Planning for Human-Robot Teaming

Kartik Talamadupula[†] and **Subbarao Kambhampati**[†] and **Paul Schermerhorn**[§] and **J. Benton**[†] and **Matthias Scheutz**[§]

[†]**Department of Computer Science**
Arizona State University
Tempe, AZ 85287 USA
{krt,rao, j.benton} @ asu.edu

[§]**Cognitive Science Program**
Indiana University
Bloomington, IN 47406 USA
{pscherme,mscheutz} @ indiana.edu

Abstract

One of the most important applications of planning technology has been – and continues to be – guiding robotic agents in an autonomous fashion through complex problem scenarios. Increasingly, real-world scenarios are evolving in a way that includes humans as actors in the loop along with the robot and the planning system. These humans are stakeholders whose roles may vary between that of a commander or a system or domain expert; the one common thread is that together with the robot, they form a team that shares common goals. In this paper, we consider challenges posed by such *human-robot teaming* scenarios from a purely planning-centric perspective, and discuss the dimensions of variation within application problems in such scenarios. We seek to differentiate planning for human-robot teaming from the general area of human-robot interaction, since we are mainly interested in the planning tools that facilitate such teaming. We look at some problems that are encountered in deploying existing planning techniques in such teaming scenarios, and illustrate these with our experience in a real world search and rescue scenario. We follow this up with results from runs involving a robot controlled by a planner whose goal handling capabilities are augmented.

Introduction

One of the earliest motivations for Artificial Intelligence as a field of study was to provide autonomous control to robotic agents that carry out useful service tasks. Application scenarios for these kinds of tasks span a wide spectrum that includes military drones and mules, household assistance agents (Goebelbecker et al. 2010) and search and rescue robots (Schermerhorn et al. 2009). The concept of *teaming* between humans and robots is central to all these applications – the notion of robotic agents that support a human agent's goals while executing autonomously is a recurring theme. The level of autonomy that is desired of these robotic agents is often achievable only by integrating them with planning systems that can not only plan for the initially specified goals, but also updates to these goals as well as changes to the world and to the agent's capabilities.

Recent years have seen the emergence of fast planning algorithms and systems that can account for a large number of the features that distinguish a real world application from a theoretical scenario – time, cost, resources, uncertainty and execution failure. Though planners of the past have been able to model many of these features (Penberthy and Weld 1995), the scale-up that is required to support real world time windows has only come about in the past decade due to the use of heuristic search methods for plan synthesis. Current planners still operate under a number of restrictive assumptions, and classical planners like LAMA (Richter and Westphal 2010) are clearly the fastest of the lot. The challenge then is one of identifying the features that are essential when considering planning support for such joint human-robot endeavors, and of providing a general framework for these problems. This problem is quite distinct from the existing field of human-robot interaction (HRI), since we are interested more in what existing planning techniques can be used or extended in order to facilitate teaming scenarios. Towards this end, we discuss a new class of problems under the collective term *human-robot teaming* (HRT), and present the essential dimensions of such problems with respect to planning. The teaming aspect of these problems arises from the fact that the human and the robot are both acting towards achieving the same set of shared goals, and the relationship between them can be defined in terms of known modes of interactions in teams (e.g. colleagues, commander-subordinate, etc.). Though there has been work in the past on the intersection of tasks involving humans, robots and planners, most of that work has concentrated on a system-centric view of the interaction. Our focus in this paper is instead on the teaming, and on describing the characteristics of this problem as applicable to planning.

The rest of this paper is organized as follows: we first discuss the concept of human-robot teaming and list the dimensions of interest to planning in such scenarios. Following this, we look at a search and rescue scenario that we had to provide planning support for as a case study, and place it within the HRT spectrum. We then detail some initial work that we have undertaken in tackling some planning challenges inherent in teaming scenarios. We discuss two specific problems – handling incomplete models, and the problem of goal specification and revision. As part of the latter, we also detail our work with a robot executing in a real-world search and rescue scenario, and present the aggregated results of the robot's runs through this task guided by our planning system. Our hope is for this paper to serve as a catalyst that spurs the planning community into further defining and mapping the application-rich field of human-robot teaming, and the specific planning challenges that lie in this area.

