Human-Agent Groups: Studies, Algorithms and Challenges:
AAAI Technical Report FS-17-04

RADAR — A Proactive Decision Support System for Human-in-the-Loop Planning

Sailik Sengupta, Tathagata Chakraborti, Sarath Sreedharan
Satya Gautam Vadlamudi, Subbarao Kambhampati
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University{sailiks,tchakra2,ssreedh3}@asu.edu,
gautam.vadlamudi@capillarytech.com, rao@asu.edu

Abstract

Proactive Decision Support (PDS) aims at improving the decision making experience of *human decision makers* by enhancing both the quality of the decisions and the ease of making them. In this paper, we ask the question what role automated decision making technologies can play in the deliberative process of the human decision maker. Specifically, we focus on expert humans in the loop who now share a detailed, if not complete, model of the domain with the assistant, but may still be unable to compute plans due to cognitive overload. To this end, we propose a PDS framework RADAR based on research in the automated planning community that aids the human decision maker in constructing plans. We will situate our discussion on principles of interface design laid out in the literature on the degrees of automation and its effect on the collaborative decision making process. Also, at the heart of our design is the principle of *naturalistic decision making* which has been shown to be a necessary requirement of such systems, thus focusing more on providing suggestions rather than enforcing decisions and executing actions. We will demonstrate the different properties of such a system through examples in a fire-fighting domain, where human commanders are involved in building response strategies to mitigate a fire outbreak. The paper is written to serve both as a position paper by motivating requirements of an effective proactive decision support system, and also an emerging application of these ideas in the context of the role of an automated planner in human decision making, in a platform that can prove to be a valuable test bed for research on the same.

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

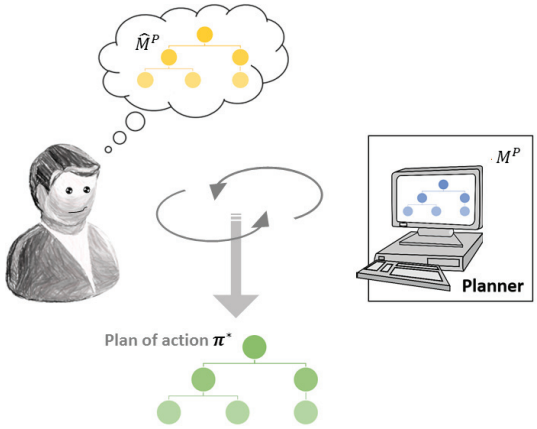

Figure 1: Planning for decision support must consider difference in models between the planner and the human.

Here we investigate the extent to which an automated planner can support the humans' 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 under execution and its results. This is in contrast to earlier work on systems such as TRAINS (Allen 1994), MAPGEN (Ai-Chang et al. 2004) and (Kim, Banks, and Shah 2017) where the planner is in the drivers seat, with the humans "advising" the planner. This is distinct from earlier work 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 due to the fact that the planner and the human can have different (possibly 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 as well as 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

