Relational Approach to Knowledge Engineering for POMDP-based Assistance Systems with Encoding of a Psychological Model

Marek Grześ, Jesse Hoey & Shehroz Khan
School of Computer Science
University of Waterloo, Canada

Alex Mihailidis & Stephen Czarnuch
Dept. of Occupational Science and Occupational Therapy
University of Toronto, Canada

Dan Jackson
School of Computing Science
Newcastle University, UK

Andrew Monk
Department of Psychology
University of York, UK

Abstract

Partially observable Markov decision process (POMDP) models have been used successfully to assist people with dementia when carrying out small multi-step tasks such as hand washing. POMDP models are a powerful, yet flexible framework for modeling assistance that can deal with uncertainty and utility. Unfortunately, POMDPs usually require a very labor intensive, manual setup procedure. Our previous work has described a knowledge driven method for automatically generating POMDP activity recognition and context sensitive prompting systems for complex tasks. We call the resulting POMDP a SNAP (SyNdetic Assistance Process). In this paper, we formalise this method using a relational database. The database encodes the goals, action preconditions, environment states, cognitive model, user and system actions, as well as relevant sensor models, and automatically generates a valid POMDP model of the assistance task. The strength of the database is that it allows constraints to be specified, such that we can verify the POMDP model is, indeed, valid for the task. To the best of our knowledge, this is the first time the MDP planning problem is formalised using a relational database. We demonstrate the method on three assistance tasks: handwashing, and toothbrushing for elderly persons with dementia, and on a factory assembly task for persons with a cognitive disability.

1 Introduction

Quality of life (QOL) of persons with a cognitive disability (e.g. dementia, developmental disabilities) is increased significantly if they can engage in "normal" routines in their own homes, workplaces, and communities. However, they generally require some assistance in order to do so. For example, difficulties performing activities of daily living at home, such as preparing food, washing, or cleaning, or in the workplace, such as factory assembly, may trigger the need for personal assistance or relocation to residential care settings (Gill and Kurland 2003). Moreover, it is associated with diminished QOL, poor self-esteem, anxiety, and social isolation for the person and their caregiver (Burns and Rabins 2000).

Technology to support people in their need to live independently is currently available in the form of personal and social alarms and environmental adaptations and aids. Looking to the future, we can imagine intelligent, pervasive computing technologies using sensors and effectors that help with more difficult cognitive problems in planning, sequencing and attention. In the example of assisting people with dementia, the smart environment would prompt whenever the residents get stuck in their activities of daily living.

The technical challenge of developing useful prompts and a sensing and modelling system that allows them to be delivered only at the appropriate time is difficult, due to issues such as the system needing to be able to determine the type of prompt to provide, the need for the system to recognize changes in the abilities of the person and adapt the prompt accordingly, and the need to give different prompts for different sequences within the same task. However, such a system has been shown to be achievable through the use of advanced planning and decision making approaches. One of the more sophisticated of these types of systems is the COACH (Hoey et al. 2010). COACH uses computer vision to monitor the progress of a person with dementia washing their hands and prompts only when necessary. COACH uses a partially observable Markov decision process (POMDP), a temporal probabilistic model that represents a decision making process based on environmental observations. The COACH model is flexible in that it can be applied to other tasks (Hoey et al. 2005). However, each new task requires substantial re-engineering and re-design to produce a working assistance system, which currently requires massive expert knowledge for generalization and broader applicability to different tasks. An automatic generation of such prompting systems would substantially reduce the manual efforts necessary for creating assistance systems, which are tailored to specific situations and tasks, and environments. In general, the use of *a-priori* knowledge in the design of assistance systems is a key unsolved research question. Researchers have looked at specifying and using ontologies (Chen et al. 2008), information from the Internet (Pentney, Philipose, and Bilmes 2008), logical knowledge bases (Chen et al. 2008; Mastrogiovanni, Sgorbissa, and Zaccaria 2008), and programming interfaces for context aware human-computer interaction (Salber, Dey, and Abowd 1999).

In our previous work, we have developed a knowledge driven method for automatically generating POMDP activity recognition and context sensitive prompting systems (Hoey et al. 2011). The approach starts with a description of a task and the environment in which it is to be carried out that is relatively easy to generate. Interaction Unit (IU) analysis (Ryu and Monk 2009), a psychologically motivated method for transcoding interactions relevant for fulfilling a certain task, is used for obtaining a formalized, i.e., machine interpretable task description. This is then combined with a specification of the available sensors and effectors to build a work-

ing model that is capable of analyzing ongoing activities and prompting someone. We call the resulting model a SyNdetic Assistance Process or SNAP. However, the current system uses an ad-hoc method for transcoding the IU analysis into the POMDP model. While each of the factors are well defined, fairly detailed and manual specification is required to enable the translation.

The long-term goal of the approach presented in this paper is to allow end-users, such as health professionals, caregivers, and family members, to specify and develop their own context sensitive prompting systems for their needs as they arise. This paper describes a step in this direction by defining a relational database that serves to mediate the translation between the IU analysis and the POMDP specification. The database encodes the constraints required by the POMDP in such a way that, once specified, the database can be used to generate a POMDP specification automatically that is guaranteed to be valid (according to the SNAP model). According to the best of our knowledge, this is the first time the MDP planning problem is formalised using a relational database. This novel approach helps coping with a number of issues, such as validation, maintenance, structure, tool support, association with a workflow method etc., which were identified to be critical for tools and methodologies which could support knowledge engineering in planning (McCluskey 2000). This paper gives the details of this relational database, and shows how it solves these various issues. It then demonstrates the application of this method to specify a POMDP in three examples: two are for building systems to assist persons with dementia during activities of daily living, and one is to assist persons with Down's syndrome during a factory assembly task. We show how the method requires little prior knowledge of POMDPs, and how it makes specification of relatively complex tasks a matter of a few hours of work for a single coder.