Human-Robot Teaming

In this paper, we focus our attention towards the planning challenges that must be tackled in order to support human-robot teaming scenarios. Our motivation in pushing this problem is to present a larger class of applications that the planning community can provide tightly-knit support for, and to open the door to discussions on the nature of such support. Our experience with this problem stems from our work with a search and rescue scenario (detailed in the next section) and the planner extensions that were required in order to support tasks from that scenario. In later sections, we detail some of these extensions along with references to more detailed work.

We begin by clarifying the nomenclature: *human-robot teaming* scenarios are those involving (possibly multiple) human and robotic agents that acquire their "teaming" nature from the autonomous behavior of the robotic agent(s). Though it is assumed that the top-level goals are determined and specified by the human, the robot is completely autonomous in that it only receives a set of goals that it must accomplish within some specified constraints while respecting the notion of optimizing some pre-defined metric. For the robot to exhibit this autonomy, it is imperative that the system be equipped with a planner that can handle the various dimensions of the environment that the team must operate in. We briefly describe these dimensions and the challenges in supporting them here, as a list of the defining components of a human-robot teaming (HRT) scenario:

Scenario One of the most important factors in HRT is the problem scenario in which the team is executing – here, we use "scenario" in a ubiquitous sense to describe the particular task or collection of tasks at hand that the team is interested in solving. More often than not, these scenarios are very close approximations of real-world applications in which essential tasks were carried out by humans before the advent of the human-robot team. As such, there is almost always a large amount of domain knowledge available about the scenario that may be exploited – such knowledge resides either with humans who have been actors in the scenario previously, or in carefully compiled manuals and technical documents. Taking the scenario into consideration when planning for a human-robot teaming problem is key, since it determines the kinds of tasks that must be supported. The scenario also determines the kind of features a planner must support in order to guide the robot in pursuit of the team's goals; we elucidate on this in subsequent sections.

Robot The robot is the central actor in an HRT scenario, since it has upon itself the responsibility of executing all the actions and gathering sensory feedback from the world to relay back to the human team member and the team's repository of knowledge. Teaming must account for the fact that there exist various types of robots with varying capacities, and that the type of robot in use may change according to the scenario. A planning system that is providing support in such HRT scenarios must be able to deal with robots with different capabilities (for e.g., mobile robots versus grippers) and must take into account the constraints that arise. The model of a robot's capabilities determines the actions that may be used in fulfilment of the scenario's top-level goals, and hence this is an imperative feature when evaluating an HRT scenario.

Human (User) The human user is a key player in an HRT scenario, since the robot (and consequently the planner) are often achieving goals specified by the user. These users can belong to one of three general categories, contingent on their level of familiarity with the robot's working and the overall integrated system in use:

1. Novice: A novice user is one who does not understand the intricacies of the system (representation, component interaction etc.) and is merely using the robot (and the system) as an assistant. An example of a scenario featuring such a novice user might be a robot designed to assist the elderly at home – in such scenarios, allowances have to be made in the system for information that may be incompletely or wrongly specified by running extra validation via the robot.
2. Domain Expert: A domain expert is a user who is an authority on the environment that the robot is executing in, and is in a position to pass on new goals and information to the robot as they become available. Such a user if often the dominant force in the human-robot team, and it can generally be assumed that knowledge gleaned from this user is reliable and may be acted upon without further deliberation. Examples of such users include commanders in military or rescue teams who have a very good understanding of their environment, yet may not be at all familiar with the system's internal representations.
3. System Expert: A user who is a system expert exhibits a high degree of familiarity with the integrated system that manages the robot – its setup, the representation scheme used, and its various capabilities and shortcomings. Such a user is often the person that set up the system integration in the first place (or someone who is debugging it); as such, examples from real-world scenarios may be researchers and programmers who are in-charge of maintaining the overall system. System experts may also be domain experts, but this is not necessarily always the case.

Model Management The model is the system's internal representation of the dynamics of the environment and world that it must handle. Various components within the system may have different models at different levels of detail; for example, the planner may maintain a PDDL model of the system, while lower level components like the path-finding apparatus may use a grid-like approximation of the world. On a higher level, it is almost always the case that the robotic and human actors in the team have different versions of the world model, and one of the planner's responsibilities might be to bridge this gap and produce plans that will succeed upon execution. The model is usually encoded before the scenario begins, but may also be provided piecemeal by the human component of the team. Learning is another possibility – though we do not currently support it, learning certain characteristics and parameters of the environment automatically through the robot is a promising direction for future work.