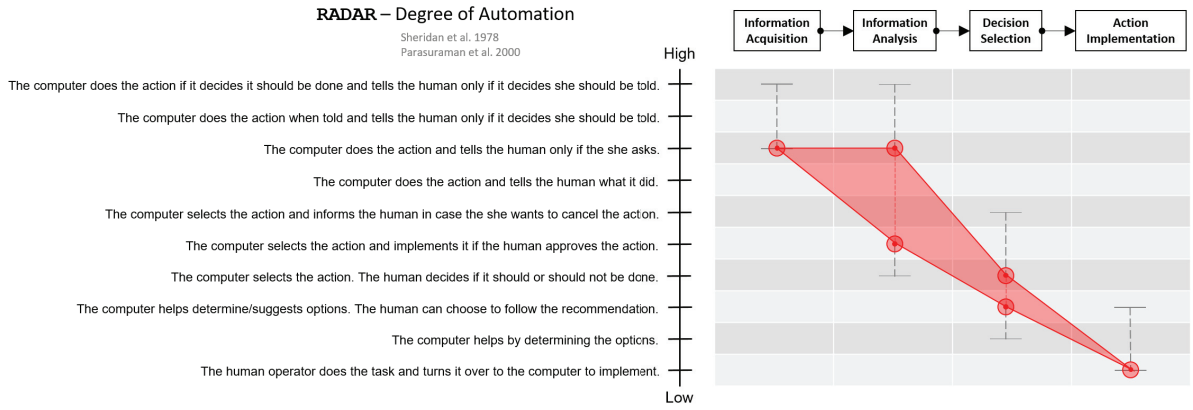

Figure 2: Degrees of automation of the various stages of decision support, and the role of RADAR in it.

in the loop, i.e. appear as suboptimal in her domain. This can occur either when the human or the planner has a faulty model of the world. This is an ideal opportunity to provide model updates or explanations and reconcile this model difference through iterative feedback from the human.

Though having to deal with an incomplete model is the usual case in many mixed initiative settings, i.e. an automated support component, without a full model, cannot actually generate entire plans from scratch but can sometimes complete or critique existing ones (Manikonda et al. 2017). The extent to which a planner can be of help is largely dependent on the nature of the model that is available. 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 more sophisticated decision support. Such technologies can be helpful in complex tasks such as disaster response, where the mental overload of the human can affect the quality of decision making.

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 automated software interfaces 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, to 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.

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 is complex time-critical scenarios (Klein 2008). It is acknowledged as a necessary element in PDS systems (Morrison et al. 2013). Systems which do not support NDM have been found to have detrimental impact on work flow causing frustration to decision makers (Feigh et al.). At the heart of this concept lies, as we discussed before, the requirement of letting the human 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 generate 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, (s)he still has the authority to ask RADAR to explain its plan when it finds it to be inexplicable (Chakraborti et al. 2017a). We postulate that such a system must be augmentable, context sensitive, controllable and adaptive to the humans decisions. Various elements of human-automation interaction such as adaptive nature and context sensitivity are presented in (Sheridan and Parasuraman 2005). (Warm, Parasuraman, and Matthews 2008) show that vigilance requires hard mental work and is stressful via converging evidence from behavioral, neural and subjective measures. Our system may be considered as a part of such vigilance support thereby reducing the stress for the human.

Degrees of Automation One of the seminal works by (Sheridan and Verplank 1978) builds a model that enumerates ten levels of automation in software systems depending on the autonomy of the automated component. Later, in the study of mental workload and situational awareness of humans performing alongside automation software, (Parasuraman 2000) separates automation into four parts- Information Acquisition, Information Analysis, Decision Selection and Action Implementation (see Figure 2). We use this system as an objective 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. (Parasuraman and Manzey 2010) shows that human use of automation may result in automation bias leading to omission and commission errors, which underlines the importance of reliability of the automation (Parasuraman and Riley 1997). Indeed, it is well known (Wickens et al. 2010), that climbing the automation ladder in Figure 2 might well improve operative performance but drastically reduce response quality when failures occur. Hence, to meet the requirement of naturalistic decision making, we observe a downward trend in automation levels (in Figure 2) as we

