Planning with Resource Conflicts in Human-Robot Cohabitation

Tathagata Chakraborti[1], Yu Zhang[1], David E. Smith[2], Subbarao Kambhampati[1]
[1]Department of Computer Science, Arizona State University, AZ
[2]Autonomous Systems & Robotics Intelligent Systems Division, NASA Ames Research Center, CA
tchakra2@asu.edu, yzhan442@asu.edu, david.smith@nasa.gov, rao@asu.edu

ABSTRACT

In order to be acceptable members of future human-robot ecosystems, it is necessary for autonomous agents to be respectful of the intentions of humans cohabiting a workspace and account for conflicts on shared resources in the environment. In this paper we build an integrated system that demonstrates how maintaining predictive models of its human colleagues can inform the planning process of the robotic agent. We propose an Integer Programming based planner as a general formulation of this flavor of "human-aware" planning and show how the proposed formulation can be used to produce different behaviors of the robotic agent, showcasing compromise, opportunism or negotiation. Finally, we investigate how the proposed approach scales with the different parameters involved, and provide empirical evaluations to illustrate the pros and cons associated with the proposed style of planning.

1. INTRODUCTION

In environments where multiple agents are working independently, but utilizing shared resources, it is important for these agents to model the intentions and beliefs of other agents so as to act intelligently and prevent conflicts. In cases where some of these agents are human, as in the case of assistive robots in household environments, these are required (rather than just desired) capabilities of robots in order for them to be considered "socially acceptable" - this has been one of the important objectives of "human-aware" planning, as evident from existing literature in human-aware path planning [13, 10] and human-aware task planning [5, 9, 3, 15]. An interesting aspect of many of these scenarios, is the presence of many of the aspects of multi-agent environments, but absence of typical assumptions often made in explicit teaming scenarios between humans and robots, as pointed out in [4]. Probabilistic plan recognition plays an important role in this regard, because by not committing to a plan, that presumes a particular plan for the other agent, it might be possible to minimize suboptimal (in terms of redundant or conflicting actions performed during the execution phase) behavior of the autonomous agent.

Here we look at possible ways to minimize such suboptimal behavior by ways of *compromise*, *opportunism* or *negotiation*. Specifically, we ask the question, what information can be extracted from the predicted plans, and how this information can be used to guide the behavior of the autonomous agent. There has been previous work [1, 6] on some of the modeling aspects of the problem, in terms of planning with uncertainty in resources and constraints. In this paper we provide an integrated framework (shown in Figure 1) for achieving these behaviors of the autonomous agent, particularly in the context of stigmergic coordination of human-robot cohabitation. To this end, we modularize our architecture so as to handle the uncertainty in the environment separately with the planning process, and show how these individual modules interact with each other by the way of usage profiles of the concerned resources.

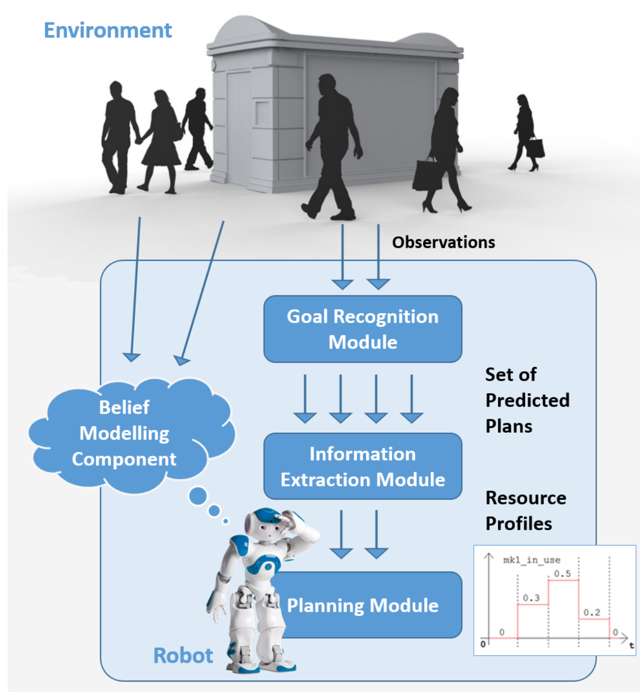


Figure 1: Schematic diagram of our integrated system for belief modeling, goal recognition, information extraction and planning. The robot maintains a belief model of the environment, and uses observations from the environment to extract information about how the world may evolve, which is then used to drive its own planning process.

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

The general architecture of the system is shown in Figure 1. The autonomous agent, or the robot, is acting (with independent goals) in an environment co-habited with other agents (humans), who are similarly self-interested. The robot has a model of the other agents acting independently in its environment. These models may be partial and hence the robot can only make uncertain predictions on how the world will evolve with time. However, the resources in the environment are limited and are likely to be constrained by the plans of the other agents. The robot thus needs to reason about the future states of the environment in order to make sure that its own plans do not produce conflicts with respect to the plans of the other agents.

With the involvement of humans, however, the problem is more skewed against the robot, because humans would expect a higher priority on their plans - robots that produce plans that clash with those of the humans, without any explanation, would be considered incompatible for such an ecosystem. Thus the robot is expected to follow plans that preserve the human plans, rather than follow a globally optimal plan for itself. This aspect makes the current setting distinct from normal human robot teaming scenarios and produces a number of its own interesting challenges. *How does the robot model the human's behavior? How does it plan to avoid friction with the human plans? If it is possible to communicate, how does it plan to negotiate and refine plans?* These are the questions that we seek to address in this work. Our approach models human beliefs and defines resource profiles as abstract representations of the plans predicted on the basis of these beliefs. The robot updates its beliefs upon receiving new observations, and passes on the resultant profiles onto its planner, which uses an IP-formulation to minimize the overlap between these resource profiles and those produced by the human's plans.

The contribution of our paper is thus three-fold, we (1) propose *resource profiles* as a concise mode of representing different types of information from predicted plans; (2) develop an IP-based planner that can utilize this information and provide different modalities of conformant behavior; and (3) provide an integrated framework that supports the proposed mode of planning - the modular approach also provides an elegant way to handle different challenges separately (e.g. uncertainty and/or nested beliefs of humans leaves the planner). The planner, as a consequence of these, has properties not present in existing planners - for example, the work that probably comes closest is [9] that models a specific case of compromise only, while the formulation is also likely to blow up in presense of large hypothesis sets due to absence of concise representation techniques like the profiles. We will discuss the trade-offs and design choices in more detail in the evaluation sections.

