Herding the crowd: Using automated planning for better crowdsourced planning. *Journal of Human Computation*, *4*. doi:10.15346/hc.v4i1

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

Mercier, H., & Sperber, D. (2011). Why do humans reason? arguments for an argumentative theory. *Behavioral and Brain Sciences*, *34*(2), 57–74. doi:10.1017/S0140525X10000968

Mishra, A. P., Sengupta, S., Sreedharan, S., Chakraborti, T., & Kambhampati, S. (2019). Cap: A decision support system for crew scheduling using automated planning. *Naturalistic Decision Making*.

Morrison, J. G., Feigh, K. M., Smallman, H. S., Burns, C. M., & Moore, K. E. (2013). The quest for anticipatory decision support systems. *Human Factors and Ergonomics Society Annual Meeting*.

Narayanan, V., Zhang, Y., Mendoza, N., & Kambhampati, S. (2015). Automated planning for peer-to-peer teaming and its evaluation in remote human-robot interaction. In *Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction extended abstracts*, Portland, US (pp. 161–162).

Nguyen, T. A., Do, M., Gerevini, A. E., Serina, I., Srivastava, B., & Kambhampati, S. (2012). Generating diverse plans to handle unknown and partially known user preferences. *Artificial Intelligence*. doi:10.1016/j.artint.2012.05.005

Parasuraman, R. (2000). Designing automation for human use: Empirical studies and quantitative models. *Ergonomics*, *43*, 931–951. doi:10.1080/001401300409125

Parasuraman, R., & Manzey, D. H. (2010). Complacency and bias in human use of automation: An attentional integration. *Human Factors: the Journal of the Human Factors & Ergonomics Society*, *52*, 381–410. doi:10.1177/0018720810376055

Parasuraman, R., & Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. *Human Factors: the Journal of the Human Factors and Ergonomics Society*, *39*, 230–253. doi:10.1518/001872097778543886

Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. *Trans. Sys. Man Cyber. Part A*, *30*, 286–297. Retrieved from. doi:10.1109/3468.844354

Perera, V., Selvaraj, S. P., Rosenthal, S., & Veloso, M. (2016, August). Dynamic generation and refinement of robot verbalization. In *Robot and human interactive communication (ro-man) (pp. 212–218)*. Columbia University, NY.

Rader, E., Cotter, K., & Cho, J. (2018). Explanations as mechanisms for supporting algorithmic transparency. In *Proceedings of the 2018 chi conference on human factors in computing systems,* Montreal, Quebec (p. 103).

Ramírez, M., & Geffner, H. (2009). Plan recognition as planning. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence*, Pasadena, California (pp. 1778–1783).

Ramírez, M., & Geffner, H. (2010). Probabilistic plan recognition using off-the-shelf classical planners. In *Aaai conference on artifical intelligence* (pp. 1121–1126).

Riabov, A., Sohrabi, S., & Udrea, O. (2014). New algorithms for the top-k planning problem. In *Proceedings of the scheduling and planning applications workshop (spark) at the 24th international conference on automated planning and scheduling (icaps)*, Portsmouth, New Hampshire (pp.10–16).

Sengupta, S., Chakraborti, T., & Kambhampati, S. (2018). Ma-radar-a mixed-reality interface for collaborative decision making. In *Proceedings of user interfaces and scheduling and planning (uisp) workshop at 28th international conference on automated planning and scheduling (icaps)*, Delft, Netherlands (pp. 40–45).

Sengupta, S., Chakraborti, T., Sreedharan, S., Vadlamudi, S. G., & Kambhampati, S. (2017). RADAR - A proactive decision support system for human-in-the-loop planning. In *Proceedings of user interfaces and scheduling and planning (uisp) workshop at 27th international conference on automated planning and scheduling (icaps)*, Pittsburgh, USA

Sheridan, T. B., & Parasuraman, R. (2005). Human-automation interaction. *Reviews of Human Factors and Ergonomics*, *1*, 89–129. doi:10.1518/155723405783703082

Sheridan, T. B., & Verplank, W. L. (1978). *Human and computer control of undersea teleoperators (Tech. Rep.)*. Massachusetts Inst of Tech Cambridge Man-Machine Systems Lab.

Shneiderman, B., & Maes, P. (1997). Direct manipulation vs. interface agents. *interactions*, *4*(6), 42–61. doi:10.1145/267505.267514

Smith, D. E. (2012). Planning as an iterative process. In *Proceedings of the Twenty-Sixth aaai Conference on Artificial Intelligence*, Toronto, Ontario, Canada (pp. 2180–2185).

Sohrabi, S., Baier, J. A., & McIlraith, S. A. 2011. Preferred explanations: Theory and generation via planning. In *Proceedings of the Twenty-fifth aaai Conference on Artificial Intelligence*, San Fransisco, California, USA (pp. 261–267). AAAI Press.https://ipc2018-probabilistic.bitbucket.io/#domains

Track, I. P. C. I. P. (2018). Academic advising domain. Retrieved from https://ipc2018-probabilistic.bitbucket.io/#domains

Warm, J. S., Parasuraman, R., & Matthews, G. (2008). Vigilance requires hard mental work and is stressful. *Human Factors: the Journal of the Human Factors and Ergonomics Society*, *50*, 433–441. doi:10.1518/001872008X312152

Wickens, C. D., Li, H., Santamaria, A., Sebok, A., & Sarter, N. B. (2010). Stages and levels of automation: An integrated meta-analysis. In *Proceedings of the human factors and ergonomics society annual meeting*, San Fransisco, California (Vol. 54, pp. 389–393). doi:10.1177/154193121005400425

Zhang, Y., Narayanan, V., Chakraborti, T., & Kambhampati, S. (2015). A human factors analysis of proactive support in human-robot teaming. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany (pp. 3586–3593).

Zhang, Y., Sreedharan, S., Kulkarni, A., Chakraborti, T., & Hankz Hankui Zhuo, S. K. (2017). Plan explicability and predictability for robot task planning. In *2017 IEEE international conference on robotics and automation (icra)*, Marina Bay Sands, Singapore (pp. 1313–1320).

Zhang, Y., Sreedharan, S., Kulkarni, A., Chakraborti, T., Zhuo, H. H., & Kambhampati, S. (2016). Plan explicability for robot task planning. In Rss workshop on planning for hri: Shared autonomy and collaborative robotics.

Zhu, L., & Givan, R. (2003). Landmark extraction via planning graph propagation. *ICAPS Doctoral Consortium*, 156–160.

Zhuo, H. H., Nguyen, T., & Kambhampati, S. (2013). Refining incomplete planning domain models through plan traces. In *Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence*, Beijing, China (pp. 2451–2457).