Human-Aware Planning for Robots Embedded in Ambient Ecologies

Marcello Cirillo[a], Lars Karlsson[a], Alessandro Saffiotti[a]

[a]*AASS Research Center*
Örebro University, Sweden

Abstract

We address the issue of human-robot cohabitation in smart environments. In particular, the presence of humans in a robot's work space has a profound influence on how the latter should plan its actions. We propose the use of *Human-Aware Planning*, an approach in which the robot exploits the capabilities of a sensor-rich environment to obtain information about the (current and future) activities of the people in the environment, and plans its tasks accordingly.

Here, we formally describe the planning problem behind our approach, we analyze its complexity and we detail the algorithm of our planner. We then show two application scenarios that could benefit from the techniques described. The first scenario illustrates the applicability of human-aware planning in a domestic setting, while the second one illustrates its use for a robotic helper in a hospital. Finally, we present a five hour-long test run in a smart home equipped with real sensors, where a cleaning robot has been deployed and where a human subject is acting. This test run in a real setting is meant to demonstrate the feasibility of our approach to human-robot interaction.

Keywords: Human-aware planning, Planning under uncertainty, Human-robot interaction, Planning in intelligent environments

1. Introduction

A very promising convergence is taking place between the fields of ambient intelligence and autonomous robotics. Researchers in both areas are increasingly interested in the possibilities offered by the inclusion of robotic devices inside smart environments. On the one hand, this inclusion could empower a smart environment with unprecedented sensing and actuation capabilities. On the other hand, it could make a robot more robust and autonomous by exploiting the capabilities offered by the other devices in the environment. For instance, a robotic vacuum cleaner could rely on the home localization system to know its position in the home, and hence to optimize its cleaning strategy.

The convergence of ambient intelligence and autonomous robotics has given birth to several new research areas, including network robot systems [1], ubiquitous robotics [2], and robot

Email addresses: marcello.cirillo@aass.oru.se (Marcello Cirillo), lars.karlsson@aass.oru.se (Lars Karlsson), alessandro.saffiotti@aass.oru.se (Alessandro Saffiotti)

ecologies [3, 4, 5]. In all these areas, the main goal is to design intelligent robotic environments, that is, environments where close communication is established among sensing and robotic devices. The coordinated cooperation of such devices can enact and support complex tasks to help the users in their everyday life. The whole environment is often viewed as an *ecosystem* of components, and human users are considered as the core of this ecosystem. Intelligent robotic environments are expected to open a wide space of new applications, ranging from domestic robots aimed at improving the quality of life of the elderly [6], to service robots operating in a factory outside the confined space of a working cell. Most of these new applications have one aspect in common: robots would operate in the same space as humans, and interact with them as members of the same ecosystem.

The cohabitation of humans and robots in the same space induces many difficult technical challenges. Safety and human-robot interaction immediately come to mind, and intensive work is on-going on both problems since a few years, e.g., [7, 8]. There are, however, other challenges which have attracted less attention, but which are nonetheless very important. In particular, the presence of humans in the robot's work space has a profound influence on how the robot should plan its actions. Action planning has been recognized as been an important capability for an intelligent environment [9], and it becomes essential when a robot is embedded into it. Unfortunately, classical AI planning techniques, in which the robot is in full control and the state of the world is only affected by the actions of the robot [10], are not applicable here. To be effective in an environment populated by humans, the robot should take into account the actions performed by the humans both when planning its tasks, and during execution [11].

In this paper, we propose to use *Human-Aware Planning* [12] to enable the co-habitation of robots and humans in the same smart environment. In Human-Aware Planning, the robot uses a forecast of the plan, or set of possible plans, that the human is expected to perform, and generates its own plan in such a way to achieve its goals while respecting a given set of *interaction constraints* with the human. Interaction constraints can encode safety conditions, comfort conditions, or task related conditions. For instance, these constraints may impose that the robot should never operate in the same room where the human is expected to be. Human-Aware Planning relies on the availability of a forecast of the human plans. When the robot and the human are embedded in a *smart environment*, this forecast can be based on the data collected by the sensors in the environment, e.g., using techniques for activity recognition [13, 14, 15].

The example in figure 1 illustrates Human-Aware Planning. We consider a smart, sensor-rich environment that includes a robotic vacuum cleaner. The robot has the goal to clean the apartment in which a human lives. The environment provides the robot with a prediction of the actions that the human is going to do in the morning (at the bottom of the figure) and the robot plans its cleaning operations (the policy at the top of the figure) so as not to disturb the human. This is indeed a simple example, where we only have one possible prediction for the human actions, and where the planner generates a simple (linear) robot policy. However, this example shows a core point of our approach: human actions are predicted, not planned, and they are used as input for the task planner of the robot.

We have presented a first step toward Human-Aware Planning in a previous paper [12], in the form of a specific technique. This technique relies on a few simplifying assumptions, one of which being that there is a single human in the environment. In order to make this technique applicable to robots embedded in real (domestic or industrial) intelligent environments, however, we need to overcome this limitation, and account for the presence of multiple humans at the same time. In this paper, we extend our approach in order to consider multiple humans in the planning process. Multiple alternative plans for each human are also considered and the planning process

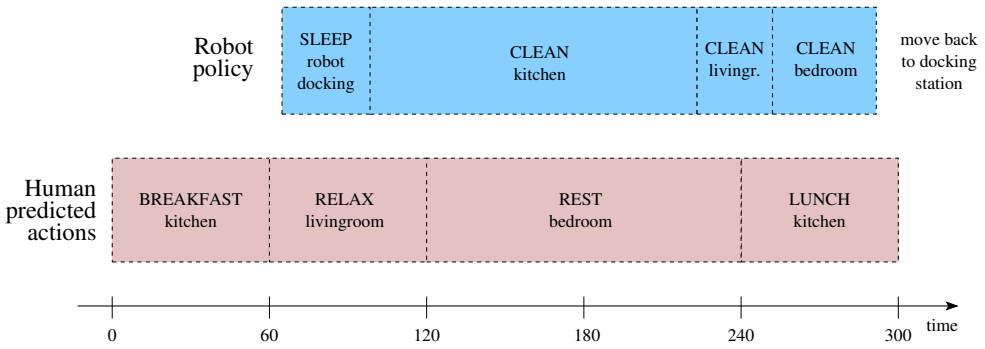

Figure 1: A simple example of human-aware planning. A robotic vacuum cleaner is deployed in a sensor-rich environment and it is equipped with a planner that takes as input the predicted actions that the human user will perform in the following hours (bottom part). The output of the planner is the policy of the robot (top). The moving actions of the robot from one room to the other have been omitted for space reasons. Since the robot is required to end its tasks at the docking station, a final **move** action has been specified at the end of the policy.

can deal with partial observability of the domain, including partial observability of the actions performed by the humans. Accordingly, our planner generates policies that are conditional on what the humans will be observed to do. Actions have duration in our model, although for the purpose of this paper we assume that durations are deterministic.

Predictions on users' future plans and observations of the actions performed by human agents are key points on which our approach relies. However, predictions cannot be generated by the planner, and observations on humans' actions cannot be delegated to the robot during execution, because they would be limited by its current location and sensing capabilities. Our approach heavily relies on the assumption that a sensor-rich, smart environment will provide those services. In our vision, the human-aware planner is a component that reasons at a high abstraction level as part of a larger infrastructure, where the different aspects of the classic sense-plan-act loop are delegated to different components. Information from the sensors distributed in the environment are gathered and processed to deduce contextual information about the state of the users, and to forecast their future activities [16, 15]. Such information is then passed to the planner, that generate action policies. At execution level, the robot performs the actions detailed in the policy, according to the contextual information provided by the environment. The policy is valid as long as each human user performs the sequence of actions described in one of the forecasts. The detection of a change in the course of actions is delegated again to other components in the environment, that would inform the planner, providing new possible predictions for future human actions. In our current approach, such changes are handled through replanning.

The rest of this paper is organized as follows: section 2 presents a survey of relevant work on human-robot interaction and cohabitation. In section 3, we formally describe the human-aware planning problem, while in section 4 we detail the algorithm we implemented and used to demonstrate the feasibility of our approach in two realistic scenarios and in a real test run in a smart environment (section 5). Section 6 concludes.

2. Related Work

The research in human-robot interaction and cohabitation is currently very active. In the robotics field, many attempts have tried to integrate autonomous entities in everyday environments, such as homes [17], train stations [18], shopping malls [19], and in city streets (e.g., the DustBot project [20], where a group of cooperating robots are used to improve the management of urban hygiene, or the URUS project [21], that aims at the development of new ways of cooperation between network robots and human beings in urban areas). Although most of these works do not exploit sensor rich environments where computational power is distributed in different interconnected components, some research groups have started to explore an ecological approach to service robots [22]. An example of this approach is the PEIS Ecology project [5], in which advanced domestic tasks are performed by the cooperation of many simple robot devices pervasively embedded in the environment.

In the field of robotics, the works that consider human-robot co-habitation often take a viewpoint which is different from the one that we adopt here, by focusing on aspects such as safety (e.g., in the MORPHA project [23]), acceptable motion (e.g., within COGNIRON [24] and in the frame of the URUS project [25]) or human-aware manipulation [26]. The problem of task planning in the presence of humans is currently still open, although some researchers have started exploring the issue [27, 28]. Montreuil et al. [29], Clodic et al. [8] and Galindo et al. [30] have addressed the complementary problem of how a robot could generate collaborative plans involving a human. In our work, by contrast, the robot does not plan actions for the human, but instead tries to forecast actions and plans of the human from previous actions. Our approach also diverges from the techniques developed for plan merging [31], because in our case the human is not controllable, and because some of the human's actions can prompt new goals for the robot. As a consequence, the actions of the human cannot be re-scheduled, and the plan of the robot cannot be created *a priori* and then rearranged. There has been extensive work on planning with external events (as human actions can be considered). An early example is the work of Blythe [32], which used Bayesian nets to compute the probability of success in the presence of external events. Our approach is also somewhat reminiscent of early work on collaborative planning [33].

At the same time, psychologists and designers are at work to create more socially acceptable robots and to understand the reactions generated by robots on human subjects in different environments and situations. Examples of such interdisciplinary studies can be found in the work of Hiolle et al. [34], where the researchers tried to assess how humans would react to the interaction with robots showing different attachment profiles, and in the study of Lee et al. [35], in which designers, computer scientists, behavioral science and robotics experts worked together to design a socially acceptable robot intended to provide snacks in office buildings. Similar studies have been performed that specifically address the needs of the elderly population [36].

An important property of our planning problem is that actions can be executed simultaneously and have durations, and that property has been addressed before in the literature. For instance, Mausam and Weld [37] present an MDP model (fully observable) based on the concept of interwoven epoch search space, adapted from Haslum and Geffner [38].

3. The Human-Aware Planning Problem

We identified a set of important functionalities that need to be incorporated in a planner to support human-aware planning in an intelligent environment:

1. Support for alternative hypotheses of the humans' plans, where a human plan — or human *agenda* — is a sequence of actions that the human might perform in the future;
2. Support for multiple humans who can be present in the environment at the same time;
3. Temporal duration of actions, both from the robot and from the human side;
4. Possibility to specify interaction constraints (*IC*), that is, rules that determine how the robot should react to specific human actions and to situations in the environment;
5. Support for partial goal achievement; and
6. Support for observations on effects of human actions.