The rest of the paper is organized as follows. We will start with a brief introduction of the agent models that comprise the belief component, and describe how it facilitates plan recognition. Then, in Sections 2.3 and 2.4, we are going to go into details of how resource profiles may be used to represent information from predicated plans, and describe how our planner converts this information into constraints that can be solved as an integer program during the plan generation process. In Section 3 we will demonstrate how the planner may be used to produce different modes of autonomous behavior. Finally in Section 4 we will provide empirical evaluations of the planner's internal properties.

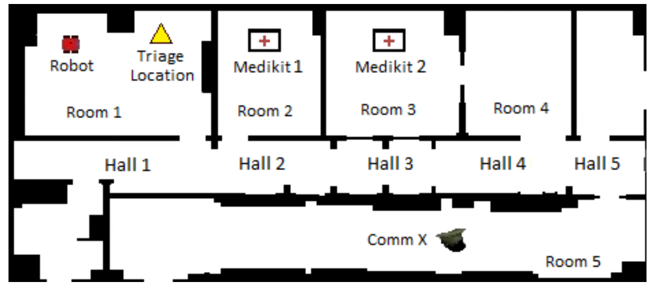


Figure 2: Use case - Urban Search And Rescue (USAR).

2. PLANNING WITH CONFLICTS ON SHARED RESOURCES

We will now go into details about each of the modules shown in Figure 1. The setting (adopted from [14]) involves a commander `CommX` and a robot in a typical USAR (Urban Search and Rescue) task illustrated in Figure 2. The commander can perform triage in certain locations, for which he needs the medkit. The robot can also fetch medkits if requested by other agents (not shown) in the environment. The shared resources here are the two medkits - some of the plans the commander can execute will lock the use of and/or change the position of these medkits, so that from the set of probable plans of the commander we can extract a probability distribution over the usage (or even the position) of the medkit over time based on the fraction of plans that conform to these facts. These *resource availability profiles* (i.e. the distribution over the usage or position of the medkit evolving over time) provide a way for the agents to minimize conflicts with the other agents. Before going into details about the planner that achieves this, we will first look at how the agents are modeled and how these profiles are computed in the next section.

2.1 The Belief Modeling Component

The notion of modeling beliefs introduced by the authors in [14] is adopted in this work. Beliefs about state are defined in terms of predicates $bel(\alpha, \phi)$, where α is an agent with belief $\phi = true$. Goals are defined by predicates $goal(\alpha, \phi)$, where agent α has a goal ϕ. The set of all beliefs that the robot ascribes to α together represents the perspective for the robot of α. This is obtained by a belief model Bel_α of agent α, defined as $\{ \phi \mid bel(\alpha, \phi) \in Bel_{self} \}$, where Bel_{self} are the first-order beliefs of the robot (e.g., $bel(self, at(self, room1))$). The set of goals ascribed to α is similarly described by $\{goal(\alpha, \phi) | goal(\alpha, \phi) \in Bel_{self}\}$.

Next, we turn our attention to the domain model D_α of the agent α that is used in the planning process. We use PDDL [11] style agent models for the rest of the discussion, but most of the analysis easily generalizes to other related modes of representation. Formally, a planning problem $\Pi = \langle D_\alpha, \pi_\alpha \rangle$ consists of the domain model D_α and the problem instance π_α. The domain model of α is defined as $D_\alpha = \langle T_\alpha, V_\alpha, S_\alpha, A_\alpha \rangle$, where T_α is a set of object types; V_α is a set of variables that describe objects that belong to types in T_α; S_α is a set of named first-order logical predicates over the variables V_α that describe the state; and A_α is a set of operators available to the agent. The action models $a \in A_\alpha$ are represented as $a = \langle \mathbb{N}, \mathbb{C}, \mathbb{P}, \mathbb{E} \rangle$ where $\mathbb{N}$ denotes the name of that action; $\mathbb{C}$ is the cost of that action; $\mathbb{P} \subseteq S_\alpha$ is the list of pre-conditions that must hold for the action a to be applicable in a particular state $s \subseteq S_\alpha$ of the environment; and $\mathbb{E}_a = \langle eff^+(a), eff^-(a) \rangle$, $eff^{\pm}(a) \subseteq S_\alpha$ is a tuple that contains the add and delete effects of applying the action to a state. The transition function $\delta(\cdot)$ determines the next state after the application of action a in state s as $\delta(a, s) \models \bot$ if $\exists f \in \mathbb{P}$ s.t. $f \notin s; \delta(a, s) \models (s \setminus eff^-(a)) \cup eff^+(a)$ otherwise.

The belief model, in conjunction with beliefs about the goals / intentions of another agent, will allow the robot to instantiate a planning problem $\pi_\alpha = \langle \mathbb{O}_\alpha, \mathbb{I}_\alpha, \mathbb{G}_\alpha \rangle$, where $\mathbb{O}_\alpha$ is a set of objects of type $t \in T_\alpha$; $\mathbb{I}_\alpha$ is the *initial state* of the world, and $\mathbb{G}_\alpha$ is a set of *goals*, which are both sets of the predicates from S_α initialized with objects from $\mathbb{O}_\alpha$. First, the initial state $\mathbb{I}_\alpha$ is populated by all of the robot's initial beliefs about the agent α, i.e. $\mathbb{I}_\alpha = \{\phi \mid bel(\alpha, \phi) \in Bel_{robot}\}$. Similarly, the goal is set to $\mathbb{G}_\alpha = \{\phi \mid goal(\alpha, \phi) \in Bel_{robot}\}$. Finally, the set of objects $\mathbb{O}_\alpha$ consists of all the objects that are mentioned in either the initial state, or the goal description: $\mathbb{O}_\alpha = \{o \mid o \in (\phi \mid \phi \in (\mathbb{I}_\alpha \cup \mathbb{G}_\alpha))\}$. The solution to the planning problem is an ordered sequence of actions or *plan* given by $\pi_\alpha = \langle a_1, a_2, \ldots, a_{|\pi_\alpha|} \rangle$, $a_i \in A_\alpha$ such that $\delta(\pi_\alpha, \mathbb{I}_\alpha) \models \mathbb{G}_\phi$, where the cumulative transition function is given by $\delta(\pi, s) = \delta(\langle a_2, a_3, \ldots, a_{|\pi|} \rangle, \delta(a_1, s))$. The cost of the plan is given by $C(\pi_\alpha) = \sum_{a \in \pi_\alpha} \mathbb{C}_a$ and the optimal plan π^*_α is such that $C(\pi^*_\alpha) \leq C(\pi_\alpha)\ \forall \pi_\alpha$ with $\delta(\pi_\alpha, \mathbb{I}_\alpha) \models \mathbb{G}_\alpha$. This planning problem instance (though not directly used in the robot's planning process) enables the goal recognition component to solve the compiled problem instances. More on this in the next section.