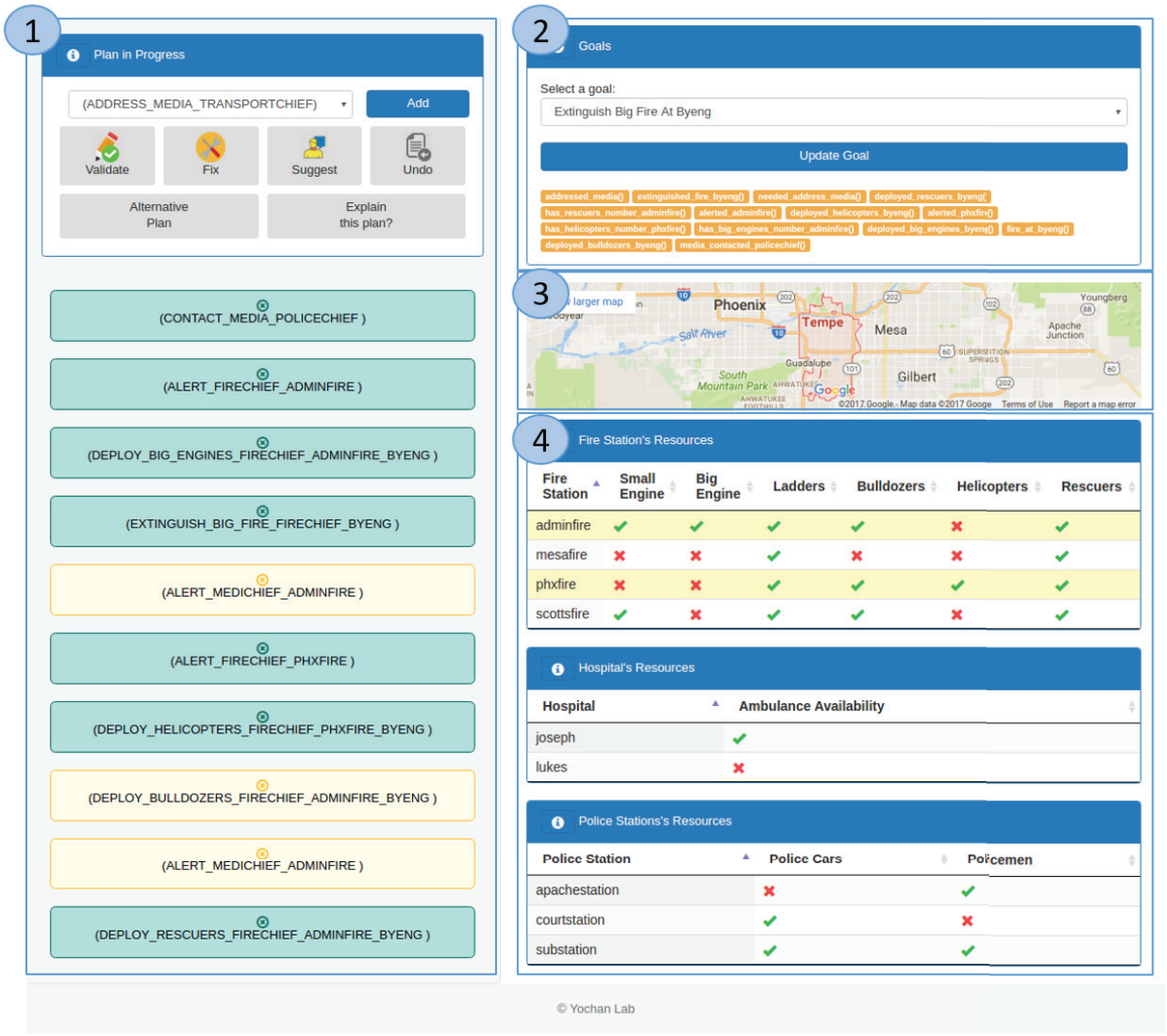

Figure 3: RADAR interface showing decision support for the human commander making plans in response to a fire.

progress from data acquisition and analysis (which machines are traditionally better at) to decision making and execution.

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 (eg. action suggestion, plan critiques). The current system mainly addresses the decision making aspect, which requires the ability to both interpret as well as steer effectively, even as it situates itself in the level of automation it can provide in the context of naturalistic decision making.

RADAR

We will now go into details of the RADAR interface and its integration with planning technologies to enable different forms of proactive decision support. A video walkthrough demonstrating the different capabilities of the system is available at https://goo.gl/c8kk3X.

The Fire-fighting Domain For the remainder of the discussion, we will use a fire-fighting scenario to illustrate our ideas. The domain model used by the system (assumed to be known and available for a well-defined task such as this) is represented in PDDL and is assumed to be very close, if not identical, to that of the expert in the loop. The scenario plays out in a particular location (we use Tempe as a running example) and involves the local fire-fighting chief, who along with the local police, medical and transport authorities, is trying to build a plan in response to the fire using the given platform augmented with decision support capabilities.