Our human-aware planner incorporates all these functionalities. It takes as inputs sets of possible human agendas (one set for each human present in the environment), the interaction rules and a set of goals and generates a policy that is compliant with these inputs. The possible agendas, together with an associated probability distribution, are assumed to be estimated by a separate plan recognition module. As one doesn't know in advance which of these agendas is the actual one, the planning problem has the property of partial observability. The planner, which is an extension of PTLplan [39], generates policies which are conditional on observations relating to the human's actions.

3.1. States and situations

A *state* s is represented in terms of a set of state variables and their values. The set of all states is denoted S.

An *action* a has preconditions $Pr_a : S \rightarrow \{T, F\}$, time duration $t_a \in \mathbb{R}^+$, a cost function $Cost_a : S \rightarrow \mathbb{R}^+$ and a transition function $Res_a : S \times S \rightarrow [0, 1]$ such that $Res_a(s, s')$ is the probability of going to state s' when action a is performed in state s. We assume in this paper that an action always takes the same amount of time to perform, and cannot be interrupted once started. The end state, however, can both depend on the starting state and be stochastic. Note that actions include both those performed by the human and by the robot. In the human case, the specification of preconditions is not compulsory, since we do not want to predicate on what the human can do. However, they can be specified to introduce new goals for the robot. For instance, in a domain in which a robot is helping a human operator in a factory, the operator may require at some point a specific tool to perform an action. In such case, the human action would have as a precondition the presence of the tool in the right location and every plan of the robot that would not make sure that the tool is delivered on time would be discarded. Another way for human actions to introduce new goals for the robot is by their effects on the environment. For instance, in the cleaning robot example, a human action that entails a massive use of the kitchen would introduce the new goal for the robot to clean the room. We use HA and RA to denote the sets of human actions and robot actions, respectively.

An *agenda* is a finite list of consecutive human actions: $(a_1, a_2, ..., a_n)$, with $a_i \in HA$ for all i.

A *situation* is a tuple $\langle s, rt, hta \rangle$ where s is a state and $rt \in \mathbb{R}^+$ is the time when the robot latest action ended. hta is a non empty set of tuples $hta = \{\langle ht_1, ha_1 \rangle, \ldots \langle ht_m, ha_m \rangle\}$, where $ht_i \in \mathbb{R}^+$ is the time when the ith human latest action ended and ha_i is the rest of the agenda for the same human. The cardinality of hta is equal to the number n of humans present in the environment that the robot should consider for planning. The set of situations is denoted Σ. The ordering of the $\langle ht_i, ha_i \rangle$ tuples is fixed over Σ, that is, in every situation $\langle ht_i, ha_i \rangle$ will refer to the i-th human agent.

3.2. Actions

We can now define what happens when the robot performs an action $a \in RA$ in a situation $\langle s, rt, hta \rangle$ by extending the function Res_a to transitions between situations:

$$Res_a(\langle s, rt, hta \rangle, \langle s', rt', hta' \rangle)$$

The definition of Res_a is composed of two parts: the first is the recursive case when there is a human agenda ha_i whose next action a'_i finishes before or at the same time of the robot current action a and the second part is applied when the action a of the robot finishes before the first action of every human agenda $ha_i : \langle ht_i, ha_i \rangle \in hta$.

In the recursive case, a human action a'_i is completed, resulting in a new situation $\langle s'', rt, hta'' \rangle$ from which one continues:

$$\sum_{s''} Res_{a'_i}(s, s'') \cdot Res_a(\langle s'', rt, hta'' \rangle, \langle s', rt', hta' \rangle)$$

Therefore, we first have to check if there are tuples in *hta* that contain an agenda whose first action will end before the robot action, that is, $\exists \langle ht_i, ha_i \rangle \in hta : \ a'_i = first(ha_i) \wedge ht_i + t_{a'_i} \leq rt + t_a$. Of all the tuples found, we choose the one whose agenda starts with the action that ends before all the others. In case two agendas start with actions that end at the same time, we use the ordering of the tuples in *hta* to discriminate which one will be applied first:

$$\begin{aligned}
\exists \langle ht_i, ha_i \rangle \in hta : \quad & a'_i = first(ha_i) \ \wedge ht_i + t_{a'_i} \leq rt + t_a \wedge \\
& \forall k, (k < i), \langle ht_k, ha_k \rangle \in hta : \\
& \quad \forall a'_k (a'_k = first(ha_k) \Rightarrow ht_k + t_{a'_k} > ht_i + t_{a'_i}) \wedge \\
& \nexists j, (j > i), \langle ht_j, ha_j \rangle \in hta : \\
& \quad \forall a'_j (a'_j = first(ha_j) \Rightarrow ht_j + t_{a'_j} < ht_i + t_{a'_i})
\end{aligned}$$

Each time that the recursive case of Res_a is applied (that is, until there's no human action in all the agendas that ends before the robot action), a new intermediate situation $\langle s'', rt, hta'' \rangle$ is generated, where

$$\begin{aligned}
\exists ht''_i, \exists ha''_i : \ & ht''_i = ht_i + t_{a'_i} \wedge ha''_i = rest(ha_i) \wedge \\
& hta'' = (hta \backslash \{\langle ht_i, ha_i \rangle\}) \cup \{\langle ht''_i, ha''_i \rangle\}
\end{aligned}$$

The robot time *rt* remains unchanged in the transition, since no robot action has yet been applied, while the state transition is $Res_{a'_i}(s, s'')$.

In case the action of the robot finishes before the first action of every human agenda, the transition function leads to a new situation in which the agendas are unchanged, while robot time and state reflect the results of the robot action.

The extension of Res_a to transition between situations can therefore be expressed as in the following equation. Note that the first situation is when the robot action is started ($\langle s, rt, hta \rangle$),

and the second situation is when it ends ($\langle s', rt', hta' \rangle$).

$$
\begin{array}{l}
Res_a(\langle s, rt, hta \rangle, \langle s', rt', hta' \rangle) = \\
\left\{ \begin{array}{l}
\sum_{s''} Res_{a'_i}(s, s'') \cdot Res_a(\langle s'', rt, hta'' \rangle, \langle s', rt', hta' \rangle) \\
\quad \text{when } \exists \langle ht_i, ha_i \rangle \in hta, \exists ht''_i, \exists ha''_i : \\
\qquad a'_i = first(ha_i) \ \wedge ht_i + t_{a'_i} \leq rt + t_a \ \wedge ht''_i = ht_i + t_{a'_i} \ \wedge ha''_i = rest(ha_i) \wedge \\
\qquad\quad hta'' = (hta \backslash \{\langle ht_i, ha_i \rangle\}) \cup \{\langle ht''_i, ha''_i \rangle\}) \wedge \\
\qquad \forall k, (k < i), \langle ht_k, ha_k \rangle \in hta : \\
\qquad\quad \forall a'_k (a'_k = first(ha_k) \Rightarrow ht_k + t_{a'_k} > ht_i + t_{a'_i}) \wedge \\
\qquad \nexists j, (j > i), \langle ht_j, ha_j \rangle \in hta : \\
\qquad\quad \forall a'_j (a'_j = first(ha_j) \Rightarrow ht_j + t_{a'_j} < ht_i + t_{a'_i}) \\
\\
Res_a(s, s') \\
\quad \text{when } rt' = rt + t_a \ \wedge \nexists \langle ht_i, ha_i \rangle \in hta : \\
\qquad \forall a'_i (a'_i = first(ha_i) \Rightarrow rt' < ht_i + t_{a'_i}) \ \wedge hta' = hta
\end{array} \right.
\end{array}
$$

By construction, the result of the application of the function Res_a to transition between states is a probability distribution. Given the probabilistic approach underlying our human-aware planner for generating valid policies, we must prove that this property still holds when we extend the function Res_a to transit between situations:

Proposition 1. *The result of the function Res_a is always a probability distribution.*

Proof. As action durations are deterministic, both on the human and the robot side, and rt and ht_i are strictly monotone, the generation of situations when a robot action is applied is a tree structured in levels, where situation $\sigma = \langle s, rt, hta \rangle$ is at level 0 with an associated probability of 1. We now prove by induction that each subsequent level k is a probability distribution over n_k situations.

In the base case (level 0), we have only one situation σ with probability $p_{0,1} = 1$, which is a probability distribution. Assume that at level k we have a probability distribution $p_{k,1}, p_{k,2}, \ldots, p_{k,n_k}$ over n_k situations $\sigma_{k,1}, \sigma_{k,2}, \ldots, \sigma_{k,n_k}$, such that $p_{k,1} + p_{k,2} + \cdots + p_{k,n_k} = 1$. We can prove that also at level $k+1$ we will still have a probability distribution over n_{k+1} situations.

If we apply action a_{k+1} to $\sigma_{k,i}$ ($0 \leq i \leq n_k$), we get a number of new situations $\sigma_{k+1,1}$, $\sigma_{k+1,2}, \ldots, \sigma_{k+1,n^i_{k+1}}$ belonging to level $k+1$. Given the starting situation $\sigma_{k,i}$, the action a_{k+1}, gives by construction a probability distribution $q_{i,1}, q_{i,2}, \ldots, q_{i,n^i_{k+1}}$ over the new situations ($q_{i,1} + q_{i,2} + \cdots + q_{i,n^i_{k+1}} = 1$). Thus, the sum of probabilities at level $k+1$ will be:

$$
\begin{array}{l}
p_{k,1} \cdot q_{1,1} + p_{k,1} \cdot q_{1,2} + \cdots + p_{k,1} \cdot q_{1,n^1_{k+1}} + \cdots + \\
\qquad p_{k,n_k} \cdot q_{n_k,1} + p_{k,n_k} \cdot q_{n_k,2} + \cdots + p_{k,n_k} \cdot q_{n_k,n^{n_k}_{k+1}} = \\
p_{k,1} \cdot (q_{1,1} + q_{1,2} + \cdots + q_{1,n^1_{k+1}}) + \cdots + p_{k,n_k} \cdot (q_{n_k,1} + q_{n_k,2} + \cdots + q_{n_k,n^{n_k}_{k+1}}) = \\
p_{k,1} \cdot 1 + \cdots + p_{k,n_k} \cdot 1 = p_{k,1} + \cdots + p_{k,n_k} = 1
\end{array}
$$

If the final situation $\sigma' = \langle s', rt', hta' \rangle$ is at level k and can be reached through different branches, then there will be multiple identical situations $\sigma_{k,i} \equiv \sigma'$ on level k and $Res_a(\sigma, \sigma')$ will return the sum of the probabilities associates with each one of them. □

As an example, illustrated in figure 2, consider three states s, s_1 and s_2, an initial situation $\langle s, 5, \{\langle 3, (watchTV, eatDinner) \rangle, \langle 4, (sleep) \rangle\}\rangle$ and a robot action *clean*, such that $t_{clean} = 5$ and

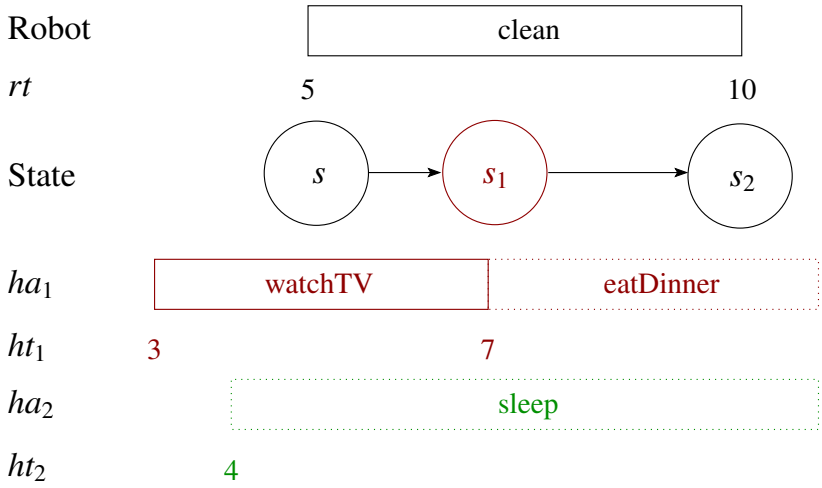

Figure 2: Example of robot and human action transitions. In the initial situation $\langle s, 5, \{\langle 3, (watchTV, eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle$ the planner tries to apply the robot action *clean* ($t_{clean} = 5$). Here, $ht_1 = 3$, $first(ha_1) = watchTV$ ($t_{watchTV} = 4$), $ht_2 = 4$ and $first(ha_2) = sleep$ ($t_{sleep} = 7$). Since $t_{clean} + rt = 10$ and $t_{watchTV} + ht_1 = 7$, the action of the robot will finish after the end of the first human action in the agenda ha_1. Therefore, the human action *watchTV* is first applied, leading to a new situation $\langle s_1, 5, \{\langle 7, (eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle$, where ht_1 and ha_1 are updated, and the effects of *watchTV* lead to a new state s_1. The preconditions of the *clean* actions are checked also in this new situation, as well as the interaction constraints between robot and human. Since both *eatDinner* and *sleep* would end after the robot action *clean*, the latter is finally applied, generating the final situation $\langle s_2, 10, \{\langle 7, (eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle$ in which *rt* is progressed and the effects of the robot action are reflected in s_2.