Goal Management Distinct from the notion of encoding and using the world model is the idea of specifying and updating goals. The very utility of a human-robot teaming scenario stems from the assumption that smaller goals will be specified by a human in the pursuit of some overarching purpose, and that the robot will autonomously perform actions in the world that will lead to the fulfilment of those goals. In such scenarios, it is imperative that the definition of "goal" be expanded so that the user may specify different kinds of goals (and possibly plan trajectory constraints as goals (Baral and Zhao 2007)). Since the responsibility of planning for robot's actions falls on the planner, providing means to specify new types of goals is a key planning challenge in such scenarios. Flexibility in goal specification includes supporting ideas like goals with different priorities (using rewards), goals that need not be achieved (using soft goals) and goals that condition on currently unknown facts – and just as important, the ability to specify changes to goals on the fly. We talk about this problem and the work we have done in this direction in detail in a later section.

Communication Given that the team is composed of separate human and robotic components, communication is a non-trivial issue, and is quite often the bottleneck in specifying and achieving goals optimally. Communication affects everything in the scenario, from the specification and modification of the domain model and goals, to the human user's knowledge of the evolution of the world – which may sometimes only be via the robot. The problem with respect to communication is unifying the various representations used by different components in a manner that minimizes information loss and processing time. For example, a commander may only specify information using natural language, but the system must translate this into a form suitable for the planner (and robot) to use; the system must also convey the results of action execution and goal achievement (whether failure or success) to the commander.

Related Work

Though there has been no prior work that directly addresses the problem of planning for human-robot teaming, there is definitely a large volume of work that is related to various aspects of this problem. As shown in Figure 1, previous work can be classified into three parts – human-robot interaction, human-planner interaction and planner-robot interaction. More specifically:

- Planning and execution monitoring deals with the interactions between a fully autonomous robot and a planner.
- Human-robot interaction (HRI) works toward smooth interactions between a human user and a robot.
- Mixed initiative planning relates to interactions between humans who are receiving plans and the automated planners that generate them.

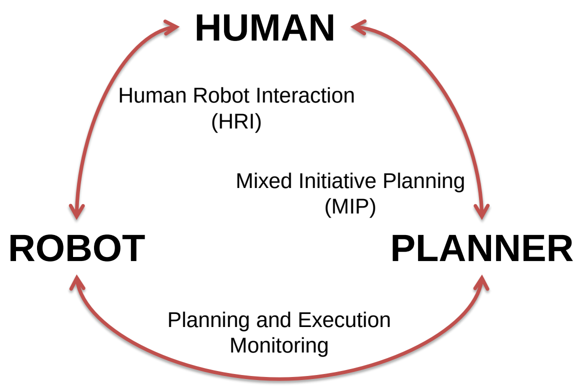
HUMAN
Human Robot Interaction (HRI)
Mixed Initiative Planning (MIP)
ROBOT
PLANNER
Planning and Execution Monitoring

Figure 1: The various modes of interaction in human-robot scenarios.

Since the focus of this work is on providing planning support for human-robot teams, the most interesting work is that which relates planning and execution monitoring to mixed initiative planning. A lot of work has been done in both these areas, and their intersection; the closest work seems to be Bagchi et al.'s (Bagchi, Biswas, and Kawamura 1996) system for controlling service robots. In their system, the robot is equipped to handle the user's changing goals and advice at different levels of detail via a planner that can refine and modify goals dynamically. There has also been work on how humans interact with planners, and how the process of accepting user input can be streamlined. In particular, work by Myers [1996, 1998] has dealt specifically with *advisable planning* that allows a human to specify partial plans, recommendations or methods to evaluate plan quality, all in natural language. Space precludes a detailed description of all past related work; the reader is directed to (Talamadupula et al. 2010a) for a complete listing.

However, planning for human-robot teaming tasks has received a significant amount of attention very recently as well. In particular, two sub-problems have seen a lot of interest. The first is the idea of using two (or more) distinct models during the planning process – a higher, more task-oriented model while trying to come up with actions that support end goals; and a lower-level model to decompose those tasks in tune with the capabilities of the robotic agent being used. The other idea that has received some attention has been that of robotic proactiveness, and the notion that a robot may "ask for help" if it (the planner) is unable to come up with a course of action to fulfil a particular goal. Both these problems have much scope for work, and there exists some work in the planning community currently under review that addresses them.