2.2 The Goal Recognition Component

For many real world scenarios, it is unlikely that the goals of the humans are known completely, and that the plan computed by the planner is exactly the plan that they will follow. We are only equipped with a *belief* of the likely goal(s) of the human - and this may not be a full description of their actual goals. Further, in the case of an incompletely specified goal, there might be a set of *likely* plans that the human can execute, which brings into consideration the idea of incremental goal recognition over a possible goal set given a stream of observations.

2.2.1 Goal Extension

To begin with, it is worth noting that the robot might have to deal with multiple plans even in the presence of completely specified goals (even if the other agents are fully rational). For example, there may be multiple optimal ways of achieving the same goal, and it is not obvious beforehand which one of these an agent is going to end up following. In the case of *incompletely* specified goals, the presence of multiple likely plans become more relevant. To accommodate this, we extend the robot's current belief of an agent α's goal, $\mathbb{G}_\alpha$, to a *hypothesis goal set* Ψ_α. The computation of this goal set can be done using the planning graph method [2]. In the worst case, Ψ_α corresponds to all possible goals in the final level of the converged planning graph. Having further (domain-dependent) knowledge (e.g. in our scenario, information that `CommX` is only interested in triage-related goals) can prune some of these goals by removing the goal conditions that are not typed on the triage variable.

2.2.2 Goal / Plan Recognition

In the present scenario, we thus have a set Ψ_α of goals that α may be trying to achieve, and observations of the actions α is currently executing. At this point we refer to the work of Ramirez and Geffner who in [12] provided a technique to compile the problem of plan recognition into a classical planning problem. Given a sequence of observations θ, we recompute the probability distribution Θ over $G \in \Psi_\alpha$ by using a Bayesian update $P(G|\theta) \propto P(\theta|G)$, where the likelihood is approximated by the function $P(\theta|G) = 1/(1 + e^{-\beta\Delta(G,\theta)})$ where $\Delta(G, \theta) = C_p(G - \theta) - C_p(G + \theta)$. Here $\Delta(G, \theta)$ gives an estimate of the difference in cost C_p of achieving the goal G without and with the observations, thus increasing $P(\theta|G)$ for goals that explain the given observations. Thus, solving two compiled planning problems, with goals $G - \theta$ and $G + \theta$, gives us the required posterior update for the distribution Θ over possible goals of α. The details of the approach is available at [12].

The specific problem we will look at now is how to inform the robot's own planning process from the recognized goal set Ψ_α. In order to do this, we compute the optimal plans for each goal in the hypothesis goal set Ψ_α, and associate them with the probabilities of these goals from the distribution thus obtained. Information from these plans is then represented concisely in the form of resource profiles.

Notes on the Recognition Module

For our plan recognition module we use a much faster variation [7] of the above approach that exploits cost and interaction information from plan graphs to estimate the goal probabilities. This saves on the computational effort of having to solve two planning problems per goal. Also, note that while computing the plan to a particular goal G, we use a compiled problem instance with the goal $G + \theta$ to ensure that the predicted plan conforms to the existing observations. Details on the compilation is available at [12].

Also, the output of the planner does not need to be associated with probabilities - this is just the most general formulation. If we want to deal with just a set of plans that the robot needs to be aware of, we can treat the plan set either with a uniform distribution and/or by requiring exactly zero conflicts in the objective of the planner (this will become clearer in Section 2.4) depending on the preference.

Perhaps the biggest computational issue here is the need to compute optimal plans. While we still do it for our domain, as we will note later in Section 2.3, this might not be necessary, and suboptimal plans may be used in larger domains where computation is an issue.

2.3 Resources and Resource Profiles

As we discussed previously, since the plans of the agents are in parallel execution, the uncertainty introduced by the commander's actions cannot be mapped directly between the commander's final state and the robot's initial state. However, given the commander's possible plans, the robot can extract information about at what points of time the shared resources in the environment are likely to be locked by the commander. This information can be represented by *resource usage profiles* that capture the expected (over all the recognized plans) variation of probability of usage or availability over time. The robot can, in turn, use this information to make sure that the profile imposed by its own plan has minimal conflicts with those of the commander's.

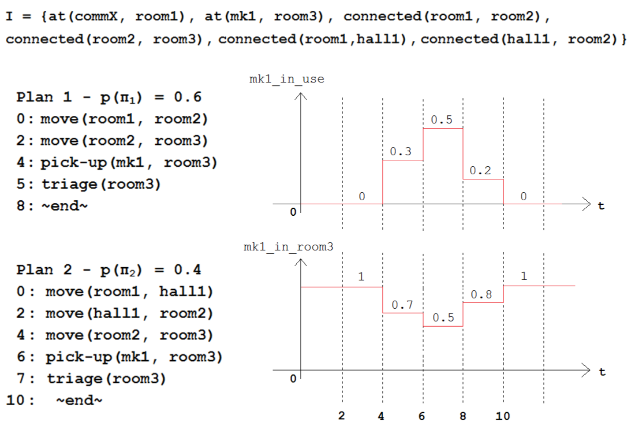


Figure 3: Different types of resource profiles.

Formally, a profile is defined as a mapping from time step T to a real number between 0 and 1, and is represented by a set of tuples as follows $\mathcal{G} : \mathbb{N} \rightarrow [0,1] \equiv \{(t,g) : t \in \mathbb{N}, g \in [0,1],$ such that $G(t) = g$ at time step $t\}$.

The concept of resource profiles can be handled at two levels of abstraction. Going back to our running example, shared resources that can come under conflict are the two (locatable typed objects) medkits, and the profiles over the medkits can be over both usage and location, as shown in Figure 3. These different types of profiles can be used (possibly in conjunction if needed) for different purposes. For example, just the usage profile shown on top is more helpful in identifying when to use the specific resource, while the resource when bound with the location specific groundings, as shown at the bottom can lead to more complicated higher order reasoning (e.g. the robot can decide to wait for the commander's plans to be over, as he inadvertently brings the medkit closer to it with high probability as a result of his own plans). We will look at this again in Section 3.

Let the domain model of the robot be $D_R = \langle T_R, V_R, S_R, A_R \rangle$ with the action models $a = \langle \mathbb{N}, \mathbb{C}, \mathbb{P}, \mathbb{E} \rangle$ defined in the same way as described in Section 2.1. Also, let $\Lambda \subseteq V_R$ be the set of shared resources and for each $\lambda \in \Lambda$ we have a set of predicates $f^\lambda \subseteq S_R$ that are *influenced* (as determined by the system designer) by λ, and let $\Gamma : \Lambda \rightarrow \mathcal{P}(\xi)$ be a function that maps the resource variables to the set of predicates $\xi = \cup_\lambda f^\lambda$ they influence. Without any external knowledge of the environment, we can set $\Lambda = V_\alpha \cap V_R$ and $\xi = S_\alpha \cap S_R$, though in most cases these sets are much smaller. In the following discussion, we will look at how the knowledge from the hypothesis goal set can be modeled in terms of resource availability graphs for each of the constrained resources $\lambda \in \Lambda$.