Overview of the Interface - This interface, as shown in Fig-

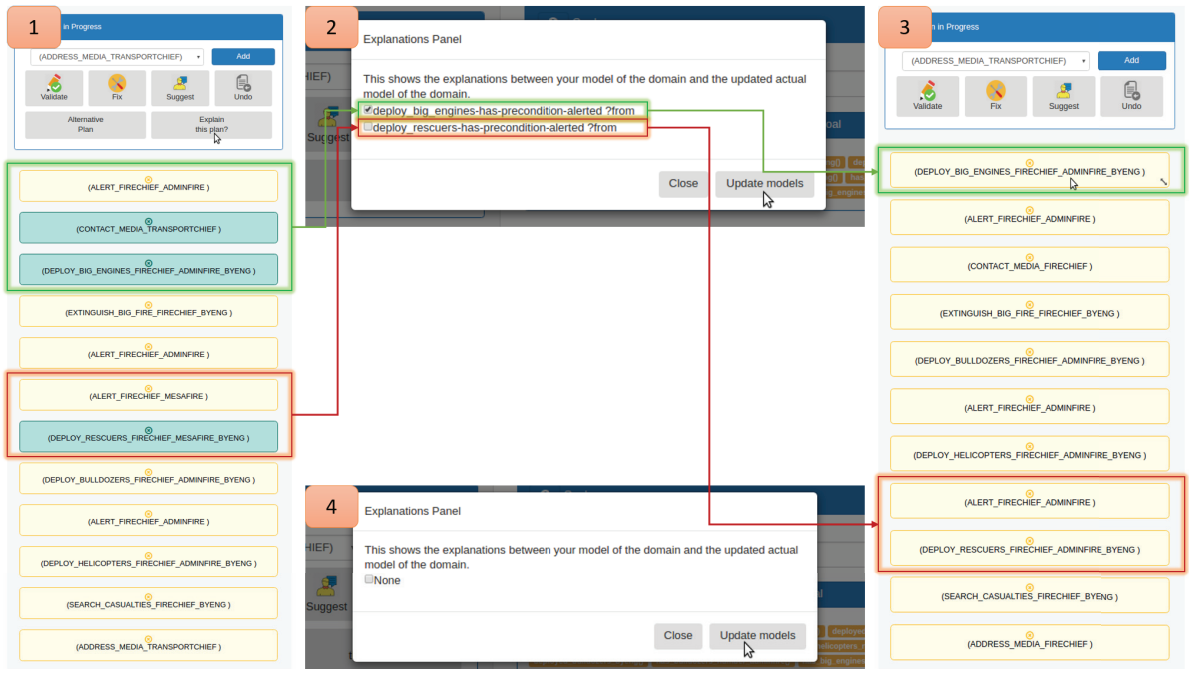

Figure 4: (1) RADAR knows that in the environment, the commander needs to inform the Fire Station's Fire chief before deploying big engines and rescuers. In green, Adminfire's Fire Chief is alerted to deploy big engines from Admin Fire Station. In red, Mesa fire stations' Fire Chief is alerted to deploy 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 it 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.

ure 3, consists of four main components. This includes -

1. **Planning Panel** - This is the most critical part of the system. It displays the plan under construction, and provides the human with abilities to reorder / add / delete actions in the plan, validate a partial plan, fix a broken plan, suggest new better ones, provide explanation on the current one, etc. by accessing the options at the top of the panel. This will be the primary focus for our discussion in the upcoming sections.
2. **Goal Selection Panel** - This lets the user set high level goals or tasks to be accomplished. Once a goal is selected, the system sets up a planning problem instance given its knowledge of the initial state of the world. It also summarizes this task to the user by displaying the necessary landmarks to be attained in order to achieve the goal.
3. **Map Panel** - This provides visual guidance to the decision making process, thereby reducing the information overload and improving the situational awareness of the human. The map can be used to point of areas of interest, location and availability of resources, routes, etc. Note that due to modular design, this part of the User Interface can be used to display other relevant information for different domains by simply changing a template file.
4. **Resource Panel** - The human commanders have access to resources as seen in the tables on the right in Figure 3. For example, the police can deploy police cars and policemen, and the fire chief can deploy fire engines, ladders, rescuers, etc. They can also acquire or update the availability of these on the go by clicking on the red crosses or green tick respectively, if the system's data is stale. The system highlights parts of the table that are relevant to the plan currently under construction – an example of decision-driven data support.