An HRT Case Study: Search and Rescue

One of the primary applications of human-robot teams is in scenarios where a human actor has a plethora of knowledge about the problem at hand, yet cannot act in the world due to inherent dangers to human life – emergency responders and firefighters are among the best examples of such hu-

Feature \ Task	Search and Report	Reconnaissance	Kitchen Robot
Robot	Mobile	Mobile	Mobile and Manipulator
Human (User)	Domain Expert	System Expert	Novice
Model	Less Dynamic	Dynamic	Highly Dynamic
Goals	Changing	Static	Changing
Communication	Natural Language	APIs	Natural Language

Table 1: The dimensions of a Human-Robot Teaming scenario illustrated for a few example tasks.

mans. In such cases, having a robotic agent as part of the team greatly increases the chances of achieving the desired end-goals (rescuing people, putting out a fire etc.) without exposing the human component of the team to risks.

In this paper, we use a specific scenario that we had to provide planning support for to illustrate the challenges that crop up when planning for human-robot teaming. This is the urban search and rescue (USAR) scenario – a team consisting of a human and a robot is tasked with finding and reporting the location of critical assets (e.g. injured humans) in an urban setting (usually a building). A given USAR scenario may consist of multiple problems, each with different challenges that depend on the end goals that must be satisfied - examples of such problem tasks are given in table 1, along with an analysis of the how the various HRT features apply to these tasks. The human member of the team usually has intimate knowledge of the setting of the scenario, but cannot perform the required tasks due to inherent dangers like fires, gas leaks, collapsed structures etc. Examples of tasks in the USAR scenario include transporting essential materials to a specified location or entity; and reconnaissance tasks like reporting the locations of trapped or injured humans to the commander and taking pictures of objects or areas of interest. In the following, we present two USAR tasks that are of particular interest to us as examples to illustrate the planning challenges that are inherent in human-robot teaming.

Search and Report

In this problem, the robot's main goal is to deliver essential medical supplies within the area of interest – during its run, the robot may be given additional information and goals about other assets. The human component of the team (the commander) has intimate knowledge of the building's layout, but is removed from the scene and can only interact with the robot via on-board wireless communication. The robot begins in a long hallway that has doors leading off into rooms on either side.

Initially, the robot is unaware that that these rooms may contain injured or trapped humans, and its goal is to reach the end of the hallway to deliver the supplies by a given deadline. As the robot executes a plan to achieve that goal, the human commander notes that it is passing the rooms and recalls that injured humans may be trapped in these rooms. The commander then passes on this information linking rooms to injured humans to the robot, and specifies a new goal on reporting the location of as many such humans as possible given the time and resource constraints imposed by the achievement of its original goal. In a succeeding section, we talk about the changes to goal specification that had to be handled to enable a planner to handle such a task, and present some past work on this application.

Reconnaissance

The other task that we touch on is based on a classic robotic application - reconnaissance. Quite often in urban settings, there arise situations where more information is needed about certain objects before making a decision, yet the cost or risk of obtaining that information manually (for humans) is too high. A sterling example is defusal of explosives; often, a closer look and more accurate imagery is required in order to determine whether an unrecognized objects is dangerous and should be disposed off in a suitable manner. Given the high costs of adverse events in such scenarios, it is much more preferable to have robotic agents do the close-up examination required. The challenge in these scenarios is to support changes to the model of the robotic agent's capabilities as conveyed to the planner – it is entirely possible that given a new task like taking a zoomed-in picture of a suspicious box, the commander may either give the robot new effectors that accomplish such a task, or may simply specify a way of using the robot's existing capabilities to simulate such an effect. These are changes to the model that the planner is planning with, and hence need to be accomodated to enable goal achievement.