The remainder of this paper is structured as follows. First, we give an overview of the basic building blocks: Knowledge engineering requirements, POMDPs, and IU analysis. Then, Section 3 describes the relational database we use, frames the method as a statistical relational model, and shows how the database can be leveraged in the translation of IU analysis to POMDP planning system. Section 4 shows how the method can be applied to three tasks, and then the paper concludes.

2 Overview of the method

2.1 Requirements from Knowledge Engineering

The IU analysis and the sensor specification need to be translated into a POMDP model, and then the policy of action can be generated. The relational database provides a natural link between these two elements of the prompting system, and the use of the database represents additionally a novel approach to knowledge engineering (KE) for planning. For an extensive review of challenges which KE for planning faces, the reader is referred to (McCluskey 2000). This area is essentially investigating the problem of how planning domain models can be specified by technology designers who are not necessarily familiar with the AI planning technology. In (McCluskey 2000), authors collected a number of requirements which such a methodology should satisfy. Some of most important ones are: (1) acquisition, (2) validation, (3) maintenance, and additionally the representation language should be: (4) structured, (5) associated with a workflow method, (6) easy to assess with regard to the complexity of the model, (7) tool supported, (8) expressive and customizable, and (9) with a clear syntax and semantics. In our work on the SNAP process, we found that these requirements can be to a great extent supported when one applies the relational database formalism to store and to process the domain model. The acquisition step (1) does not have its full coverage in our case since, e.g., the types of planning actions are known, as well as the structure of the IU analysis. This allows specifying the structure of the relational database and designing SQL-tables beforehand and reusing one database model (see Section 3) in all deployments of the system. The database technology is a standard method of storing data, and checking validation (2) of the data is highly supported. This includes both simple checks of data types, as well as arbitrarily complex integrity checks with the use of database triggers. Once the database of a particular instance is populated, the designer can automatically generate a SNAP for a particular user/task/environment combination taking input for the sensors through the ubiquitous sensing technician's interface, and the POMDP can be fed into the planner, and then simulated. Since, the overall process is straightforward for the designer, this allows for a traditional dynamic testing of the model, where the designer can adjust the domain model easily via the database interface, generate a new POMDP file, and then simulate it and assess its prompting decisions. This shows that also maintenance (3) is well supported in our architecture. The SQL relational language is also flexible in representing structured (4) objects. In our work, it is used in conjunction with a workflow method (5), where the technology designer follows specific steps which require populating specific tables in the database. The relational database technology is one of the most popular ways of storing data, and it is vastly supported by tools and those tools are nowadays becoming familiar even to a standard computer user. In our implementation, a PHP-based web interface is used, which from the user's point of view does not differ from standard database-based systems.

2.2 Partially observable Markov decision processes

A POMDP is a probabilistic temporal model of a system interacting with its environment (Åström 1965), and is described by (1) a finite set of state variables, the cross product of which gives the state space, S; (2) a set of observation variables, O (the outputs of some sensors); (3) a set of system actions, A; (4) a reward function, $R(s, a, s')$, giving the relative utility of transiting from state s to s' under action a; (5) a stochastic transition model $Pr : S \times A \rightarrow \Delta S$ (a mapping from states and actions to distributions over states), with $Pr(s'|s, a)$ denoting the probability of moving from state s to s' when action a is taken; and (6) a stochastic observation model with $Pr(o|s)$ denoting the probability of making observation o while the system is in state s. Figure 1(a) shows a POMDP as a Dynamic Bayesian network (DBN) with actions and rewards, where arrows are interpretable as causal links between variables.

2.3 Specifying the task: Interaction Unit Analysis

Task analysis has a long history in Human Factors (Kirwan and Ainsworth 1992) where this approach is typically used to help define and break-down 'activities of daily living'

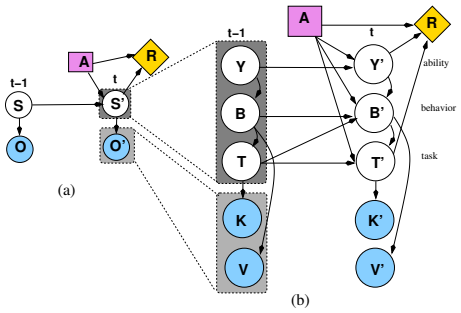

Figure 1: Two time slices of (a) a general POMDP; (b) a factored POMDP for interactions with assistive technology.

(ADL)– i.e. activities that include self-care tasks, household duties, and personal management such as paying bills. The emphasis in task analysis is on describing the actions taken by a user and the intentions (goals and sub-goals) that give rise to those actions. There has been less emphasis on how actions are driven by the current state or changes in the environment. Syndetic modeling (Duke et al. 1998) remedies this omission by describing the conjunction of cognitive and environmental precursors for each action. Modeling both cognitive and environmental mechanisms at the level of individual actions turns out to be much more efficient than building separate cognitive and environmental models (Ryu and Monk 2009).