Consider the set of plans Ψ_α^P containing optimal plans corresponding to each goal in the hypothesis goal set, i.e. $\Psi_\alpha^P = \{\pi_G^* = \langle a_1, a_2, \ldots a_t \rangle \mid \delta(\pi_G^*, \mathbb{I}_\alpha) \models G,\ a_i \in A_\alpha \forall i,\ G \in \Psi_\alpha\}$ and let $l(\pi)$ be the likelihood of the plan π modeled on the goal likelihood distribution $\forall\ G \in \Psi_\alpha,\ p(G) \sim \Theta$ as $l(\pi_G) = c|\pi_G| \times p(G)$, where c is a normalization constant.

At each time step t, a plan $\pi \in \Psi_\alpha^P$ may lock one or more of the resources λ. Each plan thus provides a profile of usage of a resource with respect to the time step t as $\mathcal{G}_\pi^\lambda : \mathbb{N} \rightarrow \{0,1\} = \{(t,g) \mid t \in [1, |\pi|]$ and $g = 1$ if λ is locked by π at step t, 0 otherwise$\}$ such that $\mathcal{G}_\pi^\lambda(t) = g\ \forall\ (t,g) \in \mathcal{G}_\pi^\lambda$. The resultant usage profile of a resource λ due to all the plans in Ψ_α^P is obtained by summing over (weighted by the individual likelihoods) all the individual profiles as $\mathcal{G}^\lambda : \mathbb{N} \rightarrow [0,1] = \{(t,g) \mid t \in \{1, max_{\pi \in \Psi_\alpha^P} |\pi|\}$ and $g \propto \frac{1}{|\Psi_\alpha^P|} \sum_{\pi \in \Psi_\alpha^P} \mathcal{G}_\pi^\lambda(t) \times l(\pi)\}$.

Similarly, we can define profiles over the actual groundings of a variable (shown in the lower part of Figure 3) as $\mathcal{G}_\pi^{f^\lambda} = \{(t,g) \mid t \in [1, |\pi|]$ and $f^\lambda = 1$ at step t of plan π, 0 otherwise$\}$, and the resultant usage profile due to all the plans in Ψ_α^P is obtained as before as $\mathcal{G}^{f^\lambda} = \{(t,g) \mid t = 1, 2, \ldots, \max_{\pi \in \Psi_\alpha^P} |\pi|$ and $g \propto \frac{1}{|\Psi_\alpha^P|} \sum_{\pi \in \Psi_\alpha^P} \mathcal{G}_\pi^{f^\lambda}(t) \times l(\pi)\}$. These profiles are helpful when actions in the robot's domain are conditioned on these variables, and the values of these variables are conditioned on the plans of the other agents in the environment currently under execution.

One important aspect of this formulation that should be noted here is that the notion of "resources" is described here in terms of the subset of the common predicates in the domain of the agents ($\xi \subseteq S_\alpha \cap S_R$) and can thus be used as a generalized definition to model different types of conflict between the plans between two agents. In as much as these predicates are descriptions (possibly instantiated) of the typed variables in the domain and actually refer to the physical resources in the environment that might be shared by the agents, we will stick to this nomenclature of calling them *"resources"*. We will now look at how an autonomous agent can use these resource profiles to minimize conflicts during plan execution with other agents in its environment.

Notes on Usefulness of Profile Computation

One interesting aspect of computing resource profiles is that it provides a powerful interface between the belief on the environment and the planner. On the one hand, note that the input from the previous stage (goal/plan recognition module) is as generic as possible - a set of plans possibly associated with probabilities. Given any changes in preceding stages, e.g. modeling stochasticity or more complex belief models, still yields a set of plans that the robot needs to be aware of. Thus the plan set and resource profiles provide a surprisingly simple yet powerful way of abstracting away relevant information for the planner to use.

The profiles may also be leveraged to address different modalities of conformant behavior, for example with multiple humans and their relative importance, by (1) weighing the contributions from individual profiles by the normalized priority of the human, which would cause the planner to avoid conflicts with these profiles more than with those with lower priorities; or (2) requiring zero conflicts on a subset of profiles which would cause the planner to avoid a subset of conflicts at all costs, while minimizing the rest.

A somewhat implicit advantage of using profiles is its ability to form *regions of interest* given the possible plans. This will become clear later in Section 4.2 when we show that the predicted conflicts provide well-informed guidance to avoiding real conflicts during execution (as evident by the robustness in performance with just 1-3 observations, and zero actual conflicts in low probability areas in the computed profiles). Right now this has the implication that we need not necessarily compute perfect plan costs and goal distributions to get good plans.

2.4 Conflict Minimization

The planning problem of the robot - given by $\Pi = \langle D_R, \pi_R, \Lambda, \{G^\lambda | \forall \lambda \in \Lambda\}, \{G^{f^\lambda} \mid \forall f \in \Gamma(\lambda), \forall \lambda \in \Lambda\}\rangle$ - consists of the domain model D_R and the problem instance $\pi_R = \langle \mathbb{O}_R, \mathbb{I}_R, \mathbb{G}_R\rangle$ similar to that described in section 2.3, and also the constrained resources and all the profiles corresponding to them. This is because the planning process must take into account both goals of achievement as also conflict of resource usages as described by the profiles. Traditional planners provide no direct way to handle such profiles within the planning process. Note here that since the execution of the plans of the agents is occurring in parallel, the uncertainty is evolving at the time of execution, and hence the uncertainty cannot be captured from the goal states of the recognized plans alone, and consequently cannot be simply compiled away to the initial state uncertainty for the robot and solved as a conformant plan. Similarly, the problem does not directly compile into action costs in a metric planning instance because the profiles themselves are varying with time. Thus we need a planner that can handle these resource constraints that are both stochastic and non-stationary due to the uncertainty in the environment. To this end we introduce the following IP-based planner (partly following the technique for IP encoding for state space planning outlined in [16]) as an elegant way to sum over and minimize overlaps in profiles during the plan generation process. The following formulation finds such T-step plans in case of non-durative or instantaneous actions.