$Res_{clean}(s_1, s_2) = 1$. The two human agents in the environment have different agendas, that are composed by subsets of the human actions *watchTV*, *eatDinner* and *sleep* such that:

$t_{watchTV} = 4$ and $Res_{watchTV}(s, s_1) = 1$;
$t_{eatDinner} = 4$ and $Res_{eatDinner}(s_2, s_3) = 1$; and
$t_{sleep} = 7$ and $Res_{sleep}(s_3, s_4) = 1$.

For the sake of simplicity, we can assume that no one of the human actions has preconditions. In that case, the first human agent will complete *watchTV* at $ht_1 + t_{watchTV} = 3 + 4 = 7$, then the robot will complete the *clean* action at $rt + t_{clean} = 5 + 5 = 10$, as follows:

$$
\begin{aligned}
&Res_{clean}(\langle s, 5, \{\langle 3, (watchTV, eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle,\\
&\qquad \langle s_2, 10, \{\langle 7, (eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle =\\
&Res_{watchTV}(s, s_1) \cdot Res_{clean}(\langle s_1, 5, \{\langle 7, (eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle,\\
&\qquad\qquad \langle s_2, 10, \{\langle 7, (eatDinner)\rangle, \langle 4, (sleep)\rangle\}\rangle =\\
&Res_{watchTV}(s, s_1) \cdot Res_{clean}(s_1, s_2) \;=\; 1.0
\end{aligned}
$$

Actions with probabilistic outcome are also supported. In the example in figure 3, the initial situation is defined as:

$\langle s, 5, \{\langle 4, (cook, eatDinner)\rangle, \langle 4, (prepareSnack, watchTV)\rangle\}\rangle$

The actions in the agendas of the two users are defined as:

$t_{prepareSnack} = 2$,
$Res_{prepareSnack}(s, s_1) = 0.5$ and $Res_{prepareSnack}(s, s_2) = 0.5$;
$t_{cook} = 3$,

$Res_{cook}(s_1, s_3) = 0.5$, $Res_{cook}(s_1, s_4) = 0.5$,
$Res_{cook}(s_2, s_4) = 0.5$ and $Res_{cook}(s_2, s_4) = 0.5$;

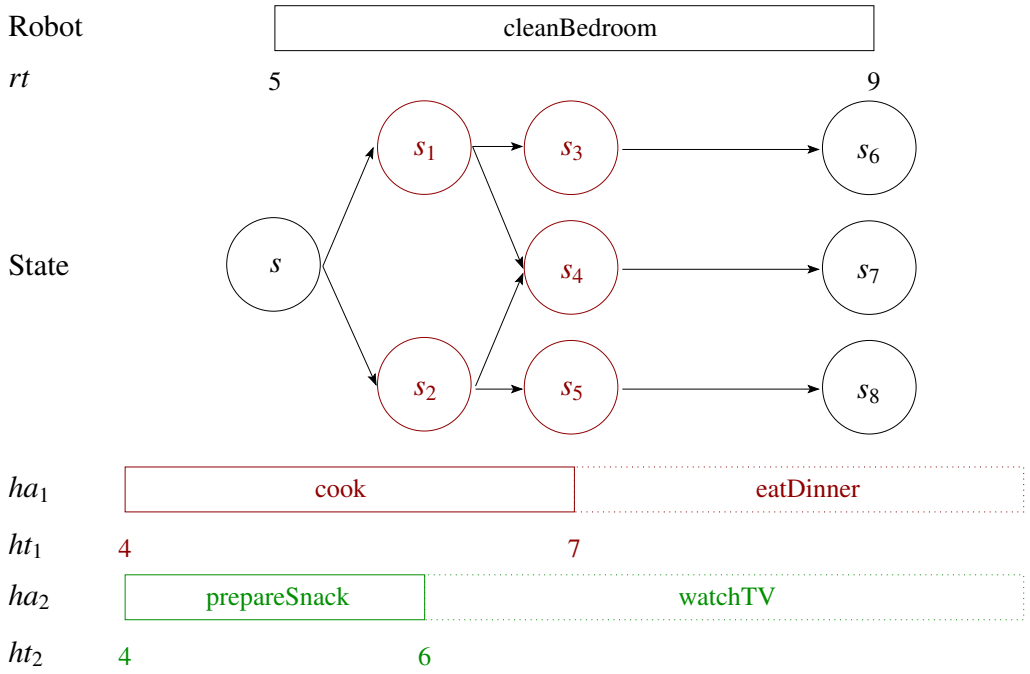

Figure 3: Example of robot and human action transitions, when actions have probabilistic outcomes. In the initial situation $\langle s, 5, \{\langle 4, (cook, eatDinner)\rangle, \langle 4, (prepareSnack, watchTV)\rangle\}\rangle$ the planner tries to apply the robot action *cleanBedroom* ($t_{cleanBedroom} = 4$). Here, $ht_1 = 4$, $first(ha_1) = cook$ ($t_{cook} = 3$), $ht_2 = 4$ and $first(ha_2) = prepareSnack$ ($t_{prepareSnack} = 2$). Since $t_{cleanBedroom} + rt = 9$, $t_{cook} + ht_1 = 7$ and $t_{prepareSnack} + ht_2 = 6$, the action of the robot will finish after the end of both human actions $first(ha_1) = cook$ and $first(ha_2) = prepareSnack$. Since it will be the first to end, $prepareSnack$ is first applied. Due to its probabilistic outcome ($Res_{prepareSnack}(s, s_1) = 0.5$, $Res_{prepareSnack}(s, s_2) = 0.5$) its application leads to two different situations, $\langle s_1, 5, \{\langle 4, (cook, eatDinner)\rangle, \langle 6, (watchTV)\rangle\}\rangle$ and $\langle s_2, 5, \{\langle 4, (cook, eatDinner)\rangle, \langle 6, (watchTV)\rangle\}\rangle$, where ht_2 and ha_2 are updated, and the effects of $prepareSnack$ lead to two different possible states, s_1 and s_2. The difference between the two states is that in s_1 the kitchen is marked as dirty, while in s_2 $prepareSnack$ had no effects on the conditions of the room. Human action *cook*, from the agenda ha_1, is then applied with the same mechanism and, finally, the planner can apply the robot action *cleanBedroom*, leading to the three final situations $\langle s_6, 9, \{\langle 7, (eatDinner)\rangle, \langle 6, (watchTV)\rangle\}\rangle$, $\langle s_7, 9, \{\langle 7, (eatDinner)\rangle, \langle 6, (watchTV)\rangle\}\rangle$ and $\langle s_8, 9, \{\langle 7, (eatDinner)\rangle, \langle 6, (watchTV)\rangle\}\rangle$.

3.3. Interaction constraints and action preconditions

Interaction constraints and human action preconditions are two different ways that our planner supports to specify how the robot should interact with the human agents.

While goals are objectives that the robot should achieve at the end of the policy execution, interaction constraints and human action preconditions are continuously verified at planning time to generate valid policies, that is, policies in which the robot performs actions needed to support the activity of the human and in which the interaction between robot and human agents is conforming to pre-defined rules. Interaction constraints, in particular, can be considered as maintenance goals.

Another important difference is that goals can be violated in some cases, leading to a less efficient, but still acceptable policy. On the other side, a violation of interaction constraints or the non-achievement of the preconditions of a human action would not lead to a valid policy.

For example, in the vacuum cleaner scenario, we could specify an *IC* stating that the robot should never clean or station in a room where, according to the human agendas, one of the users will perform some actions.

Our planner allows the specification of *IC* in the form of temporal logic formulae. The constraints are checked in every situation and, in case of violation, the situation is discarded and the corresponding branch is dismissed. An example of how interaction constraints are applied during the search is shown in figure 4. Interaction rules can also include temporal constraints for the robot, e.g., a service should be provided withing 10 minutes after the human has gone to bed.

Each *IC* can specify an interaction rule between the robot and every human agent in the environment, or a constraint between the robot and a subset of the human agents. For instance, the following interaction constraint enforces that the robot should never perform an action in a room occupied by a human:

```
always (forall r: (not (exists h: robot-in=r and human-in(h)=r)))
```

In case we design a scenario in which the robot should avoid only one of the human agent (`human1`), then the *IC* would be:

```
always (forall r: (not (robot-in=r and human-in(human1)=r)))
```

Although it would be syntactically correct to specify interaction constraints between human agents, it is semantically meaningless to express such constraints, that would only predicate on human agendas. As it is not possible to restrain the language in which interaction constraints are specified without loosing expressiveness, we add the further limitation that the constraint system must not induce constraints on human agendas. This limitation cannot be checked syntactically, but it can be checked at execution time.

While interaction constraints are checked every time that a new situation is generated, human action preconditions must be verified only at fixed moments in time. For this reason, human actions with preconditions are always instantaneous. Using again our running example of the robotic vacuum cleaner, we could imagine a human action *receiveGuests* that takes place in the livingroom:

```
name: receiveGuests
precond: dirt(livingroom) = 0
results: human-in(human1)=livigroom
time: 0
```

The precondition of this action states that the livingroom should be clean, and any policy that would lead to a situation in which at the beginning of the action the room is not clean would be discarded.