Incomplete Models

We now turn our attention to the first of two important planning challenges that arise when supporting human-robot teaming scenarios, that of changes to the model while the agent is executing in the world. Take the example of the first task in the USAR scenario – search and report – where the human commander is removed from the scene due to the inherent dangers of the situation. The agent thus needs to act in an autonomous manner to achieve the goals prescribed to it. To this end, the agent follows a domain theory that is provided by a domain expert; however, *updates* to this domain may be specified while the agent is executing a plan in the world. Updates to the agent's knowledge of the world may also arrive in tandem with execution. In such circumstances, two things are of essence: first, we need a representation for *specifying* such updates and integrating them into the knowledge base of the planner that is guiding the agent. Subsequent to this, the problem changes to one of *reasoning* about the changes and their effect on the current plan's validity and metrics. Replanning from scratch is a trivial approach – however, with sophisticated update methods and reward models, such a method can account for com-

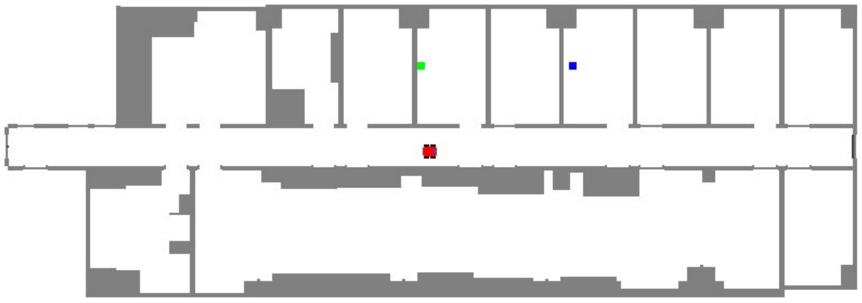

Figure 2: A map of a sample search and report scenario; boxes in rooms are stand-ins for humans, where green (at left) indicates injured and blue (at right) indicates normal.

mitments (Cushing, Benton, and Kambhampati 2008). Nevertheless, such methods ignore the fact that many changes are localized to a certain portion of the domain and may not require the (often expensive) re-computation of a new plan.

As automated planning systems move into a support role for real world problems that involve such teaming, the problem of incompletely or incorrectly specified domain theories is a recurring one. The shortcomings associated with not having a completely specified domain theory manifest themselves as reduced robustness in synthesized plans, and subsequent failures during the execution of such plans in the world. One way of dealing with such contingencies is to employ a reactive approach that replans from scratch (as mentioned above) – however, this approach will fail if there are parts of the domain model that are never revealed to the system (and planner) at all. Consider for example the act of opening doors that are locked, yet the planner's model does not support the notion of locks on doors. A reactive planner would keep trying an 'open' action with no success, and very little information in terms of why the action was failing.

More generally, it is the case in many scenarios that though plan synthesis is performed using a rudimentary domain model and less than complete information about the world, there are domain experts who specify changes to the specific problem instance and sometimes the domain model itself during the planning process. Quite often it is useful to take this new information into account, since it may help prevent execution failures when the plan is put into action. Additionally, new information about the domain or the problem may open up new ways of achieving the goals specified, thus resulting in better plan quality as well as more robust plans. Given the progress in automated planning technology and a planning system that can be engineered to handle such changes, it would be wasteful to stick to reactionary planning and not exploit the plan improvements that are possible when changes in the model are taken into account.

Model Management

As described above, model updates are a reality that many real world integrated systems have to contend with. When viewed through the prism of the search and rescue scenario that we support, the process of updating these models breaks down into one of two kinds of tasks:

1. **Model Maintenance**: This is a more complete solution to the problem of model updates, founded in the formal representation of domain models. The expectation in this kind of update is that the domain expert will specify changes to the model that will ultimately be in the same format that is used for representation. In this type of update, post-update consistency of the model is a trivial issue since the specification mechanisms will ensure that only permissible updates can be specified. This kind of update is suited to scenarios where the planner is part of a larger integrated system and communicates via some kind of API that supports the planner's internal representation.
2. **Model Revision**: This approach seeks to incorporate changes that are specified in a representation that is less formal than the one used by the planner. An example of such a scenario would be a human commander specifying (via natural language) that there have been certain changes to the problem at hand – the planner needs to make the best it can of these updates and change the current plan accordingly. Note that it is significantly harder to offer guarantees about goal achievement, plan validity, or model consistency with this approach.