The task analysis technique (Wherton and Monk 2009), breaks a task down into a set of goals, states, abilities and behaviours, and defines a hierarchy of tasks that can be mapped to a POMDP, a policy for which will be a situated prompting system for a particular task (Hoey et al. 2011). The technique involves an experimenter video-taping a person being assisted during the task, and then transcribing and analysing the video. The end-result is an Interaction Unit (IU) analysis that uncovers the states and goals of the task, the client's cognitive abilities, and the client's actions. A simplified example for the first step in tea-making (getting out the cup and putting in a tea-bag) is shown in Table 1. The rows in the table show a sequence of steps, with the client's current goals, the current state of the environment, the abilities that are necessary to complete the necessary step, and the behaviour that is called for. The abilities are broken down into ability to *recall* what they are doing, to *recognise* necessary objects like the kettle, and to perceive *affordances* of the environment.

A second stage of analysis involves proposing a set of sensors and actuators that can be retrofitted to the user's environment for the particular task, and providing a specification of the sensors that consists of three elements: (1) a name for each sensor and the values it can take on (e.g. on/off); (2) a mapping from sensors to the states and behaviours in the IU analysis showing the evidentiary relationships, and (3) measurements of each sensor's reliability at detecting the states/behaviours it is related to in the mapping.

The IU analysis (e.g. Table 1) can be converted to a POMDP model by factoring the state space as shown in Figure 1(b). The method is described in detail in (Hoey et al. 2011), here we give a brief overview. The *task* variables are a characterisation of the domain in terms of a set of high-level variables, and correspond to the entries in the state column in Table 1. For example, in the first step of tea making, these include the box condition (open, closed) and the cup contents (empty or with teabag). The task states are changed by the client's *behavior*, B, a single variable with values for each behaviour in Table 1. For the first IU group in tea making, these include opening/closing the box, moving the teabag to the cup, and doing nothing or something unrelated (these last two behaviours are always present). The client's *abilities* are their cognitive state, and model the ability of the client to recall (Rl), recognise (Rn) and remember affordances (Af). For the first IU group, these include the ability to recognise the tea box and the ability to perceive the affordance of moving the teabag to the cup.

The system actions are prompts that can be given to help the client regain a lost ability. We define one system action for each necessary ability in the task. The actions correspond to a prompt or signal that will help the client with this particular ability, if missing. *Task* and *behavior* variables generate observations, O. For example, in a kitchen environment there may be sensors in the countertops to detect if a cup is placed on them, sensors in the teabags to detect if they are placed in the cup, and sensors in the kettle to detect "pouring" motions. The sensor noise is measured independently (as a miss/false positive rate for each state/sensor combination) (Pham and Olivier 2009; Hoey et al. 2011).

3 Relational Database

The technology designer should be able to define the prompting system (planning problem specification) easily and with minimal technical knowledge of planning. The approach we are proposing in this paper is to provide a relational database which can be populated by the designer using standard database tools such as forms or web interface, and then translating the database into the POMDP specification using a *generator* software, which implements parts of the overall relational model not stored in the database. This is in accordance with the main goal of the way relational databases should be used. The database in itself explicitly stores the minimum amount of information which is sufficient to represent the concept. Those relations which are not represented explicitly are then extracted on demand using SQL queries. We adhere to this standard in our design and our generator contains such implicit parts of the model which are not stored in the database. Below, we show how our methodology of specifying planning tasks is motivated and justified by locating this work in the context of relevant AI research on planning, and probabilistic and relational modelling.

In the application areas which are considered in this paper, planning problems are POMDPs. POMDPs can be seen as Dynamic Decision Networks (DDNs). In POMDP planners, DDNs have propositional representation, where the domain has a number of attributes, and attributes can take values from their corresponding domains. The problem with designing methodologies for such propositional techniques is that the reuse of the model in new instances is not straightforward, and a relational approach becomes useful. In the case of modelling POMDPs, Statistical Relational Learning (Getoor and Taskar 2007) is the way to make relational specification of DDNs possible.

IU	Goals	Task States	Abilities	Behaviours
1	Final	cup empty on tray, box closed	Rn cup on tray, Rl step	No Action
2	Final, cup TB	cup empty on tray, box closed	Af cup on tray WS	Move cup tray→WS
3	Final, cup TB	cup empty on WS, box closed	Rl box contains TB, Af box closed	Alter box to open
4	Final, cup TB	cup empty on WS, box open	Af TB in box cup	Move TB box→cup
5	Final	cup tb on WS, box open	Af box open	Alter box to closed
	Final	cup tb on WS, box closed		

Table 1: IU analysis of the first step in tea making. Rn=recognition, Rl=Recall, Af=Affordance, tb=teabag, ws=work surface.

3.1 Statistical Relational Learning

Probabilistic Relational Models (PRM) define a template for a probability distribution (Getoor and Taskar 2007), that specifies a concrete distribution when ground with specific data. The family of distributions that can be obtained from a specific PRM is what we seek in the problem of specifying POMDPs for prompting systems. Our goal is to have a template which would be flexible and general enough to represent POMDPs for different tasks, but also specific enough so that one relational model would be sufficient. Figure 2 shows the main components of the probabilistic model specified using the PRM.