3.4. Partial observability

The framework presented above is sufficient as long as we have full observability, that is, we always know the current situation, including the agendas of the human agents. Obviously, this is seldom the case in any real world application. Therefore, we introduce partial observability [40] by defining a *belief situation* to be a probability distribution over situations. This implies that the robot can have alternative hypotheses about the current state of the world, the humans' agendas,

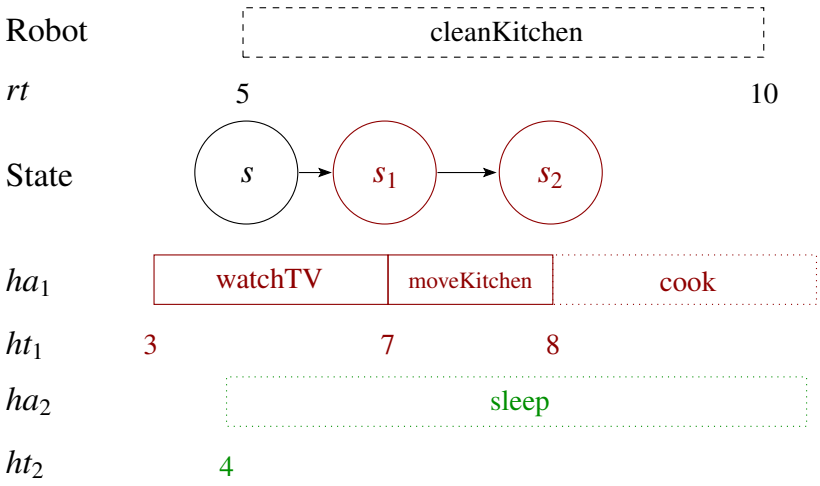

Figure 4: In the example above, an interaction constraint is specified such that the robot should never perform cleaning duties in the presence of a human. In the initial situation $\langle s, 5, \{\langle 3, (watchTV, moveKitchen, cook)\rangle, \langle 4, (sleep)\rangle\}\rangle$ ($t_{watchTV} = 4$, $t_{moveKitchen} = 1$, $t_{cook} = 3$ and $t_{sleep} = 7$) the planner tries to apply the robot action *clean* ($t_{clean} = 5$). As explained in section 3.1, the planner first checks in the human agendas (ha_1 and ha_2, in this case) if there are human actions that will end before the action of the robot. In this case, *watchTV* and *moveKitchen* from ha_1 will finish before *clean*. Therefore, two new situation are generated in sequence, $\langle s_1, 5, \{\langle 7, (moveKitchen, cook)\rangle, \langle 4, (sleep)\rangle\}\rangle$ and $\langle s_2, 5, \{\langle 8, (cook)\rangle, \langle 4, (sleep)\rangle\}\rangle$, applying the functions $Res_{watchTV}(s, s_1)$ and $Res_{moveKitchen}(s_1, s_2)$. In both situations the interaction constraint is checked and the planner detects a violation of the *IC* in s_2, because the robot would be cleaning the kitchen while one of the human agents is located there. Hence, the situation is discarded and the action is marked as not applicable.

and so on. In addition, partial observability involves *observations* as results of actions. Those are introduced in the standard way by defining a function $Obs_a : S \times O \rightarrow [0, 1]$ which specifies the probability of having an observation $o \in O$ when action a is performed resulting in state s. An observation is a set of literals, which can be empty. If a is an action by the robot, then a is assumed to be a sensing action (like testing whether a door is open or closed). If a is an action by the human, on the other hand, then o is still assumed to be an observation by the robot, but an indirect observation of a or some effect of a (if the TV is switched off, the system can observe that the *watchTV* action is over). Notice that also in the second case, the observation is dependent on the state, so we can specify that a human action only results in an observation under certain conditions.

The observation function Obs_a so far only concerns states, not situations. For the latter, it becomes a function $Obs_a : \Sigma \times \Sigma \times \Omega \rightarrow [0, 1]$ which specifies the probability of an *observation sequence* $\omega \in \Omega$ (where Ω is the set of all observation sequences) when a transition from one situation to another occurs. This observation sequence consists of the observations resulting from all human actions that can provide observations, being completed between the situation in which a robot action starts and the one in which the same action ends. The last element of the observation sequence would be the observation resulting from the robot action. Note that this can be an empty observation, if a is not a sensing action. It must also be noticed that some human actions are not observed by the system and give the empty observation. As we said (section 3.1), the order of the tuples in *hta* in every situation σ is always consistent with the humans that are considered in the planning process. Therefore, it is always possible to ascribe an observation of

the effect of a human action a_i from the agenda of the i-th tuple $\langle ht_i, ha_i \rangle$ to the i-th human agent.

As in the case of the extended Res_a function (in section 3.2), also the extended Obs_a function is composed by two parts: the recursive case when there is a human agenda ha_i whose next action a'_i finishes before or at the same time of the robot current action a, and the case in which the action a of the robot finishes before the first action of every human agenda $ha_i : \langle ht_i, ha_i \rangle \in hta$.

$$
\begin{array}{l}
Obs_a(\langle s, rt, hta \rangle, \langle s', rt', hta' \rangle, \omega) = \\
\left\{ \begin{array}{l}
\sum_{s''}(Obs_{a'_i}(s'', first(\omega)) \cdot Res_{a'_i}(s, s'') \cdot Obs_a(\langle s'', rt, hta'' \rangle, \langle s', rt', hta' \rangle, rest(\omega))) \\
\quad \text{when } \exists \langle ht_i, ha_i \rangle \in hta, \exists ht''_i, \exists ha''_i : \\
\qquad a'_i = first(ha_i) \ \wedge ht_i + t_{a'_i} \leq rt + t_a \ \wedge ht''_i = ht_i + t_{a'_i} \wedge \\
\qquad\quad ha''_i = rest(ha_i) \wedge \\
\qquad\quad hta'' = (hta \backslash \{\langle ht_i, ha_i \rangle\}) \cup \{\langle ht''_i, ha''_i \rangle\}) \wedge \\
\qquad \forall k, (k < i), \langle ht_k, ha_k \rangle \in hta : \\
\qquad\quad \forall a'_k (a'_k = first(ha_k) \Rightarrow ht_k + t_{a'_k} > ht_i + t_{a'_i}) \wedge \\
\qquad \nexists j, (j > i), \langle ht_j, ha_j \rangle \in hta : \\
\qquad\quad \forall a'_j (a'_j = first(ha_j) \Rightarrow ht_j + t_{a'_j} < ht_i + t_{a'_i}) \\
\\
Obs_a(s', first(\omega)) \cdot Res_a(s, s') \\
\quad \text{when } rt' = rt + t_a \wedge rest(\omega) = [\,] \wedge \nexists \langle ht_i, ha_i \rangle \in hta : \\
\qquad \forall a'_i (a'_i = first(ha_i) \Rightarrow rt' < ht_i + t_{a'_i}) \ \wedge hta' = hta
\end{array} \right.
\end{array}
$$

In the standard POMDP model [40], different observations result in different belief situations. As belief states are probability distributions over states, we can write $b(s)$ as the probability assigned to the state s by belief state b. It is possible to calculate the current belief state as the conditional probability distribution over the actual states, given the sequence of observations and actions so far. Given the current belief state b, the action performed a and the observation perceived o, it is possible to calculate the next belief state b' as [41]:

$$b'(s') \ = \ \frac{Obs_a(s', o) \cdot \sum_s b(s) \cdot Res_a(s, s')}{\eta}$$

We can easily adapt the formula above to calculate the new belief situation $b'(\sigma')$, that is, the probability of a situation σ' in belief situation b', resulting from an action a performed in a belief situation b with observations ω. Note that in this case the factor $Res_a(s, s')$ is part of the definition of function $Obs_a(\sigma, \sigma', \omega)$:

$$b'(\sigma') \ = \ \frac{\sum_\sigma b(\sigma) \cdot Obs_a(\sigma, \sigma', \omega)}{\eta}$$

The denominator η in the equation above is a normalizing factor, and is defined as $P(\omega|b, a)$ below. The posterior probability for a certain observation sequence ω when action a is taken in b is:

$$P(\omega|b, a) = \sum_\sigma \sum_{\sigma'} b(\sigma) \cdot Obs_a(\sigma, \sigma', \omega)$$

As there is a one-to-one correspondence between belief situations and observation sequences, we also have that $P(b'|b, a) = P(\omega|b, a)$.

3.5. Planning problem

A human-aware planning problem consists of:

1. An initial belief situation b_0, where the different situations can contain both alternative humans' agendas (as obtained from a plan recognition system), and different states.
2. A set of goals G to achieve in terms of constraints on states (logical formulae), with different values $V(g) \in [0, 1]$ such that $\sum_{g \in G} V(g) = 1$. Each $V(g)$ represents the importance value associated to the achievement of the corresponding goal.
3. A set of interaction constraints IC not to violate.
4. A set of robot actions RA.
5. A set of human actions HA.

As we said, one of the inputs that the planner receives is the set of possible agendas forecasted for the human users. The initial belief situation b_0 must contain one situation for each possible combination of the agendas of the different users. In this way, we are sure not to overlook future situations.

To show a simple yet concrete example of a planning problem we can go back once more to the robotic vacuum cleaner scenario. Two people are present in the environment, and, for each of them the planner receives two possible agendas:

$human1 : (prepareMeal, eat, goOut), (prepareMeal, eat, watchTV)$

$human2 : (sleep), (nap, watchTV)$

Therefore, the initial belief situation includes four different situations, each with a state representing the initial state of the world – as detected by the intelligent environment – , and a combination of the human agendas (where $\forall i : ht_i = 0$, $t_{prepareMeal} = 5$, $t_{eat} = 5$, $t_{goOut} = 20$, $t_{sleep} = 30$, $t_{watchTV} = 10$ and $t_{nap} = 4$):

$$\{\langle s, 0, \{\langle 0, (prepareMeal, eat, goOut)\rangle, \langle 0, (sleep)\rangle\}\rangle,$$
$$\langle s, 0, \{\langle 0, (prepareMeal, eat, goOut)\rangle, \langle 0, (nap, watchTV)\rangle\}\rangle,$$
$$\langle s, 0, \{\langle 0, (prepareMeal, eat, watchTV)\rangle, \langle 0, (sleep)\rangle\}\rangle,$$
$$\langle s, 0, \{\langle 0, (prepareMeal, eat, watchTV)\rangle, \langle 0, (nap, watchTV)\rangle\}\rangle\}$$

The set of goals includes two distinct objectives: to clean the kitchen and to clean the bedroom. In our setup, we attribute a greater value V to the former of these goals:

$$V(\texttt{dirt(kitchen)} = \texttt{0}) = 0.6$$
$$V(\texttt{dirt(bedroom)} = \texttt{0}) = 0.4$$

The IC of this human-aware planning problem is simple: the robot should never perform an action or part of an action in the same room where one of the human agents is supposed to be:

```
always (forall r: (not (exists h: robot-in=r and human-in(h)=r)))
```

Finally, we complete the description of this human-aware planning problem providing the library of human actions the system can predict (HA) and a list of actions the robot can perform (RA). In this case, we do not specify preconditions for human actions. Robot actions are parametric, as it can be seen in the following example, where the room to be cleaned is specified at planning time when the action is instantiated:

```
name: robot-clean(r)
precond: room(r) and dirt(r) > 0
results: dirt(r):=(dirt(r) - 1) and cost:=2
time: 10
```

3.6. Robot policy

The solution to a human-aware planning problem is a robot *policy*. A *policy* is a graph $\langle n_0, N, E\rangle$ where the nodes N are marked with the robot actions to perform there (n_0 is the initial node), and the edges E are marked with observation sequences (possibly empty). Some nodes are marked with the special actions *success* and *failure*, and these nodes are terminal.

The policy is executed by performing the action of the current node, and then selecting the edge matched with the observations that occur, following it to the next node, and repeating the process until a terminal node (*success*, *failure*) is reached.

A policy $\langle n_0, N, E\rangle$ is *admissible* in an initial belief situation b_0 if and only if the following conditions hold. Each node $n_i \in N$ can be assigned a belief situation b_i, in particular with b_0 assigned to n_0, such that: the preconditions of the action in n_i hold in all situations of b_i; and for each possible transition resulting from the action reaching to some other belief situation b_j and producing the observation sequence ω, there is an edge marked with ω from n_i to another node n_j that has been assigned b_j; and there are no other edges. In addition, there should be no edges from terminal nodes.