In the context of our current work, the first scenario is more relevant - even though the updates to the domain are specified through natural language, the planner only receives them via an established API that allows interaction with the the agent architecture. However, the bigger point is that the the kinds of update tasks that one has to consider in these kinds of scenario are really a manifestation of the agents in the system and the architecture underlying it. Most of the changes to the current world state can be described as part of a specific problem description and changes to it. However, there is another kind of update that is possible; an update to the domain model. It is quite unlikely that these kinds of updates are "discovered" as changes to the world; the more likely eventuality is that such updates are specified to the planner by a domain expert – perhaps even the person who crafted the domain in the first place. Domain design is not an

Run	Cost	Reward	Soft	Enter R_1	Report GB	Enter R_2	Report BB	Enter R_3
1	+	+	+	Yes	Yes	No	No	No
2	+	+	-	Yes	Yes	$\perp$	$\perp$	$\perp$
3	+	-	+	No	No	No	No	No
4	+	-	-	Yes	Yes	$\perp$	$\perp$	$\perp$
5	-	+	+	Yes	Yes	No	No	No
6	-	+	-	Yes	Yes	$\perp$	$\perp$	$\perp$
7	-	-	+	No	No	No	No	No
8	-	-	-	Yes	Yes	$\perp$	$\perp$	$\perp$

Table 2: Results of trial runs with a deadline of 100 time units. $\perp$ denotes that there is no feasible plan from that point on that fulfils all hard goals.

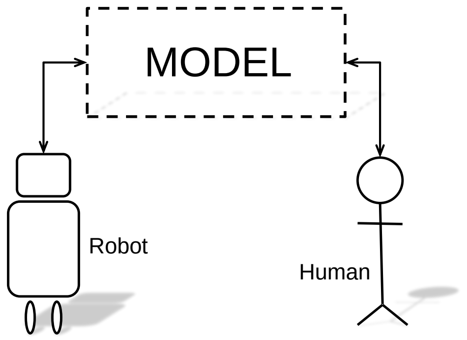

Figure 3: The robot and the human user share a model, but have different perceptions of that model.

exact science, and creating even an approximation of a real world model is fraught with errors of omission and commission. However, most domains are designed such that the first few versions are never completely off-base. Very rarely is there a need to change a significant percentage of a domain model, and more often than not, changes are updates to small portions of the description.

This is especially true in human-robot teaming scenarios like search and rescue – (human) domain experts are more likely to provide additional information that is relevant to the immediate tasks that are being performed. In terms of symbolic planning, this translates into the operators that are currently being executed as part of the overall plan. In such scenarios, it makes more sense to provide a way of updating the existing domain description and the plan that is currently executing than to throw out all the search effort and replan from scratch, since the changes to the domain may not affect a significant portion of the plan. In addition, this kind of approach is preferable even in scenarios where domain descriptions are learned (and updated) automatically through repeated planning and execution episodes.

Goal and Knowledge Revision

Along with information about the model, the other important user-specified information that the robot (and planner) must deal with is the specification of the goals that must be achieved in a given task. This goal specification may the actual goals to be achieved, as well as the values of achieving such goals, and priorities and deadlines (if any) associated with these goals. The fact that the system's goals are determined and specified by the human in the loop also introduces the possibility that goals may be specified incompletely or incorrectly at the beginning of the scenario. Such a contingency mandates a need for a method via which goals, and the knowledge that is instrumental in achieving them, can be updated.

The biggest planning challenge when it comes to the problem of goal update and revision is that most state-of-the-art planning systems today assume a "closed world" (Etzioni, Golden, and Weld 1997). Specifically, planning systems expect full knowledge of the initial state, and expect up-front specification of all goals. Adapting them to handle the "open worlds" that are inherent in real-world scenarios presents many challenges. The open world manifests itself in the system's incomplete knowledge of the problem at hand; for example, in the search and report scenario, neither the human nor the robot know where the injured humans may be. Thus an immediate ramification of the open world is that goals may often be conditioned on particular facts whose truth values may be unknown at the initial state. For example, the most critical goal in the USAR scenario – reporting the location of injured humans – is conditioned on finding injured humans in the first place. In this section, we describe recent work on bridging the open nature of the world with the closed world representation of the planner that has been done in the context of the USAR problem.