For action $a \in A_R$ at step t we have an action variable:

$$x_{a,t} = \begin{cases} 1, & \text{if action a is executed in step t} \\ 0, & \text{otherwise;} \quad \forall a \in A_R,\ t \in \{1, 2, \ldots, T\} \end{cases}$$

Also, for every proposition f at step t a binary state variable is introduced as follows:

$$y_{f,t} = \begin{cases} 1, & \text{if proposition is true in plan step t} \\ 0, & \text{otherwise;} \quad \forall f \in S_R,\ t \in \{0, 1, \ldots, T\} \end{cases}$$

Note here that the plan computed by the robot introduces a resource consumption profile itself, and thus one optimizing criterion would be to minimize the overlap between the usage profile due to the computed plan with those established by the predicted plans of the other agents in the environment. Let us introduce a new variable to model the resource usage graph imposed by the robot as follows:

$$g_{f,t} = \begin{cases} 1, & \text{if } f \in \xi \text{ is locked at plan step } t \\ 0, & \text{otherwise;} \quad \forall f \in \xi,\ t \in \{0, 1, \ldots, T\} \end{cases}$$

Further, for every resource $\lambda \in \Lambda$, we divide the actions in the domain of the robot into three disjoint sets -

$\Omega_f^+ = \{a \in A_R \text{ such that } x_{a,t} = 1 \implies y_{f,t} = 1\}$,
$\Omega_f^- = \{a \in A_R \text{ such that } x_{a,t} = 1 \implies y_{f,t} = 0\}$, and
$\Omega_f^o = A_R \setminus (\Omega_f^+ \cup \Omega_f^-)$, $\forall f \in \xi$.

These then specify respectively those actions in the domain that lock, free up, or do not affect the use of a particular resource, and are used to calculate $g_{f,t}$ in the IP. Further, we introduce a variable $h_{f,t}$ to track preconditions required by actions *in the generated plan* whose success is conditioned on the influence of the plans of the other agents on the world (e.g. position of the medkits are changing, and the action `pickup` is conditioned on it) as follows:

$$h_{f,t} = \begin{cases} 1, & \text{if } f \in \mathbb{P}_a \text{ and } x_{a,t+1} = 1 \\ 0, & \text{otherwise;} \quad \forall f \in \xi,\ t \in \{0, 1, \ldots, T-1\} \end{cases}$$

Then the solution to the IP should ensure that the robot only uses these resources when they are in fact most expected to be available (as obtained by maximizing the overlap between $h_{f,t}$ and G^{f^λ}). These act like *demand profiles* from the perspective of the robot. We also add a "no-operation" action $A_R \leftarrow A_R \cup a_\phi$ so that $a_\phi = \langle \mathbb{N}, \mathbb{C}, \mathbb{P}, \mathbb{E}\rangle$ where $\mathbb{N} = \texttt{NOOP}$, $\mathbb{C} = 0$, $\mathbb{P} = \{\}$ and $\mathbb{E} = \{\}$.

The IP formulation is given by:

$$\begin{aligned} &min\ k_1 \textstyle\sum_{a \in A_R} \sum_{t \in \{1,2,\ldots,T\}} \mathbb{C}_a \times x_{a,t} \\ &+k_2 \textstyle\sum_{\lambda \in \Lambda} \sum_{f \in \Gamma(\lambda)} \sum_{t \in \{1,2,\ldots,T\}} g_{f,t} \times G^\lambda(t) \\ &-k_3 \textstyle\sum_{\lambda \in \Lambda} \sum_{f \in \Gamma(\lambda)} \sum_{t \in \{0,1,\ldots,T-1\}} h_{f,t} \times G^{f^\lambda}(t) \end{aligned}$$

$$y_{f,0} = 1\ \forall f \in \mathbb{I}_R \setminus \xi \tag{1}$$

$$y_{f,0} = 0\ \forall f \notin \mathbb{I}_R \text{ or } f \in \xi \tag{2}$$

$$y_{f,T} = 1\ \forall f \in \mathbb{G}_R \tag{3}$$

$$x_{a,t} \leq y_{f,t-1}\ \forall a\ s.t.\ f \in \mathbb{P}_a, \forall f \notin \xi, t \in \{1, \ldots, T\} \tag{4}$$

$$h_{f,t-1} = x_{a,t}\ \forall a\ s.t.\ f \in \mathbb{P}_a, \forall f \in \xi, t \in \{1, \ldots, T\} \tag{5}$$

$$y_{f,t} \leq y_{f,t-1} + \textstyle\sum_{a \in add(f)} x_{a,t} \quad \text{s.t. } add(f) = \{a | f \in eff^+(a)\}, \forall f, t \in \{1, \ldots, T\} \tag{6}$$

$$y_{f,t} \leq 1 - \textstyle\sum_{a \in del(f)} x_{a,t} \quad \text{s.t. } del(f) = \{a | f \in eff^-(a)\}, \forall f, t \in \{1, \ldots, T\} \tag{7}$$

$$\textstyle\sum_{a \in A_R} x_{a,t} = 1, t \in \{1, 2, \ldots, T\} \tag{8}$$

$$\textstyle\sum_{a \in \Omega_f^+} \sum_t x_{a,t} \leq 1\ \forall f \in \xi, t \in \{1, 2, \ldots, T\} \tag{9}$$

$$g_{f,t} = \textstyle\sum_{a \in \Omega_f^+} x_{a,t} + (1 - \sum_{a \in \Omega_f^+} x_{a,t} - \sum_{a \in \Omega_f^-} x_{a,t}) \times g_{f,t-1} \quad \forall f \in \xi, t \in \{1, \ldots, T\} \tag{10}$$

$$h_{f,t} \times G^{f^\lambda}(t) \geq \epsilon\ \forall f \in \xi, t \in \{0, 1, \ldots, T-1\} \tag{11}$$

$$y_{f,t} \in \{0, 1\}\ \forall f \in S_R, t \in \{0, 1, \ldots, T\} \tag{12}$$

$$x_{a,t} \in \{0, 1\}\ \forall a \in A_R, t \in \{1, 2, \ldots, T\} \tag{13}$$

$$g_{f,t} \in \{0, 1\}\ \forall f \in S_R, t \in \{0, 1, \ldots, T\} \tag{14}$$

$$h_{f,t} \in \{0, 1\}\ \forall f \in S_R, t \in \{0, 1, \ldots, T-1\} \tag{15}$$

where k_1, k_2, k_3 are constants determining the relative importance of the optimization criteria and ϵ is a constant.

Here, the objective function minimizes the sum of the cost of the plan and the overlap between the cumulative resource usage profiles of the predicted plans and that imposed by the current plan of the robot itself while maximizing the validity of the demand profiles. Constraints (1) through (3) model the initial and goal conditions, while the value of the constrained variables are kept uninitialized (and are determined by their profiles). Constraints (4) and (5), depending on the particular predicate, enforces the preconditions, or produces the demand profiles respectively, while (6) and (7) enforces the state equations that maintain the add and delete effects of the actions. Constraint (8) imposes non concurrency on the actions, and (9) ensures that the robot does not repeat the same action indefinitely to increase its utility. Constraint (10) generates the resource profile of the current plan, while (11) maintains that actions are only executed if there is at least a small probability ϵ of success. Finally (12) to (15) provide the binary ranges of the variables.