In the following sections, we show how RADAR can help a commander in a disaster response scenario highlighting the degree of automation the software demonstrates in the different stages of the decision support process. The PDDL files can be accessed at https://goo.gl/htrmLQ.

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 will also assume that the system can keep track of drifting models (Bryce, Benton,

and Boldt 2016) in the background, 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 Analysis

Now, we will present details on how the proposed system can leverage planning technologies to provide relevant suggestions and alerts to the human decision maker with regards 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 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 by themselves might not achieve the goal, and this is the role of the plan suffix π_s that is yet to be decided upon. We will demonstrate below how planning technology may be used to shape each of these plan fragments for the better.

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, and Kambhampati 2013) or updated (Bryce, Benton, and Boldt 2016) on the fly in cases 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 et al. 2017), or be able to explain the rationale behind its suggestions if required (Kambhampati 1990; Sohrabi, Baier, and McIlraith 2011) in a fashion easily understood by the human user (Perera et al. 2016).

Often a key factor in these settings is the difference in the planner's model of the domain, and the human expectation of it. Thus, a valid or satisfactory explanation may require a *model reconciliation process* where the human model needs to be updated, as shown in Figure 4, in order to explain a suggestion. Here the system performs model-space search to come up with *minimal explanations* to explain the plan being suggested while at the same time not overloading the human with information not relevant to the task at hand (Chakraborti et al. 2017a). Note that here the human has the power to veto the model update if (s)he believes 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.

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 in the future. Briefly, landmarks (Hoffmann, Porteous, and 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 and 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.

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.

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. $\delta(\mathcal{I}, \pi_e \circ \pi_h) \models \bot$. 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 validators like `VAL` (Fox, Howey, and 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 $s \neq \delta(\mathcal{I}, \pi_e)$.