The first element is the relational schemata which can be formalised as a specification of types of objects, their attributes, and relations between objects of specific types. The two additional components are: for each attribute the set of parents this attribute probabilistically depends on, and the corresponding conditional probability distributions. These elements together with the relational structure is exactly what is shared between different prompting systems which we build for cognitively disabled people, and this part of the model can be designed beforehand using the outcomes of years of research in this area which ranges from artificial intelligence to cognitive psychology. We argue in this paper, that the relational database is a good way of representing and specifying these kinds of models which define POMDP planning problems. The PRM part of the model is exactly what can be designed beforehand by POMDP planning experts and cognitive scientists, and every particular deployment of the system will be reduced to populating the database by the technology designer who is not required to have POMDP specific knowledge.

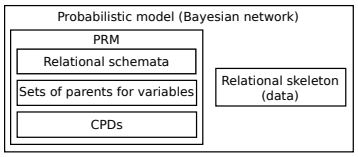

Figure 2: The probabilistic model and its components when specified as a PRM.

The relational schemata of the PRM can be represented directly as a standard relational database where tables and implicit tables determined by SQL queries define objects, and columns in tables define attributes. Relationships are modelled as primary/foreign key constraints.

The two remaining elements of the PRM are also partially incorporated in the relational database, defined as SQL queries to the database, or explicitly encoded in the software which reads the database and produces the final input file for the POMDP planner. The PRM model contains everything which is required to obtain the probabilistic model for the specific case, except for the data – objects, values of their attributes, values of probabilities, and specifications of some dependencies which are represented relationally in the model – and this complementary element is named a relational skeleton in (Getoor and Taskar 2007). This skeleton contains objects which adhere to relational constraints defined in the relational model (database tables, SQL queries, the generator implementation). Once this skeleton is provided, the PRM can be translated into a ground Bayesian network in the original case, and into a ground DDN in our implementation which has to model time (two slice Bayesian network) and decisions.

3.2 Relational Schemata

The above discussion showed how our methodology originates from the state-of-the-art methods for relational probabilistic modelling. In this paragraph, we show technical details of how the database was designed. Figure 3 shows the structure of the entire database. All tables which have their names starting with `t_iu_` represent the IU table, and the user interface shows the view of the full IU table to the user (not individual tables separately). The core table is `t_env_variables_values` which stores domain attributes and their possible values. The sensor model, `t_sensor_model`, associates environment variables with sensors (`t_observations_values`). There is also a sensor model for behaviours in the corresponding table. There is a table for possible behaviours of the client in the modelled domain (`t_behaviours`) and dynamics of the client's actions are defined in associated tables which store effects and preconditions of behaviours. Essentially, the database encoding of effects and preconditions of client's actions contains information which is equivalent to STRIPS operators in classical planning. One behaviour can have different effects depending on the precondition. Additionally, our probabilistic model can make use of the states in which a specific behaviour is impossible (`t_when_is_behaviour_impossible`). Table `t_abilities` stores client's abilities which are relevant to the task. Finally, rewards are defined in `t_rewards` and the associated table allows specifying sets of states which yield a particular value of the reward.

3.3 Relational Probabilistic Model

We present only one piece of the relational specification of probabilistic dependencies: the client behaviour dynamics model.

Let us assume that I is a set of rows in the IU table. T, T', B, B', Y, and Y' are as specified in Figure 1b. ρ is a random behaviour constant and is set to 0.01 in the current implementation. We define the following functions:

1. $row\_rel : I \times T \rightarrow \{0, 1\}$ is 1 for task states relevant in

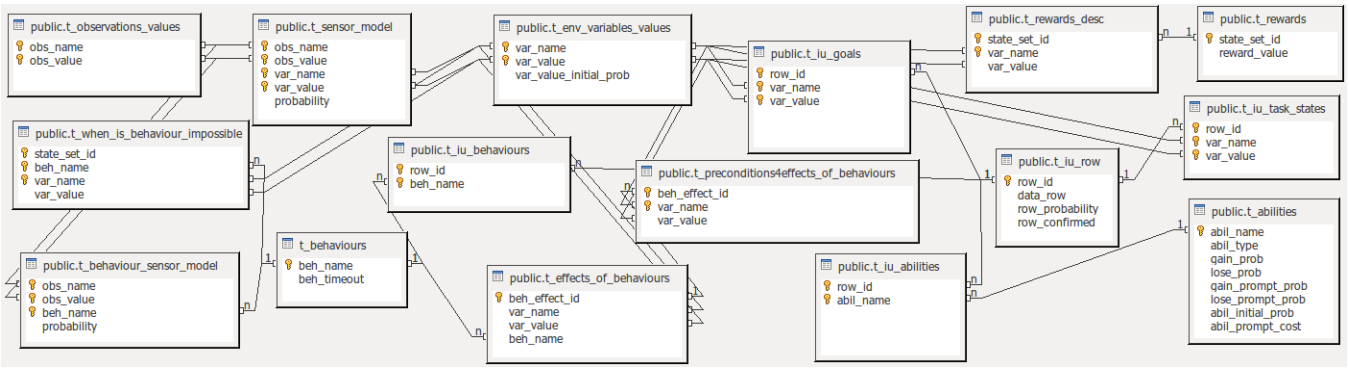

Figure 3: Complete SNAP database diagram.

row, i, and 0 otherwise. We write this as of the row, i, only: $row\_rel(i)$ leaving the remaining variables implicit. The same shorthand is applied to the other functions.