Open World Quantified Goals

Open world quantified goals (OWQG) (Talamadupula et al. 2010b) combine information about objects that *may be* discovered during execution with partial satisfaction aspects of the problem. Using an OWQG, the domain expert can furnish details about what new objects may be encountered through sensing and include goals that relate directly to the sensed objects. Newly discovered objects may enable the achievement of goals, granting the opportunity to pursue reward. For example, detecting a victim in a room will allow the robot to report the location of the victim (where reporting gives reward). Given that reward in our case is for each reported injured person, there exists a quantified goal that must be allowed partial satisfaction. In other words, the universal base, or total grounding of the quantified goal on the real

Run	Cost	Reward	Soft	Enter R_1	Report GB	Enter R_2	Report BB	Enter R_3
9	+	+	+	Yes	Yes	Yes	No	Yes
10	+	+	-	Yes	Yes	Yes	No	Yes
11	+	-	+	No	No	No	No	No
12	+	-	-	Yes	Yes	Yes	No	Yes
13	-	+	+	Yes	Yes	Yes	No	Yes
14	-	+	-	Yes	Yes	Yes	No	Yes
15	-	-	+	No	No	No	No	No
16	-	-	-	Yes	Yes	Yes	No	Yes

Table 3: Results of trial runs with a deadline of 200 time units.

world, may remain unsatisfied while its component terms may be satisfied.

As an example, we present an illustration from our scenario: the robot is directed to "report the location of all injured humans". This goal can be classified as open world, since it references objects that do not exist yet in the planner's object database; and it is quantified, since the robot's objective is to report *all* victims that it can find. In our syntax, this information is encoded as follows:

```
(:open
  (forall ?z - zone
    (sense ?hu - human
     (looked_for ?hu ?z)
      (and (has_property ?hu injured)
           (in ?hu ?z))
  (:goal (reported ?hu injured ?z)
           [100] - soft))))
```

We evaluated the efficacy of OWQGs via experimental runs in the search and report scenario introduced earlier. We used the planner SapaReplan (Cushing, Benton, and Kambhampati 2008), an extension of the metric temporal planner Sapa (Do and Kambhampati 2003) in order to implement and test the OWQGs. The task at hand was the following: the robot is required to deliver essential supplies (which it is carrying) to the end of a long hallway – this is a hard goal. The hallway has doorways leading off into rooms on either side, a fact that is unknown to the robot initially. When the robot encounters a doorway, it must weigh (via the planner) the action costs and goal deadline (on the hard delivery goal) in deciding whether to pursue a search through the doorway. In the runs described here, green boxes acted as stand-ins for victims, whereas blue boxes denoted healthy people (whose locations need not be reported) as shown in figure 2. The experimental setup consisted of three rooms, which we represent as R_1, R_2 and R_3. The room R_1 contained a green box (GB), representing a victim; R_2 contained a blue box (BB), representing a healthy person; and R_3 did not contain a box. The respective doorways leading into the three rooms R_1 through R_3 are encountered in order as the robot traverses from the beginning of the hallway to its end.

To achieve our goal of demonstrating the use of the OWQGs, we conducted a set of experiments where we varied four parameters – each of which could take on one of two values – thus giving us 16 different experimental conditions through the scenario. The factors that we varied were:

1. **Hard Goal Deadline**: The hard goal deadline was fixed at 100 time units, resulting in the runs in Table 2, and 200 time units to give the runs in Table 3.
2. **Cost**: Presence or absence of action costs to demonstrate the inhibiting effect of costly sensing actions on the robot's search for injured people.
3. **Reward**: Presence or absence of a reward for reporting injured people in rooms.
4. **Goal Satisfaction**: Label the goal of reporting injured people as either soft or hard, thus modulating the bonus nature of such goals.

In the tables provided, a + symbol stands for the presence of a certain feature, while a - denotes its absence. For example, run number 5 from table 2 denotes an instance where the deadline on the hard goal (going to the end of the hallway) was 100 time units, action costs were absent, the open world goal of reporting people carried reward, and this goal was classified as soft.

The experimental runs detailed in this section were obtained on a Pioneer P3-AT robot as it navigated the USAR scenario with the initial hard goal of getting to the end of the hallway, while trying to accrue the maximum net benefit possible from the additional soft goal of reporting the location of injured people. The robot starts at the beginning of the hallway, and initially has a plan for getting to the end in fulfilment of the original hard goal. An update is sent to the planner whenever a doorway is discovered, and the planner subsequently replans to determine whether to enter that doorway. In the first set of runs, with a deadline of 100 units on being at the end of the hallway, the robot has time to enter only the first room, R_1 (before it must rush to the end of the hallway in order to make the deadline on the hard goal).