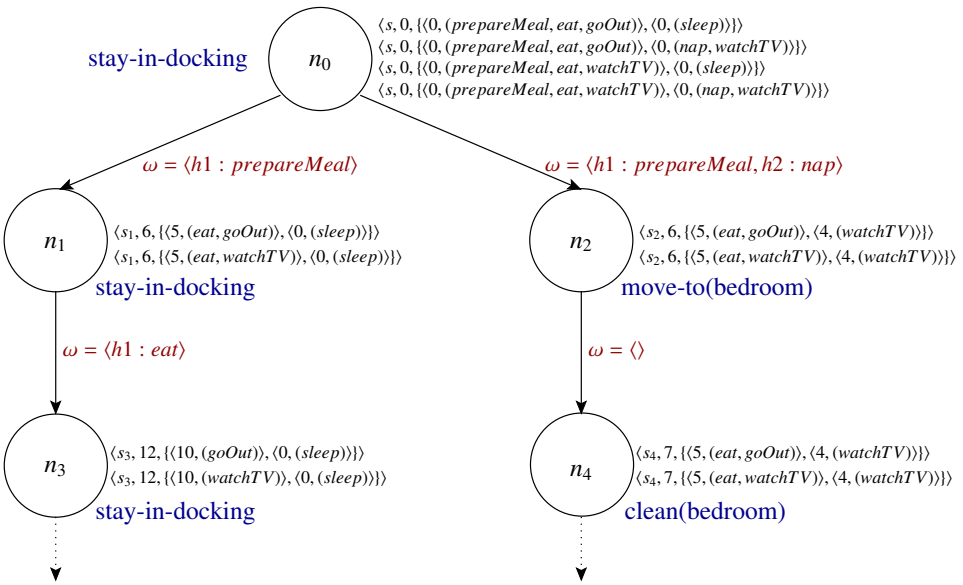

Figure 5: Example of the first nodes of a robot policy in the robotic vacuum cleaner scenario

A policy violates an interaction constraint $ic \in IC$ if and only if there is a node or an arc in it with an assigned belief situation in which ic is false.

The value (success degree) of a terminal node n_j in a policy with assigned belief situations is computed as $\Sigma_{g \in G} P(g|b_j) \cdot V(g)$, and the values of other nodes are computed (maximized) according to the Bellman equations, as is done for Markov Decision Processes [42]. The cost is computed analogously but has lower priority: if there are several policies with maximal value, the one with lowest cost will be selected. The value and cost of the entire policy are the value and cost of the initial node n_0.

Procedure `HumanAwarePlanning`$(b_0, RA, HA, IC, G, V, succ)$

```
1  B := {b_0}
2  foreach b ∈ B do
3      foreach a ∈ RA do
4          if Pre_a hold in b then
5              while computing the new {b'|P(b'|b, a) > 0} do
6                  check IC and search control
7                  progress human agendas in b
8              add the new b' to B
9              if there are b' for which some goals g ∈ G hold then
10                 compute the values of those b' using V
11                 perform Bellman update of success and cost for nodes in B
12         if termination conditions are met and value in b_0 ≥ succ then
13             return best policy
14     if no nodes in B with untried actions then
15         return failure
```

Figure 6: Algorithm for human aware planning.

A policy solves a planning problem to a degree p if: it is admissible, it only contains actions from RA, it has a value of p (or more) for the goals in G, and no interaction constraints in IC are violated. In addition, one can add requirements for the human agendas to be completed, alternatively not completed, in the *success* nodes.

Figure 5 shows an example of the first nodes of a robot policy that could be generated in the scenario described above. The initial node n_0 is assigned the initial belief situation b_0, that contains four possible situations, with the same state (s) and the four combinations of the agendas for the two human users. When the policy is generated, the planner knows which observations will become available at execution time and plans accordingly. For example, after the first waiting action (`stay-in-docking`), two possible observation sequences are available to discriminate which course of actions the users will take. On one side, one of the users (*human*1) is observed concluding the *prepareMeal* action, while no observation is available for the other user *human*2 ($\omega = \langle h1 : prepareMeal \rangle$). On the other side, the sequence includes observations on both users, as *human*2 concludes the *nap* action ($\omega = \langle h1 : prepareMeal, h2 : nap \rangle$). The system considers this information at planning time to branch in two different nodes (n_1 and n_2) and, at execution time, will choose the appropriate robot action, continuing the execution on the right branch. In this case, if the observed human activities lead to node n_2, then the robot can assume that it is safe to move to the bedroom to clean it, as neither user will occupy the room. The situations in node n_1 will obligate the robot to stay in the docking station, as the bedroom is occupied by *human*2 and there is not enough information to know if *human*1 is in the livingroom watching television or is out.

4. The Planning Algorithm

We have extended the planner PTLplan [39, 43] for probabilistic and partially observable domains to work on our representation for human-aware planning. PTLplan is a progressive planner, starting from the initial belief state, or in the extended version: belief situation, exploring belief states/situations reachable from there until a policy with sufficient value has been found. PTLplan, which itself is an extension of TLplan [44], uses temporal logic formulae to eliminate belief states/situations from the search space. The algorithm of the human aware planner (HA-PTLplan) is detailed in pseudocode in figure 6. As mentioned in the previous sections, our main extensions to the original planner are:

- The use of HA to include human actions, the addition to the belief situations of agendas consisting of such actions, and the fact that both preconditions and effects of the human actions are taken into consideration when the new belief situations are computed in steps 5 and 7;
- The possibility to generate policies with partial goal achievement and the use of weights $V(g)$ (step 10) to prioritize the accomplishment of a subset of the goals $g \in G$;
- The introduction of IC, that are checked in every state encountered while the new belief situation is computed (step 6), to avoid undesirable situations from the human side.

We refer to [39] for further technical details about the original PTLplan. We expect that other planners could be used as the basis for building a human-aware planner, by operating extensions similar to the ones above.

4.1. Termination Conditions

Since in our planning problem goals can be generated by human actions, we do not want to stop the search when we first meet a success situation, because subsequent human actions could invalidate some of the goals and a policy that greedily tries to achieve all the goals could be sub-optimal. On the other hand, we want also to avoid to plan robot actions when we have a lack of information on what the human agents are doing. We therefore adopt a search strategy in which the search is halted in every *belief situation* that either: (1) Contains at least a situation in which the agenda of one of the human agents is empty ($\exists ha_i, \langle ha_i, ht_i \rangle \in hta : rest(ha_i) = \emptyset$); or (2) Contains only human situations in which all agendas, for every human agent, have at most one action left ($\forall ha_i, \langle ha_i, ht_i \rangle \in hta : (rest(rest(ha_i)) = \emptyset \vee rest(ha_i) = \emptyset$).

4.2. Complexity

Our algorithm is based on a breadth-first search paradigm. The complexity of breadth-first search, both in memory and search time, is $O(b^d)$, where b is the number of children for each node and d is the depth of the goal. In our case, b is the number of robot action *instances* (ai_r) applicable at each planning step times the number of possible outcomes of each action.

Our algorithm stops the search when it reaches a belief situation containing situations whose agendas have zero or one action left. Therefore there are three factors that influence the size of the search space (and the time required to explore it): 1. The length in time of the human agendas that the planner receives as input; 2. The number of human agents that the planner must consider, as the initial belief situation contains one situation for each possible combination of the agendas

of the different users; 3. The use of observations, as it allows the branching of the search tree into multiple belief situations.

To calculate the order of complexity of our human-aware planning problem in the worst case scenario, when no heuristics are employed, we start with five assumptions: 1. All human agendas have the same length (in time) t_{ha}, regardless of the human agent they refer to, and are composed by the same number of actions k_{ha}; 2. The planner receives as input the same number n_{ha} of human agendas for each of the n_{humans} human agents in the environment; 3. Every human action has a fixed number of possible outcomes p (all observable); 4. Interaction constraints are never violated (no pruning of the search space); 5. All robot actions have a fixed duration t_{ra}.

Under such conditions and with only one human agent with a single agenda, and considering human actions with only one possible outcome ($p = 1$), the complexity of the search would be $O(b^{t_{ha}/t_{ra}})$. This is because we would start with one belief situation b_0 containing a single situation and the observations we could get from the human actions would not lead to branching. Therefore, since we would continue the search until we apply all the human actions in the agenda, we would need to reach a depth of t_{ha}/t_{ra}.

When n_{ha} human agendas are used for a single human agent, we would have an equal number of situations in the initial belief situation b_0. If we assume that all human actions are observable (which maximises the branching factor), this would lead to a branching of the search tree from b_0 at level 0 to n_{ha} belief situations at level 1, each containing a single situation with one human agenda. The algorithm would then apply every robot action instance to every new belief situation to obtain level 2. Therefore, the algorithm should explore n_{ha} sub trees of the same size of the one in which only one human agenda was provided. Under such conditions, the complexity of the search is $O(n_{ha} \cdot b^{t_{ha}/t_{ra}})$.

If we consider human actions with probabilistic outcomes ($p > 1$), the result above must be multiplied by the branching induced by human actions. It is straightforward to see that in such case the complexity of the search is $O(p^{k_{ha}} \cdot n_{ha} \cdot b^{t_{ha}/t_{ra}})$.

Finally, when n_{humans} agents are present, each with n_{ha} predicted agendas, we would have a $(n_{ha})^{n_{humans}}$ situations in the initial belief situation b_0, as we want to consider all possible combinations of forecasted human plans. If we assume that all human actions are observable, this would lead to a branching of the search tree from b_0 at level 0 to $n_{ha}^{n_{humans}}$ belief situations at level 1, each containing a single situation with one combination of human agendas. The algorithm would then apply every robot action instance to every new belief situation to obtain level 2. Therefore, the algorithm should explore $n_{ha}^{n_{humans}}$ sub trees of the same size of the one in which only one human agenda was provided. Moreover, the branching induced by human actions with multiple outcomes is $p^{n_{humans} \cdot k_{ha}}$. Under such conditions, the complexity of the search is $O(p^{n_{humans} \cdot k_{ha}} \cdot (n_{ha})^{n_{humans}} \cdot b^{t_{ha}/t_{ra}})$.

4.3. Pruning the search space

As seen in the previous section, using our planning algorithm without any heuristics to prune the search space becomes unpractical in any non-trivial case. In [12] we evaluated the performance of HA-PTLplan in situations where only one human agent was considered and we observed that the search space became large already when 5 agendas of 5 actions each were considered.

Although a thorough performance analysis of the current version of HA-PTLplan is out of the scope of this paper, in this section we will explain the mechanisms exploited by our planner to speed up the computation and to solve a human-aware planning problem so as to obtain the resulting policy in a time frame that is meaningful for our application scenarios.

HA-PTLplan can be provided with formulae expressed in temporal logic to prune the search space using domain knowledge. These formulae can be general, i.e., specified as separate control rules, or specific, that is, attached to a specific robot action schema and checked every time an instance of the action is tried in a belief situation.

Let's consider again our robotic vacuum cleaner scenario. Intuitively, a simple way to reduce the search space is to constrain the movements of the robot to those strictly necessary.

```
name: robot-move(r)
precond: room(r) and (not (robot-in=r))
results: robot-in:=r and cost:=1
time: 1
```

We can attach a specific control formula to the `robot-move` action schema that prevents the robot from considering any policy in which a movement doesn't involve reaching a room that needs to be cleaned or the docking station:

```
progression: dirt(r)>0 or r=robotdocking
```

In the same domain, a general control formula can then prune all those situations in which the robot arrives in a room that must be cleaned and then doesn't perform a cleaning action:

```
forall r, forall h:
  always (not ((robot-in=r and dirt(r)=d and d>0) and
               (next(dirt(r)=d and (not (human-in(h)=r))))))
```

The operator *next* specifies the check to be performed in the next belief situation relatively to the one in which the formula is first evaluated. This formula rules out any transition from a belief situation b to a belief situation b' such that: in b the robot is in a room with a dirt level $d > 0$ and in b' no human agent has entered the room and the level of dirt is unchanged.