2. $row\_rel\_b : I \times T \times B' \rightarrow \{0,1\}$ is defined as $row\_rel\_b(i,b') = row\_rel(i) \wedge behaviour(i,b')$ where $behaviour(i,b') = 1$ when b' is the behaviour of row i.
3. $row\_abil\_rel : I \times Y' \rightarrow \{0,1\}$ is 1 when all abilities of row i are satisfied by y', and 0 otherwise.
4. $goal : T \rightarrow \{0,1\}$ is 1 when t is a goal state, 0 otherwise.
5. $bn : B' \rightarrow \{0,1\}$ is 1 when $b' = nothing$, 0 otherwise.
6. $same : B \times B' \rightarrow \{0,1\}$ is 1 when $b = b'$. This is a bias which indicates that behaviours are likely to stay the same.
7. $impossible\_beh : T \rightarrow \{0,1\}$ is 0 for states t when there is no behaviour which is possible in t, and 1 when there is at least one behaviour which is possible in t.
8. For all functions defined above, $\neg f(x)$ is defined as $\neg f(x) = 1 - f(x)$ which defines a negation of $f(x)$ when 0 and 1, the domain of $f(x)$, are treated as boolean values.

The above functions are used in the definition of the dynamics of behaviours B', $beh\_dyn : B' \times Y' \times T \times B \rightarrow [0,2]$.

$$beh\_dyn =$$

$$\sum_{i \in I} [row\_abil\_rel(i) \wedge row\_rel\_b(i) \wedge p(i) \vee \quad (1)$$

$$\neg row\_abil\_rel(i) \wedge row\_rel(i) \wedge bn] \vee \quad (2)$$

$$\prod_{i \in I} [\neg row\_rel(i)] \wedge bn \vee \quad (3)$$

$$goal \wedge bn \vee \quad (4)$$

After normalisation, beh_dyn defines probability $P(b'|b,y',t)$ of b' when the previous behaviour was b, the person will have abilities y', and the system is in state t. It is important to recall that non-invasive prompts are assumed here which influence abilities Y' and not behaviours B' directly. The first term (1) of this equation is for rows which have their abilities and state relevance satisfied. (2) defines behaviour 'nothing' when state is relevant in the row but abilities are not present. (3) sets behaviour 'nothing' in all states which are not relevant in any row. (4) reflects the fact that only behaviour 'nothing' happens when the goal state has been reached. It is important to note here, that the IU analysis in original SNAP (Hoey et al. 2011) has to have task states in all rows disjunctive, which means that each state can be relevant in one row only. This is of course not always the case in practice, and we add an extension here which specifies probability $p(i)$ of a row, used in (2), when there are other rows which have the same states relevant.

All the functions necessary to specify the POMDP are represented as algebraic decision diagrams (ADDs) in SPUDD notation (Hoey et al. 1999). These functions are computed with queries on the database. Each such query extracts some subset of ADDs from the relational skeleton. The ADDs are then combined using multiplication and addition to yield the final conditional probability tables (CPTs). Some relations are explicitly represented in the database whereas others need to be extracted using more complex SQL queries. For example, data for $row\_rel(i)$, $row\_rel\_b(i)$, and $row\_abil\_rel(i)$ is read from the IU table in the database. The example subset of the relational skeleton for the IU analysis from Table 1, and diagrams for selected functions are in Figure 4. SQL queries extract compound algebraic decision diagrams from the relational skeleton, and the generator software multiplies those diagrams in order to obtain final functions, such as $P(b'|b,y,t') = row\_rel\_b(i,b')$. The following more complex SQL query example for $goal$ returns sets of states with the highest reward:

```
SELECT var_name, var_value, reward_value,
       t_rewards.state_set_id
FROM t_rewards_desc INNER JOIN t_rewards
ON t_rewards_desc.state_set_id=t_rewards.state_set_id
WHERE reward_value=(SELECT MAX(reward_value)
                    FROM t_rewards)
ORDER BY 4
```

A schematic is shown in Figure 4, where the CPT for client behaviour dynamics is gathered from the relevant tables in the relational skeleton. The original table schema in the PRM for the relations in Figure 4 can be seen in Figure 3.

3.4 Advantages of Database Engines

The advantage of the relational database is that it allows for easy implementation of the constraints required by the model. The simplest example are constraints on attribute values. For example, probabilities have to be in the range [0, 1], or literals should follow specific naming patterns (according to the requirements of the POMDP planner). These simple constraints are easily implemented in the definition of SQL tables. Some more complex constraints which involve more than one attribute are also required. For instance in the planner which we use, sensors and domain attributes are in the