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

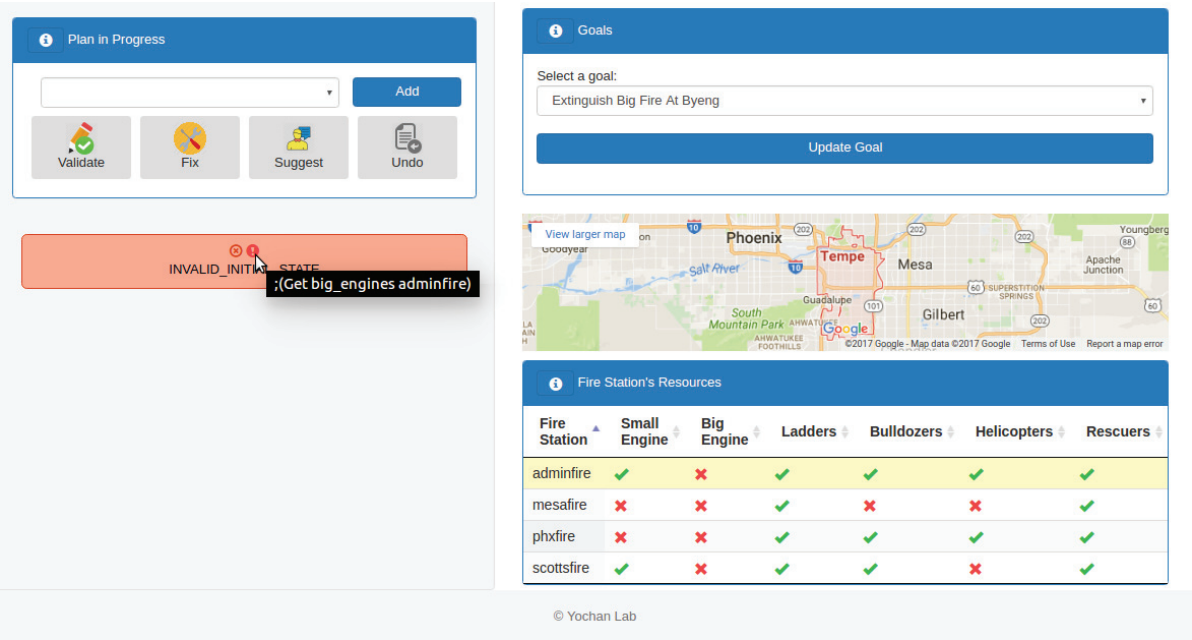

Figure 6: The lack of big engines at all the fire stations results in an initial state for the planning problem from which no plan is possible to achieve the goal of Extinguishing Big Fire at BYENG. RADAR reports this as a warning and suggests the minimal number of resources the commander needs to gather to arrive at a start state from which a plan is actually possible.

using *excuse generation* techniques studied in (Göbelbecker et al. 2010) and *plan revision* problems (Herzig et al. 2014). We achieved this using a slightly modified version of the model-space search technique introduced by (Chakraborti et al. 2017a) - 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.

Decision Selection

The decision selection process is perhaps closest to home for the planning community. Referring back to our discussion on naturalistic decision making, and the need for on-demand support, we note that the system is mostly restricted to Degree 3 and 4 of automation with respect to decision selection. We will go through some salient use cases below.

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 and 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 7 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.

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 and Geffner 2010) for a slightly different purpose than originally intended. Given a known 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 7.

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 $\mathcal{P}$, (Ramírez and Geffner 2010) can be used to 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. This is used by RADAR in determining the response to suggest or fix any hypothesis.

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

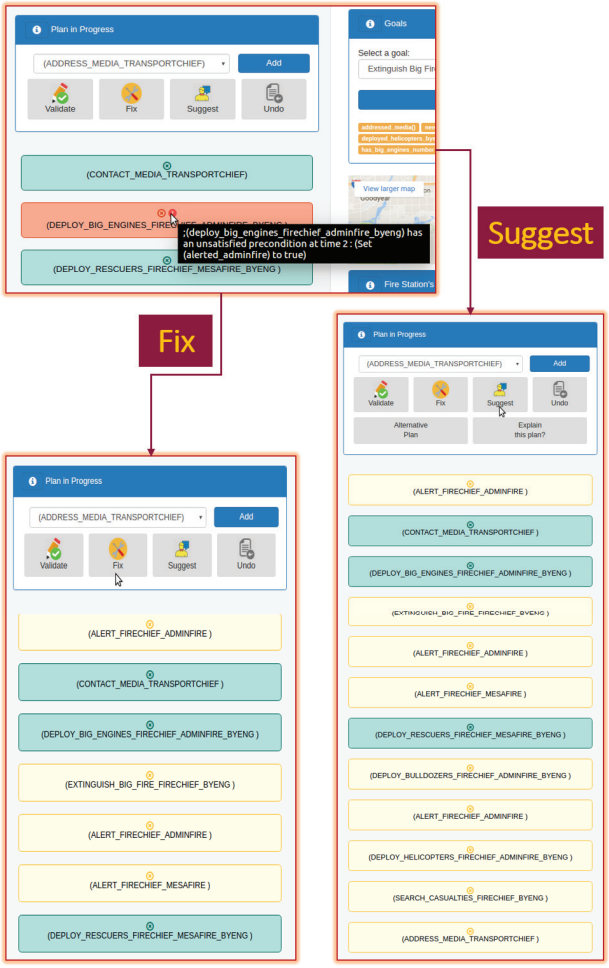

Figure 7: RADAR's 'Fix' button does plan correction, providing action suggestions. The 'Suggest' provides actions and plan suggestions to help achieve the goal.