In spite of this restriction, the robot exhibits some interesting behavior. The planner directs the robot to enter R_1 in all the runs except 3 and 7 – this can be attributed to the fact that there is no reward on reporting injured people in those cases, and the reporting goal is soft; hence the planner does not consider it worthwhile to enter the room and simply ignores the goal on reporting. The alert reader may ask why it is not the case that entering R_1 is skipped in runs 4 and 8 as well, since there is no reward on reporting injured people in those cases either; however, it must be noted that this goal is hard in cases 4 and 8, and hence the planner *must* plan to achieve it (even though there may be no injured person in that room, or reward to offset the action cost). This example illustrates the complex interaction between the various facets of a typical HRT scenario – the robot's capbilities, user-specified goals, and model parameters like costs and rewards.

In terms of computational performance, the planning time taken by the planning system was typically less than one second (on the order of a hundred milliseconds). Our empirical experience thus suggests that the planning process always ends in a specific, predictable time frame in this scenario—an important property when actions have temporal durations and goals have deadlines. Additionally, in order to test the scale-up of the system, we evaluated it on a problem instance with ten doors (and consequently more runtime objects) and found that there was no significant impact on the performance.

Conclusion

In this paper, we presented the problem of providing planning support for human-robot teaming scenarios, and outlined some of the prominent planning challenges that must be addressed in conjunction with this problem. We presented the Urban Search and Rescue scenario as a case study in which a planner actively supports a human-robot team, and showed the dimensions along which two tasks that are part of this scenario require planning support. Motivated by our applied work in this scenario, we delved deeper into two particular planning challenges – model management and dealing with incomplete and dynamic models, and the problem of goal and knowledge revision. In these sections, we outlined the extensions that had to be made to existing planning technology in order to fulfill the support role that the planner plays in the team. We concluded with a look at results from a search and report task from the USAR scenario where our planner was able to guide the robot's pursuit of fairly complex and dynamic goals.

Acknowledgements

We thank William Cushing for helpful discussions and the development of *Sapa Replan*. We also thank the anonymous reviewers for their suggestions which were incorporated in the final version of this paper. This research is supported in part by ONR grants N00014-09-1-0017 and N00014-07-1-1049, and the NSF grant IIS-0905672.

References

Bagchi, S.; Biswas, G.; and Kawamura, K. 1996. Interactive task planning under uncertainty and goal changes. *Robotics and Autonomous Systems* 18(1):157–167.

Baral, C., and Zhao, J. 2007. Non-monotonic temporal logics for goal specification. In *IJCAI*, volume 7, 236–242.

Cushing, W.; Benton, J.; and Kambhampati, S. 2008. Replanning as a Deliberative Re-selection of Objectives. *Arizona State University CSE Department TR*.

Do, M., and Kambhampati, S. 2003. Sapa: A multi-objective metric temporal planner. *Journal of Artificial Intelligence Research* 20(1):155–194.

Etzioni, O.; Golden, K.; and Weld, D. S. 1997. Sound and efficient closed-world reasoning for planning. *AIJ* 89(1-2):113–148.

Goebelbecker, 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.

Myers, K. 1996. Advisable planning systems. *Advanced Planning Technology* 206–209.

Myers, K. 1998. Towards a framework for continuous planning and execution. In *Proceedings of the AAAI Fall Symposium on Distributed Continual Planning*.

Penberthy, J., and Weld, D. 1995. Temporal planning with continuous change. In *Proceedings of the national conference on Artificial Intelligence*, 1010–1010. JOHN WILEY & SONS LTD.

Richter, S., and Westphal, M. 2010. The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research* 39(1):127–177.

Schermerhorn, P.; Benton, J.; Scheutz, M.; Talamadupula, K.; and Kambhampati, S. 2009. Finding and exploiting goal opportunities in real-time during plan execution. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems.*

Talamadupula, K.; Benton, J.; Kambhampati, S.; Schermerhorn, P.; and Scheutz, M. 2010a. Planning for human-robot teaming in open worlds. *ACM Transactions on Intelligent Systems and Technology (TIST)* 1(2):14.

Talamadupula, K.; Benton, J.; Schermerhorn, P.; Scheutz, M.; and Kambhampati, S. 2010b. Integrating a Closed-World Planner with an Open-World Robot. In *AAAI 2010.*