Note on Temporal Expressivity

At this point it is worth acknowledging the implications of having durative actions in our formulation. Note that our approach does *not* discretize time, but rather uses time points as steps in the plan - that can be easily augmented with their own durations. So in order to handle durative actions, the only (somewhat minor) change required in the formulation is in the way the conflicts are integrated (instead of summed) over in the objective function. Further, uncertainty in action durations is always a big issue in human interactions; though resource profiles cannot directly handle uncertain durations, it only affects the way the profiles are calculated, and the way in which information is expressed in it remains unchanged (i.e. expectations over action durations add an extra expectation to the already probabilistic profile computation). As noted before in Section 2.3, the ability of profiles to form regions of interest is crucial in handling such scenarios implicitly.

3. MODULATING BEHAVIOR OF THE ROBOT

The planner is implemented on the IP-solver `gurobi` and integrates [7] and [8] respectively for goal recognition and plan prediction for the recognized goals. We will now illustrate how the formulation can produce different behaviors of the robot by appropriately configuring the parameters of the planner. For this discussion we will limit ourselves to a singleton hypothesis goal set in order to observe the robot's response more clearly.

3.1 Compromise

Let us now look back at the environment we introduced in Figure 1. Consider that the goal of the commander is to perform triage in `room1`. The robot computes the human's optimal plan (which ends up using `medkit1` at time steps 7 through 12) and updates the resource profiles accordingly. If it has its own goal to perform triage in `hall3`, the plan that it comes up with given a 12 step lookahead is shown below. Notice that the robot opts for the other medkit (`medkit2` in `room3`) even though its plan now incurs a higher cost in terms of execution. The robot thus can adopt a policy of *compromise* if it is possible for it to preserve the commander's (expected) plan.

```
01 MOVE_ROBOT_ROOM1_HALL1
02 MOVE_ROBOT_HALL1_HALL2
03 MOVE_ROBOT_HALL2_HALL3
04 MOVE_ROBOT_HALL3_HALL4
05 MOVE_REVERSE_ROBOT_HALL4_ROOM4
06 MOVE_REVERSE_ROBOT_ROOM4_ROOM3
07 PICK_UP_MEDKIT_ROBOT_MK2_ROOM3
08 MOVE_ROBOT_ROOM3_ROOM4
09 MOVE_ROBOT_ROOM4_HALL4
10 MOVE_REVERSE_ROBOT_HALL4_HALL3
11 CONDUCT_TRIAGE_ROBOT_HALL3
12 DROP_OFF_ROBOT_MK2_HALL3
```

3.2 Opportunism

Notice, however, that the commander is actually bringing the medkit to `room1` as predicted by the robot, and this is a favorable change in the world, because robot can use this `medkit` once the commander is done and achieve its goal at a much lower cost. The robot, indeed, realizes this once we give it a bigger time horizon to plan with, as shown above (on the right). Thus, in this case, the robot shows *opportunism* based on how it believes the world state will change.

```
01 NOOP
02 NOOP
   ...
13 NOOP
14 PICK_UP_MEDKIT_ROBOT_MK1_ROOM1
15 MOVE_ROBOT_ROOM1_HALL1
16 MOVE_ROBOT_HALL1_HALL2
17 MOVE_ROBOT_HALL2_HALL3
18 CONDUCT_TRIAGE_ROBOT_HALL3
19 DROP_OFF_ROBOT_MK1_HALL3
```

3.3 Negotiation

In many cases, the robot will have to eventually produce plans that will have potential points of conflict with the expected plans of the commander. This occurs when there is no feasible plan with zero overlap between profiles (specifically $\sum g_{f,t} \times G^{\lambda}(t) = 0$) or if the alternative plans for the robot are too costly (as determined by the objective function). If, however, the robot is equipped with the ability to communicate with the human, then it can *negotiate* a plan that suits both. To this end, we introduce a new variable $H^{\lambda}(t)$ and update the IP as follows:

$$
\begin{aligned}
&min\ k_1 \textstyle\sum_{a \in A_R} \ \sum_{t \in \{1,2,\ldots,T\}} \mathbb{C}_a \times x_{a,t} \\
&+k_2 \textstyle\sum_{\lambda \in \lambda} \ \sum_{f \in \Gamma^{-1}(\lambda)} \ \sum_{t \in \{1,2,\ldots,T\}} g_{f,t} \times H^{\lambda}(t) \\
&-k_3 \textstyle\sum_{\lambda \in \Lambda} \ \sum_{f \in \Gamma^{-1}(\lambda)} \ \sum_{t \in \{0,1,\ldots,T-1\}} h_{f,t} \times G^{f^{\lambda}}(t) \\
&+k_4 \textstyle\sum_{\lambda \in \Lambda} \ \sum_{t \in \{0,1,\ldots,T\}} ||G^{\lambda}(t) - H^{\lambda}(t)||
\end{aligned}
$$

$$y_{f,T} \geq h_{f,t-1} \ \forall \ a \ s.t. \ f \in \mathbb{P}_a, \forall f \in \xi, t \in \{1,\ldots,T\} \tag{5a}$$

$$H^{\lambda}(t) \in [0,1] \ \forall \lambda \in \Lambda, t \in \{0,1,\ldots,T\} \tag{16}$$

$$H^{\lambda}(t) \leq G^{\lambda}(t) \ \forall \lambda \in \Lambda, t \in \{0,1,\ldots,T\} \tag{17}$$

Constraint (5a) now complements constraint (5) from the existing formulation, by promising to restore the world state every time a demand is made on a variable. The variable $H^{\lambda}(t)$, maintained by constraints (16) and (17), determine the desired deviation from the given profiles. The objective function has been updated to reflect that overlaps are now measured with the desired profile of usage, and there is a cost associated with the deviation from the real one. The revised plan now produced by the robot is shown below.

```
01 MOVE_ROBOT_ROOM1_HALL1
02 MOVE_ROBOT_HALL1_HALL2
03 MOVE_REVERSE_ROBOT_HALL2_ROOM2
04 PICK_UP_MEDKIT_ROBOT_MK1_ROOM2
05 MOVE_ROBOT_ROOM2_HALL2
06 MOVE_ROBOT_HALL2_HALL3
07 CONDUCT_TRIAGE_ROBOT_HALL3
08 MOVE_REVERSE_ROBOT_HALL3_HALL2
09 MOVE_REVERSE_ROBOT_HALL2_ROOM2
10 DROP_OFF_ROBOT_MK1_ROOM2
```

Notice that the robot restores the world state that the human is believed to expect, and can now communicate to him *"Can you please not use `medkit1` from time 7 to 9?"* based on how the real and the ideal profiles diverge, i.e. t such that $H^{\lambda}(t) < G^{\lambda}(t)$ for each resource λ.