Both general and specific control formulae can be used to prune the search space detecting inconsistencies or useless situations stemming from the actions of the robot.

The possibility of the planner to work on-line (that is, to be able to re-plan if new predictions on the humans' agendas are provided by the smart environment) is heavily dependent both on the size of the problem to be solved, and on the efficacy of the control formulae. As control formulae are domain dependent, also the possibility of our implementation of a human-aware planner to work on-line depends on the specific application. The next sections illustrate two different cases, one in which problem size and control formulae allow a continuous loop of forecast-plan-execution, and one in which the policy is generated only once and cannot be changed on-line at execution time.

5. Application Scenarios

In this section, we illustrate our approach with two realistic scenarios and we briefly describe a test run performed in a real intelligent environment, where we closed the loop from sensor data acquisition to planning and execution. The first scenario is intended to provide more practical details on the formulation of a realistic human-aware planning problem in a household setting. The second one illustrates how the human-aware planning concept can be extended beyond the scope of domestic intelligent environments.

Finally, the test run in a real environment demonstrates the real applicability of our approach, and emphasizes how it relies on an intelligent environment with non-trivial computational capabilities. Sensor data acquisition, plan forecasting, and human action recognition and monitoring [12, 16], in particular, are tasks completely delegated to the ambient ecology.

For both the application scenarios and during the complete test run in a real environment, the planner run on desktop computer, equipped with a Intel(R) Pentium(R) Dual Core 3 GHz processor and 2GB of RAM.

5.1. A domestic helper

Barbro and Per are an elderly couple still living in their apartment without special social assistance. Considering their advanced age, their son has decided to help them in their every-day life by deploying an ambient intelligent system with unobtrusive devices capable of monitoring the most important activities of the couple (e.g., their movements from one room to another).

In the environment is deployed a robotic vacuum cleaner, also equipped with a speech synthesizer. The robot is interfaced with the other ubiquitous devices and plans its actions using HA-PTLplan. Its main task is to keep the apartment clean, but it can also deliver messages to the two people, becoming the physical embodiment of the intelligent environment.

In the following, we illustrate how the planning techniques described in this paper can be used to realistically improve the integration of a robotic device into the morning schedules of the two elderly people. It is out of the scope of this paper to discuss how the morning schedules of Per and Barbro are generated: they could be learned and then identified by the environment; they could be pre-coded by the end users; or they could be generated on the fly by the environment using a set of rules and temporal reasoning. We assume that, at the beginning of the day, the planner receives from the environment the set of possible plans that Barbro and Per are likely to execute in the next three hours (from 9 am to 12 am), the goals to achieve and the interaction constraints to respect. The time granularity of the human actions is equal to 1 minute. A policy is then generated that can use the observations provided by the environment to take execution decisions.

5.1.1. Final goals

The set G of final goals of the robot specifies that all the rooms must be cleaned and that the robot has to go back to the docking station at the end of the plan. The last goal is motivated by the fact that we want to be sure that the robot is in a place where it cannot disturb Per and Barbro when we no longer have information about their activities.

The rooms the robot can access are bedroom, livingroom and kitchen. Therefore, G is composed by four goals in total, all of which have the same value ($\forall g, g \in G : V(g) = 0.25$).

5.1.2. Interaction constraint

During his morning schedule, Per needs to take his medication. This need, in our example, is injected directly into the human agenda. The policy of the domestic robot is invalidated in case it doesn't provide an alert when the medication is forgotten. The environment can observe if Per has taken his medication and such observations can be used in the policy.

The interaction constraint prunes all the policies in which the alert is not raised within 20 minutes from the time when a human agent (Per, in our example) needs to take his medication:

```
(not (exists h: human-needs-medication(h)=T and time-elapsed(h)>20))
```

5.1.3. Robot actions

The set *RA* contains 5 parametric actions the robot can perform:

```
robot-move(r)
robot-sleep(r)
robot-stay(r)
robot-clean(r)
robot-remind-medication-to-human(r,h)
```

Every action takes a parameter (`r`) that specifies the room in which the action should be performed. The last one also accepts the human agent (`h`) that should be notified. An example of the action schemas follows:

```
name: robot-clean(r)
precond: room(r) and robot-in=r and dirt(r)>0 and
         (not (exists h: (human-in(h)=r)))
results: dirt(r):=dirt(r)-1 and cost:=2
time: 10
```

The cleaning action is applicable only when the robot is in the room it intends to clean, the room needs cleaning and no human agent is present. The results are the reduction of the dirt in the room and a cost that represents battery consumption.

5.1.4. Policy generation

In the first example, the morning of the elderly couple starts at 9 am. After their toilet, they both move into the kitchen for breakfast. As usual, the bedroom is automatically marked for cleaning by the system. The system also identifies one agenda for each person, but, in the case of Per, the agenda involves taking a medication just after breakfast. Therefore, a second agenda is automatically generated to take into account the possibility that the medication has been forgotten.

The three human agendas are graphically represented at the bottom of figure 7. The initial belief situation b_0, in this case, is composed by two situations, each containing Barbro's agenda and one of the two agendas of Per.

Per spends the time after breakfast in the livingroom, reading, watching television and resting, while Barbro, after two hours, goes back to the kitchen to prepare lunch. It should be noted that the breakfast activity introduces a new goal for the robot: once the kitchen has been used, it is marked as dirty and the robot must attend to it. The small boxes in a different color that interleave the actions of the human agents represent the movements from one room to the other; these movements can be traced by the intelligent environment and used at execution time to decide which branch of the robot policy to follow. Another observation stems from the medication: at the end of the breakfast, Per can be observed (agenda **A**) – or not observed (agenda **B**)– taking his medication. This is actually the observation used by the planer to branch its policy, as can be observed in the upper part of figure 7.

The robot policy starts with the cleaning of the bedroom, which is now empty. Then, according to the observation whether Per has taken his medication or not, the robot can either move directly to clean the now empty kitchen before Barbro needs it again, or move to the livingroom to deliver the medication alert to Per before continuing with its cleaning duties.

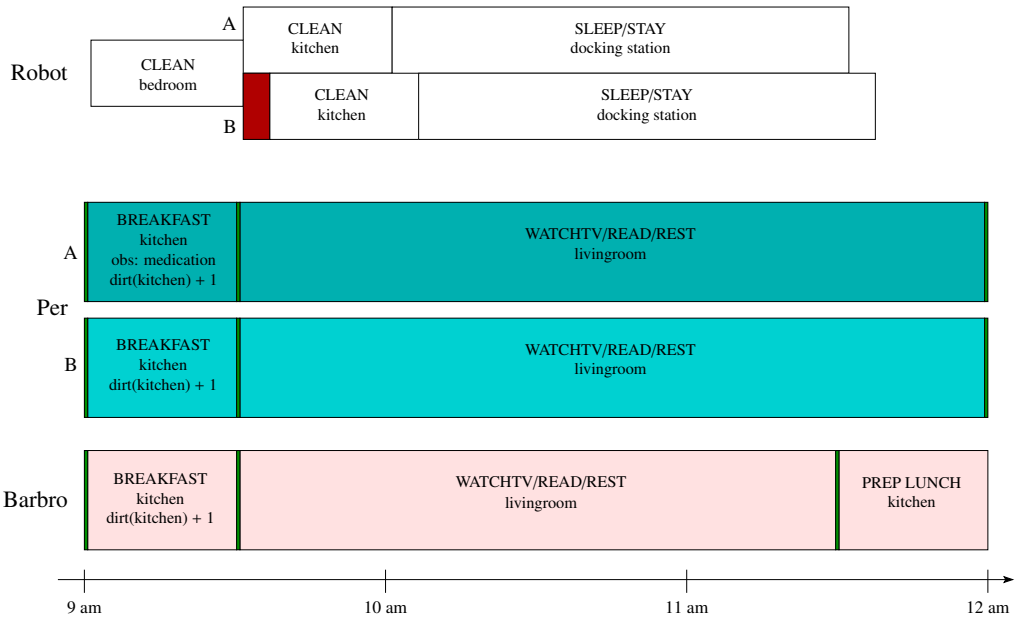

Figure 7: The human agendas (bottom) are provided as input to the planner. The policy generated for the robot (top) takes into account all the constraints that emerge from the human agendas and can use all the observations stemming from human activities to select the best course of action.

5.1.5. Speeding up the search

As seen in section 4.2, if we want to plan for a realistic time span (several hours) in environments with two people, each with several possible agendas, the search space would soon become very large. Therefore, domain knowledge is injected in the planner using general and specific control formulae (section 4.3).

An example of the heuristics used in this domain is the specific control formula attached to the `robot-move(r)` action schema:

```
:progression  (human-needs-medication(h)=T and human-in(h)=r) or
              r=robotdocking or dirt(r)>0
```

This rule allows the robot to consider a moving action only when directed to a room that needs to be cleaned, or that contains a human agent in need for a medical reminder, or else when the robot intends to go back to the docking station.

We generated the policy for the planning problem above with and without control formulae. Without using domain knowledge, the planner explored 9873 nodes and required 64 seconds to generate a solution where the interaction constraint was not violated and all the goals were achieved. Using control formulae, only 316 nodes were explored (in 2.2 seconds). The resulting policy (figure 7 at the top) was valid and all the goals were achieved.

5.1.6. Adding more agendas

Obviously in real environments the number of agendas that the planner should consider for each human is usually higher than one or two. We performed two more runs, first adding one

agenda (represented in figure 8, **A**) for Barbro and then adding also a second one (figure 8, **B**). All the agendas have the same length in time.

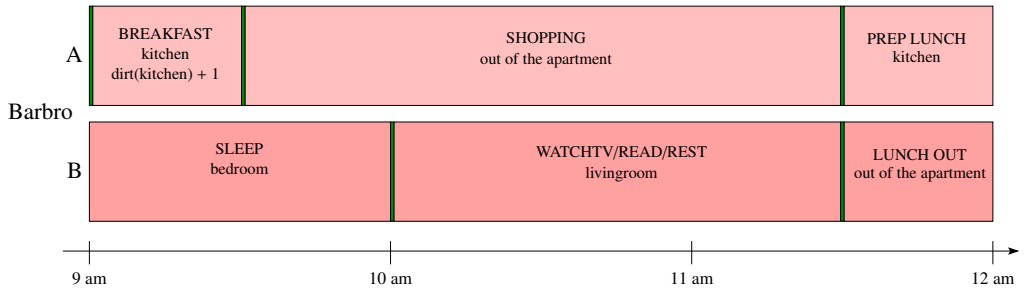

Figure 8: Two more forecasts of the possible morning activities of Barbro.

In the run in which both Barbro and Per have two agendas each, the initial belief situation b_0 is composed of four situations each, which cover all the possible combinations of the plans of the two individuals. As we expected from the complexity study in section 4.2, without domain knowledge the planner explored 19545 nodes in 131 seconds to find a successful policy. A policy with the same success rate was also found when control formulae were used. Also in this case, the nodes explored (617 in 4.6 seconds) were roughly the double of the ones needed in the run presented in the previous section.