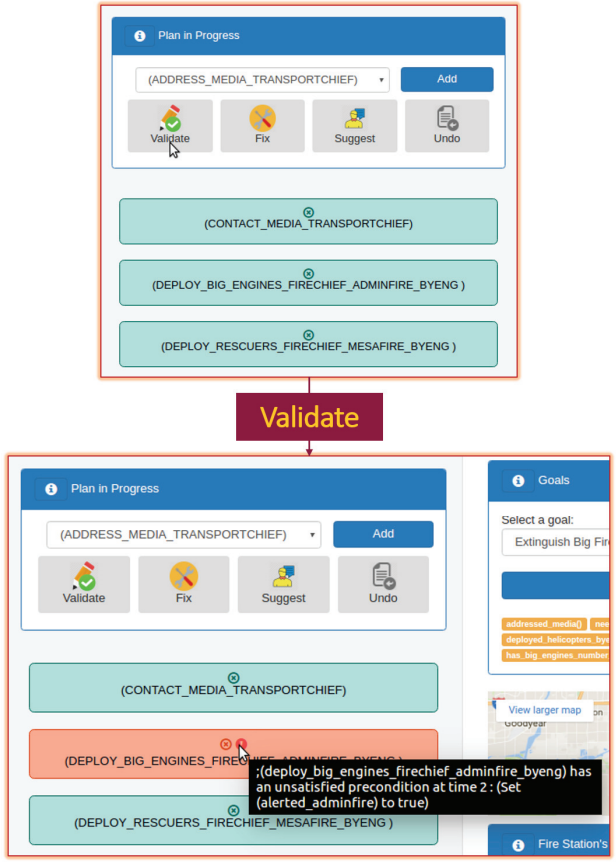

Figure 8: RADAR does plan validation of a partial plan made by the user and shows reasons as to why it is invalid.

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, and Udrea 2014) become useful. We are exploring avenues of integrating these techniques into our current system.

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 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 (Gombolay et al. 2015; Chakraborti et al. 2017b) 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.

Conclusion and Future Work

In conclusion, we motivated the use of automated planning techniques in the role of an assistant in the deliberative process of an expert human decision maker providing a detailed overview of our platform RADAR. We also showed how these capabilities complement the design principles laid out in the Human Computer Interface (HCI) community for such softwares. We look forward to conducting human studies with domain experts to evaluate RADAR's effectiveness in providing data-driven decision support and learning useful metrics to improve its decision-based data gathering aspect.

Acknowledgments This research is supported in part by the NASA grant NNX17AD06G and the ONR grants N00014161-2892, N00014-13-1-0176, N00014-13-1-0519, N00014-15-1-2027. The second author is also supported in part by the IBM Ph.D. Fellowship 2017-18.

References

Ai-Chang, M.; Bresina, J.; Charest, L.; Chase, A.; Hsu, J.-J.; Jonsson, A.; Kanefsky, B.; Morris, P.; Rajan, K.; Yglesias, J.; et al. 2004. Mapgen: mixed-initiative planning and scheduling for the mars exploration rover mission. *IEEE Intell. Sys.*

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

Bryce, D.; Benton, J.; and Boldt, M. W. 2016. Maintaining evolving domain models. In *IJCAI*.

Chakraborti, T.; Sreedharan, S.; Zhang, Y.; and Kambhampati, S. 2017a. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *IJCAI*.

Chakraborti, T.; Talamadupula, K.; Fadnis, K.; Campbell, M.; and Kambhampati, S. 2017b. UbuntuWorld 1.0 LTS – A Platform for Automated Problem Solving & Troubleshooting in the Ubuntu OS.

Feigh, K. M.; Pritchett, A. R.; Jacko, J. A.; and Denq, T. Contextual control modes during an airline rescheduling task. *Journal of Cognitive Engg. & Decision Making*.

Fox, M.; Howey, R.; and Long, D. 2005. Validating plans in the context of processes and exogenous events. In *AAAI*.

Göbelbecker, M.; Keller, T.; Eyerich, P.; Brenner, M.; and Nebel, B. 2010. Coming up with good excuses: What to do when no plan can be found. *Cognitive Robotics*.