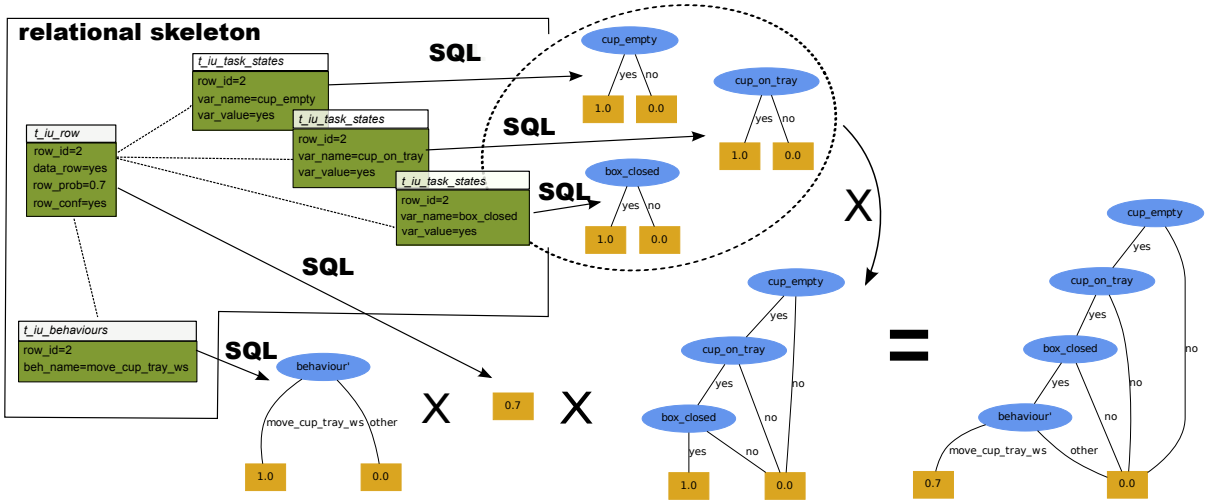

Figure 4: Subset of the relational skeleton for $P(b'|b, y, t')$ showing the generation of $behaviour(i, b') \wedge p(i) \wedge row\_rel(i) = row\_rel\_b(i, b')$ for row $i = 2$ in Table 1. The algebraic decision diagrams (ADDs) representing the relations are multiplied to give the final CPT.

same name space, which means that their names have to be different. Such things can be easily implemented using database triggers, and the user will be prompted at the input time and informed about the constraint.

4 Demonstrative Examples

We demonstrate the method on three assistance tasks: handwashing, and toothbrushing with older adults with dementia, and on a factory assembly task for persons with a developmental disability. We show that once our relational system is designed (i.e. the database and the generator which reads the database and outputs the POMDP file), the system is generic and allows the designer to deploy the system in different tasks simply by populating the database for the new task. The IU analysis for handwashing was performed by a professional IU analyser, and handwashing was used as the testbed for our method. The analysis for the other two tasks were performed by two different biomedical engineers, with limited experience with POMDPs or planning in AI. As an example of the power of our method, the factory task was coded in about six hours by the engineer. This can be compared to a manual coding of the system for handwashing (a smaller task), that took at least three months of work resulting in the system described in (Boger et al. 2006).

The IU analysis breaks an ADL down into a number of sub-tasks, or sub-goals. For the factory task, there are six sub-goals. The decomposition arises according to the elements of recall noted in the IU analysis videos. The six sub-goals are partially ordered, and the partial ordering can be specified as a list of pre-requisites for each sub-goal giving those sub-goals that must be completed prior to the sub-goal in question. Since each sub-goal is implemented as a separate POMDP controller, a mechanism is required to provide hi-level control to switch between sub-goals. We have implemented two such control mechanisms. A deterministic controller is described in (Hoey et al. 2011), and a probabilistic and distributed method in (Hoey and Grześ 2011). All controllers described in the last section are implemented in Java, and run as separate processes and can be easily distributed across several PCs ensuring scalability.

4.1 COACH and prompting

Examples of automatic generation of task policies using IU analyses and a relational database were implemented for the task of handwashing and toothbrushing. For these examples, people with mild to moderate dementia living in a long term care facility were asked to wash their hands and brush their teeth in two separate trials.

Handwashing The washroom used for this task had a sink, pump-style soap dispenser and towel. Participants were led to the sink by a professional caregiver, and were encouraged to independently wash their own hands. The IU analysis was performed on videos captured from a camera mounted above the sink. The task was broken into 5 steps, each comprised of multiple substeps: 1) wet hands; 2) get soap; 3) wash hands; 4) turn off water; and 5) dry hands. Steps 1 and 2 can be completed in any order, followed by step 3. After completion of step 3, steps 4 and 5 can be completed in any order.

Toothbrusing The videos used for the analysis captured participants in a washroom that had a sink, toothbrush, tube of toothpaste and cup, as they tried to independently brush their own teeth. A formal caregiver was present to provide coaching and assistance if required. The IU analysis was completed based on videos of several different people and included multiple different methods of completing the task. The task was divided into 6 main steps, each containing multiple sub-steps: 1) wet brush; 2) apply toothpaste; 3) brush teeth; 4) clean mouth; 5) clean brush; and 6) tidy up. Steps 1 and 2 can be completed in any order, followed by step 3. Steps 4 and 5 can also be completed in any order after step 3, and step 6 is the final step following the first 5 steps.

A policy was generated for each sub-step entered into the relational database. Simulations were run to test the hand-

washing and toothbrushing policies with a user assumed to have mild-to-moderate dementia. To simulate a person with mild-to-moderate dementia the user was forced to forget steps of the task throughout the simulation (i.e., do nothing) but respond to prompting if provided. Tables 2 (handwashing) and 3 (toothbrushing) show a sample of the belief state of the POMDP, the system's suggested action and the actual sensor states for several timesteps of two simulations. Probabilities of the belief state are represented as the height of bars in corresponding columns of each time step. In the handwashing example, the user is prompted to take the towel (t=1). Detecting that the towel was taken (t=2), the system prompts the user to dry his/her hands. The sensors indicate the towel was returned to the surface without drying the user's hands (t=3) so the system again prompts the user to take the towel. As a second example with the toothbrushing simulation (t=1), the system prompts the user to turn on the tap. The user does nothing, so the system tries to prompt the user to take the brush from the cup (in this case either turning the tap on or taking the toothbrush can happen first). The sensors indicate the brush was taken (t=3), so the system returns to prompting the user to turn on the tap.