Finally, we added also the last of Barbro agendas, in which she stays in bed longer, skipping breakfast. This means that the robot policy also considers that in some cases the bedroom could not be available for cleaning if this is the agenda that Barbro will actually follow. The policies generated with and without control formulae achieved full success. Also in this case, the domain knowledge considerably sped up the computation (from 25032 nodes in 171.1 seconds to 1368 nodes in 10.4 seconds).

As it can be seen by the experimental results, even when we consider all the agendas together, the time required by the planner to generate a valid policy in this particular scenario is well below the time granularity at which human actions are specified (1 minute). This means that, in case the environment would change its forecast of the possible human agendas, the planner could receive them and generate a new compliant policy on-line, without the necessity of taking into account planning delays.

5.2. The surgeon helper

In a department of an hospital, different surgeons share two operating rooms. A robotic helper has been deployed to assist them, bringing the needed instruments in the correct room and removing and disposing of the used material.

Although the surgeries are scheduled in advance, some problems may arise and the surgeons are allowed to file multiple schedules for the following day. In each schedule they can decide the time of the surgery, the room they want to use (if they have such a preference) and the instruments they are going to need.

The policies generated ensure that the rooms are prepared with the required material before the surgeon enters, that at the beginning of each surgery there will be no used disposable material and that, at the end of the day, all the instruments are back in the storage room, or, in case they were disposable, discarded in a dedicated location.

As soon as they enter the preparation room, the surgeons can be observed in their movements thanks to sensors distributed in the environment. Such observations generated by the intelligent environment can be used by the robot at execution time to follow the activities of the human agents and act accordingly.

An important aspect of this scenario is that the planning problems presented here are more complex than the ones solved in the previous scenarios. This is because of the higher number of human agents, objects and locations that the planner must take into account. Also, here the planner on the robot is not used on-line, as it can work during the night, and therefore the time needed for generating the policy can be in the order of hours.

5.2.1. The planning problem

The robot has three goals: to ensure that at the end of the day all the reusable instruments (e.g., x-ray machines) are back in the storage room; to dump all the disposable material in a dedicated room; and to be back at its charging station when its tasks are completed.

To ensure that the robot will not interfere with the medical staff, an interaction constraint is defined that discards every policy where the robot is in the same room as a surgeon. In this example, we also used human action preconditions: every time that a surgeon enters the room where he will work, an instantaneous action is triggered whose preconditions make sure that in the room there is no used disposable material from previous surgeries and that the instruments required by the surgeon are effectively present.

The robot is provided with a set of 5 parametric actions: it can move from room to room, pick up and put down objects, and remain idle for a short or for a long time. Obviously, considering that there are several objects of different kinds in the environment (4, in our experimental runs) and 5 different locations (the two operating rooms, the docking station, the disposal room and the storage), the number of instances of each action schema is elevated and this induces a high branching factor in the search tree.

The size of the search space is also directly related to the time span of the agendas (up to 8 hours), to the number of human agents (up to 4 in our experiments) and to the fact that each surgeon can file multiple agendas. To obtain a solution to the problem in a reasonable time (that, in this case, would be a matter of a few hours), we employed control formulas like the following:

```
(not (robot-carries-object(o)=T and object-disposable(o)=F and
      robot-in=disposal-room))
```

This rule states that a robot carrying a non-disposable object should never access the disposal-room. The formulas employed never compromised the success rate of our test runs but they provided the means to make the planning problem tractable.

5.2.2. Problem tractability

In our test runs we used 5 different agendas distributed among 4 surgeons. Each agenda had a time span from 120 to 480 time units (minutes, in our case). All the instruments needed by the surgeons, both disposable and machinery, where placed in the storage room at the beginning of each run. In all the tests, we always used a total number of 4 items, two of which disposable, and 2 operating rooms. Rooms and items can also remain unused in a run.

Already in a relatively simple scenario, with only one surgeon with a single agenda, the search space proved to be too large for a search not directed by domain knowledge. In such a case, HA-PTLplan did not manage to produce a solution of the problem in an 8 hours time

span. However, a fully successful solution was found using control formulas, exploring only 6115 nodes in less than two minutes. The generated policy satisfied all the goals, never violated interaction constraints, provided support for human action preconditions and was composed by 21 robot actions.

To check the scalability of our approach, we performed five more test runs, increasing the number of agents and agendas. In all these runs, we always employ control formulas to prune the search space. The planner found a fully successful solution in every case (that is, all the goals were achieved and no interaction constraint was violated).

In the planning problem with two surgeons with one agenda each, HA-PTLplan found a successful solution exploring 8704 nodes in about 2 minutes. The problems with three and four human agents (each with a single agenda) were solved exploring, respectively, 93679 and 251187 nodes in about 26 and 80 minutes. Finally, in the last two runs, we generated a problem with three surgeons and a total of 4 and 5 different agendas. In the first run, one of the surgeons had two agendas, thus the initial belief situation b_0 contained 2 situations. In the second run, two surgeons had two agendas each. Therefore, in this case the initial belief situation b_0 contained 4 situations, to cover all the possible combinations of human agendas. The planner solved the first problem generating a branched policy containing a total of 54 robot actions. The search space counted 142520 nodes explored in about 54 minutes. Finally, the second, and more complex problem, was solved by HA-PTLplan in less than two hours and a half exploring 298573 nodes. The branched policy, in this case, was composed by 101 robot actions.

In spite of the complexity of the problems, each policy was generated in a time span that is acceptable in this scenario, as the planner is allowed to work off-line overnight. Obviously, with problems of such complexity (many human users, objects and locations), HA-PTLplan would not be able to properly function in a closed loop, and on-line changes of users' agendas would not be possible.

5.3. Test case in a real environment

The planner we describe in this paper has been deployed in the PEIS-Home, a real robotic home environment [5], in which a variety of sensors and a robotic vacuum cleaner are deployed and where a single person is acting (see figure 9). We briefly present here a single illustrative run to demonstrate the possibility to integrate our human-aware planner in a wider, real framework, the capability of the planner to work online and, in more general terms, the feasibility of our approach to human-robot interaction.

The full deployed system is composed by three parts: a human plan recognition module [15], the planner, and an execution and monitoring module. In this test run, our system relied on the PEIS-Home infrastructure for automatic sensors' and actuators' discovery, data gathering, message dispatching, and experimental data recording functionalities.

The task of the plan recognition module is to produce an estimate of the possible agendas being executed by the human, by taking as input sensor readings and a pre-defined set of potential agendas (8, in our example). The sensors used in our experiment include a camera-based people tracker, RFID tag readers, and simple switches (e.g., to detect whether the stove was turned on or off, and to assess the state of a cupboard door). Once a policy has been generated, the executor receives from the planner the sequence of actions that the robot should perform and it sends them, one by one and with the appropriate timing, to the robot itself.

The monitoring module provides continuous support to the execution of the plan. If the plan recognition module updates the human plans identified, then a replanning signal is raised: the

Figure 9: The home environment used in our experiment.

execution is suspended, the status of the robot and the effects on the environment of the actions of the robot so far executed are passed to the planner. The planner can thus calculate a new plan, that will be consistent with both the environment status and the new human plans.

5.3.1. Test run

We have tested the system in our home environment where one user is executing daily activities in the morning, from 8 am to 1 pm, following one of eight pre-defined agendas. During the same time span, the robotic vacuum cleaner has the task to clean the floor in all the rooms that need it, avoiding interference with the user. This means that the robotic vacuum cleaner must operate and wait only in rooms which the system has predicted as not occupied by the human at that time. In our test run, all rooms were marked as dirty from the beginning, so the robot must clean all of them.

The test run was performed in real time, that means that the experiment took five hours. The data processed by the plan recognition module was collected by real sensors in real time and plan recognition, planning and plan execution were performed online. Transmission delays between the plan recognition module, the planner, and the execution and monitoring module were negligible and could be ignored.

The agenda executed by the user in our test run was the one marked **A1** in figure 10. Another agenda (**A2**) among the pre-defined ones is very similar to this one. After a delay, the plan recognition module identified these two agendas as possible given the received observations, and passed both of them to the planner as suitable candidates for the subsequent actions of the human.

In general, when more than one agenda have been identified as possible, the policy to be executed by the robot will contain branches and the observation of the last completed human activity should be used to discriminate which action to perform next. In our case, the output policy contained one branching point: the robot had to follow the policy marked **b** in case the system would observe that the human has completed the `RELAX` activity. In the absence of such observation, the robot would follow the **a** branch. It is worth noting that in both branches the robot does not go back to the kitchen after the user has had lunch. This is a design choice: the planner does not generate policies longer in time than the human agendas, since we do not want to plan actions when there is no information about the state and the activity of the human.

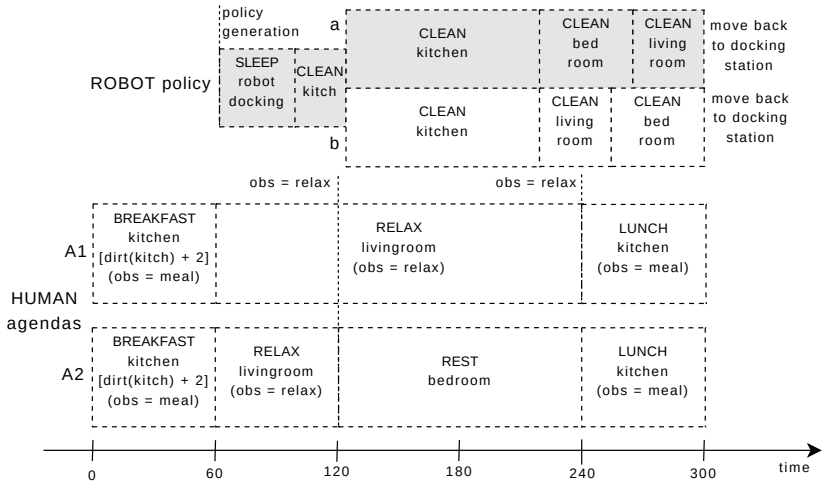

Figure 10: The robot policy generated by the planner (top) with the two agendas (bottom) identified as possible by the plan recognition module. The observation of the completion of the human activity `RELAX` allows the robot to decide which branch must be executed. The moving actions of the robot from one room to the other are omitted for clarity reasons. Only the last one is indicated at the end of the policy, as going back to the docking station is one of the goals for the robot.

In the actual run, the robot executed successfully every action marked in a darker color, and it used the observation variable `RELAX` to discern the correct course of action.

6. Conclusions and Future Work

The cohabitation of humans and robots in intelligent environments poses many new challenges. In particular, the presence of humans in the robot's work space has a significant influence on how the robot should plan its actions. The safety and comfort of the people present should always be guaranteed and that can only be achieved if the robot is aware of the humans' current and future activities and adapts its plans accordingly. An ecological approach can greatly facilitate the cohabitation, as the robot can make use of sensing and processing capabilities provided by the intelligent environment to gather information about the activities of the people present, instead of relying only on its limited perception.

In this paper, we have presented HA-PTLplan, a human aware planner designed to be part of a larger framework deployed in a real intelligent environment. The planner is able to take into account the forecasted actions of multiple humans at planning time. Such actions do not only impose constraints on the robot, e.g., never enter a room while a human is there. They can also be the source of new goals, e.g., the objects in the room should be disposed of after a human has been using them. The possibility of observing the results of some human actions can then be employed by the planner to identify the best policy to achieve the identified goals. The approach presented in this paper extends our previous work [12] in many respects, and most notably in its ability to deal with multiple humans.