Gombolay, M.; Gutierrez, R.; Clarke, S.; Sturla, G.; and Shah, J. 2015. Decision-making authority, team efficiency & human worker satisfaction in mixed human-robot teams. *Autonomous Robots*.

Helmert, M. 2006. The fast downward planning system. *JAIR*.

Herzig, A.; Menezes, V.; de Barros, L. N.; and Wassermann, R. 2014. On the revision of planning tasks. In *ECAI*.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered landmarks in planning. *JAIR*.

Kambhampati, S., and Talamadupula, K. 2015. Human-in-the-loop planning and decision support. In *AAAI Tutorial*.

Kambhampati, S. 1990. A classification of plan modification strategies based on coverage and information requirements. In *AAAI Spring Symposium on Case Based Reasoning*.

Kim, J.; Banks, C. J.; and Shah, J. A. 2017. Collaborative planning with encoding of users' high-level strategies. In *AAAI*.

Klein, G. 2008. Naturalistic decision making. *The Journal of the Human Factors and Ergonomics Society*.

Laskey, K. B.; Marques, H. C.; and da Costa, P. C. G. 2016. High-level fusion for crisis response planning. *Fusion Methodologies in Crisis Management: Higher Level Fusion and Decision Making*.

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

Manikonda, L.; Chakraborti, T.; Talamadupula, K.; and Kambhampati, S. 2017. Herding the crowd: Using automated planning for better crowdsourced planning. *Journal of Human Computation*.

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

Narayanan, V.; Zhang, Y.; Mendoza, N.; and Kambhampati, S. 2015. Automated planning for peer-to-peer teaming and its evaluation in remote human-robot interaction. In *HRI Extended Abstracts*.

Nguyen, T. A.; Do, M.; Gerevini, A. E.; Serina, I.; Srivastava, B.; and Kambhampati, S. 2012. Generating diverse plans to handle unknown and partially known user preferences. *Artif. Intell.*

Parasuraman, R., and 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*.

Parasuraman, R., and Riley, V. 1997. Humans and automation: Use, misuse, disuse, abuse. *Human Factors: The Journal of the Human Factors and Ergonomics Society*.

Parasuraman, R. 2000. Designing automation for human use: empirical studies and quantitative models. *Ergonomics*.

Perera, V.; Selvaraj, S. P.; Rosenthal, S.; and Veloso, M. 2016. Dynamic Generation and Refinement of Robot Verbalization. In *Robot and Human Interactive Communication (RO-MAN)*.

Ramírez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *AAAI*.

Riabov, A.; Sohrabi, S.; and Udrea, O. 2014. New algorithms for the top-k planning problem. In *SPARK Workshop at ICAPS*.

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

Sheridan, T. B., and Verplank, W. L. 1978. Human and computer control of undersea teleoperators. Technical report.

Sohrabi, S.; Baier, J. A.; and McIlraith, S. A. 2011. Preferred explanations: Theory and generation via planning. In *AAAI*.

Warm, J. S.; Parasuraman, R.; and Matthews, G. 2008. Vigilance requires hard mental work and is stressful. *Human Factors: The Journal of the Human Factors and Ergonomics Society*.

Wickens, C. D.; Li, H.; Santamaria, A.; Sebok, A.; and Sarter, N. B. 2010. Stages and levels of automation: An integrated meta-analysis. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*.

Zhang, Y.; Narayanan, V.; Chakraborti, T.; and Kambhampati, S. 2015. A human factors analysis of proactive support in human-robot teaming. In *IROS*.

Zhang, Y.; Sreedharan, S.; Kulkarni, A.; Chakraborti, T.; and Hankz Hankui Zhuo, S. K. 2017. Plan explicability and predictability for robot task planning. In *ICRA*.

Zhu, L., and Givan, R. 2003. Landmark extraction via planning graph propagation. *ICAPS DC*.

Zhuo, H. H.; Nguyen, T.; and Kambhampati, S. 2013. Refining incomplete planning domain models through plan traces. In *IJCAI*.