	Observations		Task				Behaviour					Ability			
Time step, t	hands_water	towel_position	hands_dry	hands_wet	towel_in_hand	towel_on_surface	other	nothing	take_towel	dry_hands	put_down_towel	Af_dry	Af_put_towel_down	Af_take_towel	System Action (Prompt)
0	wet	on_surface													Af_take_towel
1	wet	on_surface													Af_take_towel
2	wet	in_hand													Af_dry
3	wet	on_surface													Af_take_towel
4	wet	in_hand													Af_dry
5	dry	in_hand													Af_put_down_towel
6	dry	on_surface													donothing

Table 2: Example simulation in the handwashing task. The main goal shown in the subtask is to dry the hands after taking the towel from the surface, while the secondary goal is to return the towel to the surface.

	Observations			Task					Behaviour							Ability				
Time step, t	brush_wet	tap	brush_position	brush_wet	brush_in_hand	brush_in_cup	brush_on_surface	tap_on	other	nothing	alter_tap_to_on	take_brush_from_cup	wet_brush	take_brush_from_surface	Rn_brush_in_cup	Rn_brush_on_surface	Af_tap	Af_water	System Action (prompt)
0	dry	off	in_cup																Af_tap
1	dry	off	in_cup																Af_tap
2	dry	off	in_cup																Rn_brush_cup
3	dry	off	in_hand																Af_tap
4	dry	on	in_hand																Af_water
5	dry	on	in_hand																Af_water
6	wet	on	in_hand																donothing

Table 3: Example simulation in the toothbrushing task. The goal in the shown sub-task is to turn on the tap, take the toothbrush from either the surface or the cup, and wet the brush.

4.2 Factory Assembly Task

In this example, workers with a variety of intellectual and developmental disabilities are required to complete an assembly task of a 'Chocolate First Aid Kit'. This task is completed at a workstation that consists of five input slots, an assembly area, and a completed area. The input slot contain all of the items necessary to perform kit assembly-specifically the main first aid kit container (white bin), and four different candy containers that need to be placed into specific locations within the kit container. The IU analysis was completed based on videotapes of a specific adult worker who has Down's Syndrome completing this assembly task. The worker was assessed with a moderate-to-mild cognitive impairment and was able to follow simple instructions from a job coach. The IU analysis broke this task into 6 required steps: 1) prepare white bin; 2) place in bin chocolate bottle 1; 3) place in bin chocolate box 1; 4) place in bin chocolate box 2; 5) place in bin chocolate bottle 2; and 6) finish output bin and place in completed area. Steps 2, 3, 4, and 5 can be completed in any order. Through a hierarchical task analysis (Stammers and Sheppard 1991) each of these steps were further broken down into sub-steps.

Policies were generated for each of the required assembly steps and were simulated by the authors for three different types of clients: mild, moderate, and severe cognitive impairment. Figure 5 is the output of sample timestamps for step 2 for a user with severe cognitive impairment. Again, probabilities of the belief state are represented as the height of bars in corresponding columns of each time step. In this specific example, the system is more active in its prompting based on the fact that the user is assumed to have diminished abilities with respect to the different aspects that needs to be completed. For example (t=1), the worker has deteriorating ability to recognize that the slot that holds the required chocolate bottle is empty. As such, the system correctly prompts the worker to recognize that the slot is empty and needs to be filled. In another example (t=5), the system recognizes that the worker has not placed the bottle in its correct location in the white bin, and provides a prompt for the person to recall that the bottle needs to be in that position in order to reach the final goal state. When the worker does not respond to this prompt, the system decides (t=6) to play a different, more detailed, prompt (a prompt related to the affordance ability).

5 Conclusions and discussion

POMDP models have proven to be powerful for modelling intelligent human assistance (Hoey et al. 2010). Unfortunately, POMDPs, being propositional models, usually require a very labour intensive, manual setup procedure. A standard approach which can make AI models portable and configurable is to introduce the notion of objects and relations between them and this is found in relational methods such as Statistical Relational Learning. In this paper, we derive a methodology for specifying POMDPs for intelligent human assistance which is motivated by relational modelling in frame-based SRL methods (Getoor and Taskar 2007). The core of our approach is the relational database which the user populates in order to prepare the deployment of the system for a specific task. The database and the POMDP generator have a structure that is designed based on experience in the domain of assistance. Content provided by the designer of a particular implementation then encodes the goals, action preconditions, environment states, cognitive model, user and system actions, as well as relevant sensor models, and automatically generates a valid POMDP model of the assistance task being modelled. The strength of the database is