We demonstrated our approach in three different scenarios, two of which were aimed at showing the capabilities of our planner in different situations involving multiple humans. The third scenario provided a demonstration of the practical applicability of our approach in the

context of a real intelligent robotic environment, the PEIS-Home. In this last experiment, HA-PTLplan was integrated in a wider framework, so as to close the loop from real-time sensor readings to plan execution.

We believe that the major technical limitation of our specific approach to human-aware planning is the use of actions with fixed duration. This is mostly a problem for the actions of the humans, as they work as constraints in the planning process. Fixed duration of robot actions can be considered a less restrictive limitation, because, as some authors have pointed out [37], one can first plan with only the expected duration of the actions and then improve or re-plan the policy with other possible/actual durations, still obtaining policies that are quite close to the optimum. In our future work we intend to overcome this limitation, expanding the planning algorithm to cope with human activities with uncertain temporal duration.

As explained in section 4.3, the effectiveness of control formulae in pruning the search space is domain dependent. Even if our approach proves efficient in a number of different scenarios, it would fail if it were to be blindly applied to a public ambient ecology, in which a multitude of individuals are present at the same time (e.g., a smart supermarket, or an intelligent train station). If we assume that the intelligent environment would be able to provide different plan forecasts for each individual, the number of initial situations derived by all possible combinations of human agendas would be unmanageable and the search space would be too vast even with efficient control formulae.

However, this would be only one way of applying human-aware planning in public ambient ecologies. If we take the example of a guidance robot in a train station [18], only a restricted number of individuals are, at a certain moment, in need of assistance. The intelligent environment could single out those few people and invoke a human-aware planner on a subset of individuals, instead of on the whole multitude. Alternatively, the environment could create agendas that describe group behaviour, aggregating people and using human-aware planners to deal with groups, instead of single persons. In our future work, we intend to explore both possibilities.

Another interesting research direction will be to test our system in more structured environments than household settings, like a factory production line, where the human agendas are less flexible because they are defined in strict protocols. In particular, the actions of a worker in hazardous areas are strictly decided and timed beforehand, making such scenario the perfect candidate for applying our approach. This would enable the cooperation of a robot helper, minimizing the exposition of the workers to dangerous situations. We already explored a first scenario in which human-aware planning is applied to a factory environment [45], but further investigation is required before our technique could be integrated with industrial realities.

Finally, we believe that extending other search strategies and different planners to work with the formalization of a human-aware planning problem as described in this paper could lead to a performance improvement of the full deployed system, and open new possibilities for future improvements.

Acknowledgments

This work has been partially supported by CUGS (Swedish national computer science graduate school) and by the Swedish KK foundation. We are grateful to Federico Pecora and Rachid Alami for many useful discussions on human-aware planning.

References

[1] A. Sanfeliu, N. Hagita, A. Saffiotti, Robotics and Autonomous Systems 56 (10) (2008) Special Issue on Network Robot Systems.

[2] J.-H. Kim, Y.-D. Kim, K.-H. Lee, The Third Generations of Robotics: Ubiquitous Robot, in: Proceedings of the 2nd International Conference on Autonomous Robots and Agents, 2004.

[3] A. Sgorbissa, R. Zaccaria, The artificial ecosystem: a distributed approach to service robotics, in: Proceedings of the International Conference on Robotics and Automation (ICRA), 2004.

[4] C. Goumopoulos, A. Kameas, Ambient Ecologies in Smart Homes, The Computer Journal 52 (8) (2008) 922–937.

[5] A. Saffiotti, M. Broxvall, M. Gritti, K. LeBlanc, R. Lundh, J. Rashid, B. Seo, Y. Cho, The PEIS-Ecology Project: Vision and Results, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2008.

[6] A. Tapus, M. Mataric, B. Scassellati, The grand challenges in socially assistive robotics, *IEEE* Robotics & Automation Magazine 14 (1) (2007) 35–42.

[7] B. Graf, M. Hägele, Dependable Interaction with an Intelligent Home Care Robot, in: Proceedings of ICRA-Workshop on Technical Challenge for Dependable Robots in Human Environments, 21–26, 2001.

[8] A. Clodic, H. Cao, S. Alili, V. Montreuil, R. Alami, R. Chatila, Shary: a Supervision System Adapted to Human-Robot Interaction, in: Proceedings of the 11th International Symposium on Experimental Robotics (ISER), 2008.

[9] D. Cook, J. Augusto, V. Jakkula, Ambient Intelligence: Technologies, applications and opportunities, Pervise and Mobile Computing 5 (2009) 277–298.

[10] D. Nau, G. M., P. Traverso, Automated Planning: Theory & Practice, Morgan Kaufmann Publishers Inc., ISBN 1-55860-856-7, 2004.

[11] G. Hoffman, C. Breazeal, Cost-Based Anticipatory Action Selection for Human–Robot Fluency, IEEE Transactions on Robotics 23 (5) (2007) 952–961.

[12] M. Cirillo, L. Karlsson, A. Saffiotti, Human-Aware Task Planning: an Application to Mobile Robots, ACM Transactions on Intelligent Systems and Technology Special Issue on Applications of Automated Planning (2010) 15:1–15:26.

[13] A. Goultiaeva, Y. Lesperance, Incremental Plan Recognition in an Agent Programming Framework, in: Working Notes of the AAAI Workshop on Plan, Activity, and Intention Recognition, 2007.

[14] U. Naeem, J. Bigham, A Comparison of Two Hidden Markov Approaches to Task Identification in the Home Environment, in: Proceedings of the 2nd International Conference on Pervasive Computing and Applications (ICPCA), 383–388, 2007.

[15] M. Cirillo, F. Lanzellotto, F. Pecora, A. Saffiotti, Monitoring Domestic Activities with Temporal Constraints and Components, in: Proceedings of the 5th International Conference on Intelligent Environments (IE), 2009.

[16] M. Cirillo, F. Pecora, A. Saffiotti, Proactive Assistance in Ecologies of Physically Embedded Intelligent Systems: A Constraint-Based Approach, in: N.-Y. Chong, F. Mastrogiovanni (Eds.), Handbook of Research on Ambient Intelligence and Smart Environments: Trends and Perspective, vol. 370, IGI Global, ISBN 978-1-61692-857-5, 534–557, 2011.

[17] D. Lee, T. Yamazaki, S. Helal, Robotic Companions for Smart Space Interactions, Pervasive Computing, IEEE 8 (2) (2009) 78 –84.

[18] M. Shiomi, D. Sakamoto, T. Kanda, C. T. Ishi, H. Ishiguro, N. Hagita, A Semi-Autonomous Communication Robot: a Field Trial at a Train Station, in: Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction (HRI), 2008.

[19] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, N. Hagita, An Affective Guide Robot in a Shopping Mall, in: Proceedings of the 4rd ACM/IEEE International Conference on Human Robot Interaction (HRI), 2009.

[20] B. Mazzolai, V. Mattoli, C. Laschi, P. Salvini, G. Ferri, G. Ciaravella, P. Dario, Networked and Cooperating Robots for Urban Hygiene: the EU funded DustBot Project, in: Proceedings of the 5th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), 2008.

[21] A. Sanfeliu, J. Andrade-Cetto, Ubiquitous Networking Robotics in Urban Settings, in: IROS Workshop on Network Robot Systems. Toward Intelligent Robotic Systems Integrated with Environments, 2006.

[22] Z. Shi, J. Wei, X. Liu, Z. Wang, J. Tu, IRGS Protocol based mobile service robot positioning and multi-robot collaboration for smart home, in: Proceedings of the 30th Chinese Control Conference (CCC), 2011.

[23] B. Graf, M. Hans, R. Schraft, Mobile Robot Assistants: issues for dependable operation in direct cooperation with humans, *IEEE* Robotics and Automation Magazine 11 (2) (2004) 67–77.

[24] E. Akin Sisbot, A. Clodic, L. Marin, M. Fontmarty, L. Brèthes, R. Alami, Implementing a human-aware robot system, in: Proceedings of the ROMAN Symposium, 727–732, 2006.

[25] A. Pandey, R. Alami, A Framework for Adapting Social Conventions in a Mobile Robot Motion in Human-Centered Environment, in: Proceedings of the International Conference on Advanced Robotics (ICAR), 2009.

[26] E. Sisbot, L. Marin, R. Alami, Spatial Reasoning for Human Robot Interaction, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2007.
[27] R. Alami, A. Clodic, V. Montreuil, E. Sisbot, R. Chatila, Toward Human-Aware Robot Task Planning, in: AAAI Spring Symposium 'To boldly go where no human-robot team has gone before', 39–46, 2006.
[28] F. Broz, I. Nourbakhsh, R. Simmons, Planning for Human-Robot Interaction Using Time-State Aggregated POMDPs, in: Proceedings of the 23rd Conference on Artificial Intelligence (AAAI), 2008.
[29] V. Montreuil, A. Clodic, R. Alami, Planning Human Centered Robot Activities, in: Proceedings of the 2007 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2007.
[30] C. Galindo, J. Fernández-Madrigal, J. González, Multi-Hierarchical Interactive Task Planning. Application to Mobile Robotics, IEEE Transactions on Systems, Man, and Cybernetics part B 18 (3) (2008) 785–789.
[31] F. Gravot, R. Alami, An Extension of the Plan-Merging Paradigm for Multi-Robot Coordination, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2001.
[32] J. Blythe, Planning with external events, in: UAI '94: Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence, 1994.
[33] B. Grosz, S. Kraus, Collaborative plans for complex group action, Artificial Intelligence 86 (2) (1996) 269–357.
[34] A. Hiolle, K. Bard, L. Canamero, Assessing Human Reactions to Different Robot Attachment Profiles, in: Proceedings of the 18th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2009.
[35] M. K. Lee, J. Forlizzi, P. E. Rybski, F. Crabbe, W. Chung, J. Finkle, E. Glaser, S. Kiesler, The Snackbot: Documenting the Design of a Robot for Long-term Human-Robot Interaction, in: Proceedings of the 4rd ACM/IEEE International Conference on Human Robot Interaction (HRI), 2009.
[36] G. Cortellessa, G. Koch Svedberg, L. A., F. Pecora, M. Scopelliti, L. Tiberio, A Cross-Cultural Evaluation of Domestic Assitive Robots, in: Proceedings of the AAAI Fall Symposium on AI and Eldercare, 2008.
[37] Mausam, D. Weld, Planning with Durative Actions in Stochastic Domains, Journal of Artificial Intelligence Research 31 (2008) 33–28.
[38] P. Haslum, H. Geffner, Heuristic Planning with Time and Resources, in: Proceedings 6th European Conference on Planning, 2001.
[39] L. Karlsson, Conditional progressive planning under uncertainty, in: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI), 431–438, 2001.
[40] L. Kaebling, M. Littman, A. Cassandra, Planning and Acting in Partially Observable Stochastic Domains, Artificial Intelligence 101 (1–2) (1998) 99–134.
[41] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach (SE), Prentice Hall, 2003.
[42] M. L. Puterman, Markov Decision Processes: discrete stochastic dynamic programming, John Wiley & sons, 1994.
[43] A. Bouguerra, L. Karlsson, Symbolic Probabilistic-Conditional Plans Execution by a Mobile Robot, in: IJCAI-05 Workshop: Reasoning with Uncertainty in Robotics (RUR-05), 2005.
[44] F. Bacchus, F. Kabanza, Using temporal logics to express search control knowledge for planning, Artificial Intelligence 116 (1) (2000) 123–191.
[45] M. Cirillo, Planning in Inhabited Environments: Human-Aware Task Planning and Activity Recognition, Ph.D. thesis, Örebro University, 2010.