Notes on Adaptive Behavior Modeling

One might note here that people are often adaptive and it is very much possible that they may be willing to change their goals based on observing the robot or are even unwilling to negotiate if their plans conflict. Hence the policies of compromise and opportunism for the robot are complementary to negotiation in the event the latter fails. Thus, for example, the robot might choose to communicate a negotiation strategy to the human, but fall back on a compromise if that fails. It is a merit of such a simple formulation to be able to handle such interesting adaptive behaviors.

4. EVALUATION

The power of the proposed approach lies in the modular nature in which it tackles several complicated problems that are separate research areas in their own rights. As we saw throughout the course of the discussion, approaches used in the individual modules may be varied with little to no change in the rest of the architecture. For example the expressivity of the belief modeling or goal recognition component is handled separately as the planner used information from a generic plan set. Again the representation technique introduced in terms of resource profiles provide properties in terms of computational independence with respect to size of the hypothesis set and number of agents (which gets manifested in complexity in number of resources) that general planners do not have. So it becomes a design choice depending on which metric needs to be optimized.

For empirical evaluations, we simulated the USAR scenario on 360 different problem instances, randomly generated by varying the specific (as well as the number of probable) goals of the human, and evaluated how the planner behaved with the number of observations it can start with to build its profiles. We fix the domain description, location and goal of the agents, and the position of the resources, and consider randomly generated hypothesis goal sets of size 2-11. The goals of the commander were assumed to be known to be triage related, but the location of the triage was allocated randomly (one of which was again picked at random as the real goal). Finally for each of these problems, we generate 1-5 observations by simulating the commander's plan over the real goal, and use these observations known *a priori* the robot's plan generation process. The experiments were conducted on an Intel Xeon(R) CPU E5-1620 v2 3.70GHz×8 processor with a 62.9GiB memory.

4.1 Scaling Up

Our primary contribution is the formulation for planning with resource profiles, while the goal recognition component can be any off-the-shelf algorithm, and as such we compare scalability with respect to the planning component only.

- *w.r.t. Length of the Planning Horizon*

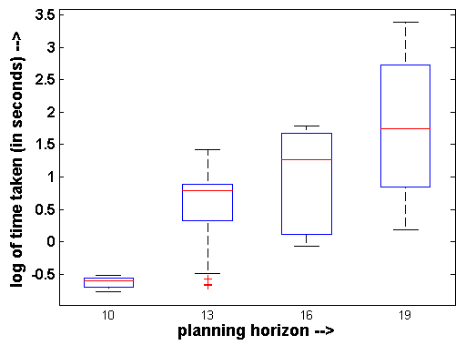
The performance of the planner with respect to the planning horizon is shown in Figure 4a. This is, as expected, the bottleneck in computation due to exponential growth of the size of the IP. It is however not prohibitively expensive, and the planner is still able to produce plans of length 20 (steps, not durations) for our domain in a matter of seconds.

- *w.r.t. Number of Resources*

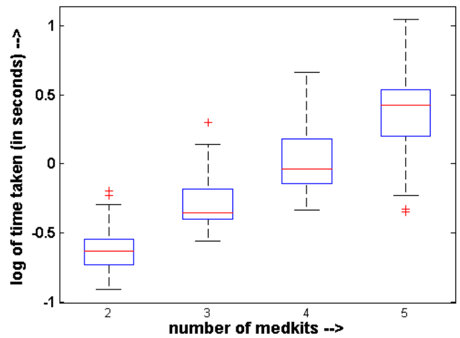
The performance of the planner with respect to the number of constrained resources (medkits, in the context of the current discussion) is shown in Figure 4b. Evidently, the computational effort is dominated by that due to the planning horizon. This reiterates the usefulness of abstracting the information in predicted plans in the form of resource profiles, thus isolating the complexities of the domain with that of the underlying planning algorithm.

- *w.r.t. the Number of Agents and Goals*

The planning module (i.e. the IP formulation) is by itself independent of the number of agents being modeled. In fact, this is one of the major advantages of using abstractions like resource profiles in lieu of actual plans of each of the agents. On the other hand, the time spent on recognition, and on calculating the profiles, is significantly affected. However, observations on multiple agents are asynchronous, and goal recognition can operate in parallel, so that this is not a huge concern beyond the complexity of a single instance. Similarly the performance is also unaffected by the size of the hypothesis set Ψ_α, as shown in Figure 5, which shows increase in the number of the possible goals does not complicate the profiles to an extent to affect the complexity.



(a) w.r.t. T ($|\Lambda| = 2$)



(b) w.r.t. #medkits ($T = 10$)

Figure 4: Performance of the planner w.r.t. planning horizon T and number of constrained resources (medkits).

4.2 Quality of the Plans Produced

We define U as the average conflicts per plan step when a demand is placed on a resource by the robot, and S as the success probability per plan step that the demand is met. C is the cost of a plan. F is the percentage of times there was an actual conflict during execution (distinct from U which estimates the possible conflict that may occur per plan step). We observe the quality of the plans produced by the planner by varying the ratio of parameters k_1 and k_3 from the objective function and the length of the planning horizon T. Similar results can be produced by varying k_1/k_2.

From Table 1, as k_1/k_3 decreases, the planner becomes more conservative (to maximize success probability) and thus plans become costlier. At the same time the *expected* success rate of actions are also increased (with simultaneous increase in usage conflict), as reflected by a higher failure rate due to actual execution time conflicts.

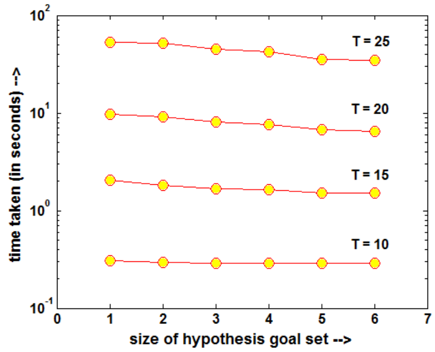


Figure 5: Performance of the planner w.r.t. size of the goal set. As expected, computational complexity is not affected.

k_1/k_3	0.05	0.5	5.0
C	9.47	6.37	6.31
U	0.18	0.17	0.17
S	0.85	0.579	0.578
F	27.5	23.0	21.3

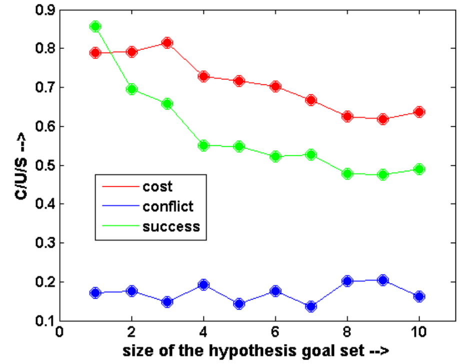
Table 1: Quality of plans produced w.r.t. k_1/k_3. Conservative plans result in lowered utility.