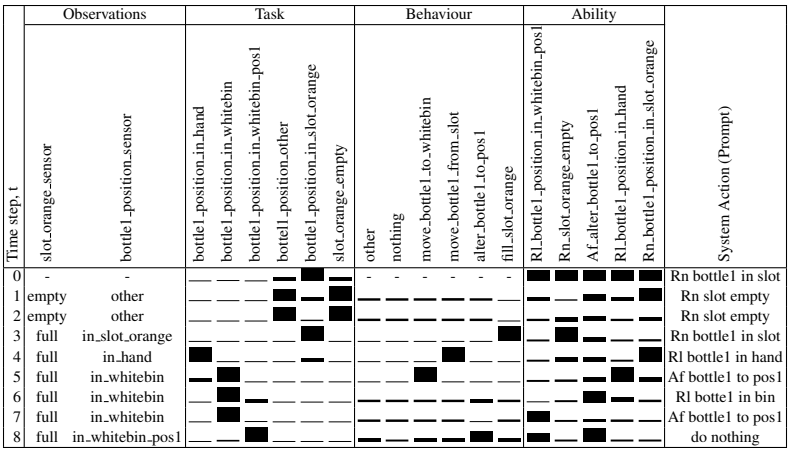

Figure 5: Example simulation in the factory assembly task. The goal in the shown sub-task is to take the bottle, named bottle 1, from the orange slot and to place the bottle in the white bin in pos1.

that it allows constraints to be specified, such that we can verify the POMDP model is, indeed, valid for the task. We demonstrate the method on three assistance tasks: handwashing and toothbrushing for elderly persons with dementia, and on a factory assembly task for persons with a cognitive disability. This demonstration shows that the system, once designed using the relational approach, can be instantiated to create a POMDP controller for an arbitrary intelligent human assistance task. The use of the relational database makes the process of specifying POMDP planning tasks straightforward and accessible to standard computer users.

6 Acknowledgements

This research was sponsored by American Alzheimers Association grant numbers ETAC-08-89008 and ETAC-07-58793.

References

Åström, K. J. 1965. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications* 10:174–205.

Boger, J.; Hoey, J.; Poupart, P.; Boutilier, C.; Fernie, G.; and Mihailidis, A. 2006. A planning system based on Markov decision processes to guide people with dementia through activities of daily living. *IEEE Transactions on Information Technology in Biomedicine* 10(2):323–333.

Burns, A., and Rabins, P. 2000. Carer burden in dementia. *International Journal of Geriatric Psychiatry* 15(S1):S9–S13.

Chen, L.; Nugent, C. D.; Mulvenna, M.; Finlay, D.; Hong, X.; and Poland, M. 2008. A logical framework for behaviour reasoning and assistance in a smart home. *International Journal of Assistive Robotics and Mechatronics* 9(4):20–34.

Duke, D.; Barnard, P.; Duce, D.; and May, J. 1998. Syndetic modelling. *Human-Computer Interaction* 13(4):337.

Getoor, L., and Taskar, B., eds. 2007. *Statistical Relational Learning*. MIT Press.

Gill, T., and Kurland, B. 2003. The burden and patterns of disability in activities of daily living among community-living older persons. *Journal of Gerontology Series A: Biological Sciences and Medical Sciences* 58A(1):M70–M75.

Hoey, J., and Grześ, M. 2011. Distributed control of situated assistance in large domains with many tasks. In *Proc. of ICAPS*.

Hoey, J.; St-Aubin, R.; Hu, A.; and Boutilier, C. 1999. SPUDD: Stochastic planning using decision diagrams. In *Proceedings of Uncertainty in Artificial Intelligence*, 279–288.

Hoey, J.; Poupart, P.; Boutilier, C.; and Mihailidis, A. 2005. POMDP models for assistive technology. In *Proc. AAAI Fall Symposium on Caring Machines: AI in Eldercare*.

Hoey, J.; Poupart, P.; von Bertoldi, A.; Craig, T.; Boutilier, C.; and Mihailidis, A. 2010. Automated handwashing assistance for persons with dementia using video and a partially observable markov decision process. *Computer Vision and Image Understanding* 114(5):503–519.

Hoey, J.; Plötz, T.; Jackson, D.; Monk, A.; Pham, C.; and Olivier, P. 2011. Rapid specification and automated generation of prompting systems to assist people with dementia. *To Appear in Pervasive and Mobile Computing*. doi:10.1016/j.pmcj.2010.11.007.

Kirwan, B., and Ainsworth, L. 1992. *The task analysis guide*. London: Taylor and Francis.

Mastrogiovanni, F.; Sgorbissa, A.; and Zaccaria, R. 2008. An integrated approach to context specification and recognition in smart homes. In *Smart Homes and Health Telematics*, 26–33. Springer.

McCluskey, L. 2000. Knowledge engineering for planning roadmap. Working Paper.

Pentney, W.; Philipose, M.; and Bilmes, J. 2008. Structure learning on large scale common sense statistical models of human state. In *Proc. AAAI*.

Pham, C., and Olivier, P. 2009. Slice&dice: Recognizing food preparation activities using embedded accelerometers. In *European Conference on Ambient Intelligence*, 34–43. Berlin: Salzburg, Austria: Springer-Verlag.

Ryu, H., and Monk, A. F. 2009. Interaction unit analysis: A new interaction design framework. *Human-Computer Interaction* 24(4):367–407.

Salber, D.; Dey, A.; and Abowd, G. 1999. The context toolkit: Aiding the development of context-enabled applications. In *Proc. of the Conference on Human Factors in Computing Systems (CHI)*, 434–441.

Stammers, R., and Sheppard, A. 1991. *Evaluation of Human Work*. Taylor & Francis, 2nd edition. chapter Chapter 6: Task Analysis.

Wherton, J. P., and Monk, A. F. 2009. Problems people with dementia have with kitchen tasks: the challenge for pervasive computing. *Interacting with Computers* 22(4):253–266.