Also note, from Table 2 the impact of the planning horizon T on the types of behaviors we discussed in the previous section. As we increase T, the plan cost falls below the optimal, indicating opportunities for opportunistic behavior on the part of the robot. The expected conflict also falls to almost 0. However the expected success rate of actions also decreases, the ratio k_1/k_2 determines how daring the robot is, in choosing between cheap versus possibly invalid plans. Note, however, the actual execution time conflict is extremely low with increasing T, for even sufficiently conservative estimates of S.

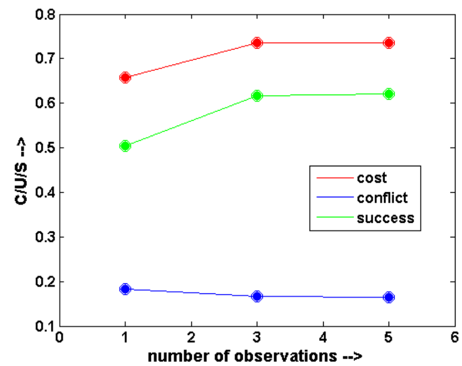
Thus we see that the robot is successfully able to navigate conflicts and find in many cases plans even cheaper than the original optimal plan, thus highlighting the usefulness of the approach. Finally, we look at the impact of the parameters in the plan recognition module in Figure 6. As expected, with bigger hypothesis sets, the success rate goes down. Interestingly, the plan cost also shows a downward trend which might be because the bigger variety in possible goals give a better idea of which medkits are generally more useful for that instance at what points of time. With more observations, as expected, the success rate goes up and the expected conflict goes down. The cost, however, increases a little as the planner opts for more conservative options.

5. CONCLUSIONS

In this paper we investigate how plans may be affected by conflicts on shared resources in an environment cohabited by humans and robots, and introduce the concept of resource profiles as a means of representation for concisely modeling the information pertaining to the usage of such resources, contained in predicted behavior of the agents. We propose a general formulation of a planner for such scenarios and show how the planner can be used to model different types of behavior of the robot by appropriately configuring the objective function and optimization parameters. Finally, we provide an end-to-end framework that integrates belief modeling, goal recognition and an IP-solver that can enforce the desired interaction constraints. One interesting research direction would be to consider nested beliefs on the agents; after all, humans are rarely completely aloof of other agents in its environment. Such interactions should have to consider evolution of beliefs with continued interactions and motivate further exploration of the belief modeling component. The modularity of the proposed approach allows for focused research on each (individually challenging) subtask without significantly affecting the others.



(a) w.r.t. $|\Psi_\alpha|$



(b) w.r.t. #obs

Figure 6: Performance of the planner w.r.t. size of goal set and number of observations ($k_1/k_3 = 0.5, T = 16$).

T	10	13	16	Optimal
C	9.0	5.6	4.53	9.0
U	0.46	0.04	≈0	n/a
S	1.0	0.48	0.25	n/a
F	53.3	11.9	6.6	53.3

Table 2: Quality of plans produced w.r.t. T. Opportunities for opportunism explored, conflicts minimized.

Acknowledgment

This research is supported in part by the ONR grants N00014-13-1-0176, N00014-13-1-0519 and N00014-15-1- 2027, and the ARO grant W911NF-13- 1-0023.

REFERENCES

[1] E. Beaudry, F. Kabanza, and F. Michaud. Planning with concurrency under resources and time uncertainty. In *Proceedings of the 2010 Conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, pages 217–222, Amsterdam, The Netherlands, The Netherlands, 2010. IOS Press.

[2] A. Blum and M. L. Furst. Fast planning through planning graph analysis. In *IJCAI*, pages 1636–1642, 1995.

[3] F. Cavallo, R. Limosani, A. Manzi, M. Bonaccorsi, R. Esposito, M. Di Rocco, F. Pecora, G. Teti, A. Saffiotti, and P. Dario. Development of a socially believable multi-robot solution from town to home. *Cognitive Computation*, 6(4):954–967, 2014.

[4] T. Chakraborti, G. Briggs, K. Talamadupula, Y. Zhang, M. Scheutz, D. Smith, and S. Kambhampati. Planning for serendipity. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[5] M. Cirillo, L. Karlsson, and A. Saffiotti. Human-aware task planning for mobile robots. In *Proc of the Int Conf on Advanced Robotics (ICAR)*, 2009.

[6] M. Cirillo, L. Karlsson, and A. Saffiotti. Human-aware task planning: An application to mobile robots. *ACM Trans. Intell. Syst. Technol.*, 1(2):15:1–15:26, Dec. 2010.

[7] Y. E.-Martín, M. D. R.-Moreno, and D. E. Smith. A fast goal recognition technique based on interaction estimates. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 761–768, 2015.

[8] M. Helmert. The fast downward planning system. *CoRR*, abs/1109.6051, 2011.

[9] U. Koeckemann, F. Pecora, and L. Karlsson. Grandpa hates robots - interaction constraints for planning in inhabited environments. In *Proc. AAAI-2010*, 2014.

[10] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard. Feature-based prediction of trajectories for socially compliant navigation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

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

[12] M. Ramirez and H. Geffner. Probabilistic plan recognition using off-the-shelf classical planners. In *In Proc. AAAI-2010*, 2010.

[13] E. Sisbot, L. Marin-Urias, R. Alami, and T. Simeon. A human aware mobile robot motion planner. *Robotics, IEEE Transactions on*, 23(5):874–883, Oct 2007.

[14] K. Talamadupula, G. Briggs, T. Chakraborti, M. Scheutz, and S. Kambhampati. Coordination in human-robot teams using mental modeling and plan recognition. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2957–2962, Sept 2014.

[15] S. Tomic, F. Pecora, and A. Saffiotti. Too cool for school - adding social constraints in human aware planning. In *Proc of the International Workshop on Cognitive Robotics (CogRob)*, 2014.

[16] T. Vossen, M. O. Ball, A. Lotem, and D. S. Nau. On the use of integer programming models in ai planning. In T. Dean, editor, *IJCAI*, pages 304–309. Morgan Kaufmann, 1999.