Corpus-Based Incremental Intention Recognition via Bayesian Network Model Construction

Han The Anh [*] and **Luís Moniz Pereira**
Centro de Inteligência Artificial (CENTRIA)
Departamento de Informática, Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
$h.anh@fct.unl.pt, lmp@di.fct.unl.pt$

Abstract

We present a method for incremental intention recognition by means of incrementally constructing a Bayesian Network (BN) model as more actions are observed. It is achieved based on a knowledge base of easily maintained and constructed fragments of BNs, connecting intentions to actions. The simple structure of the fragments enables to easily and efficiently acquire the knowledge base, either from domain experts or automatically from a plan corpus. We show experimental results improvement for the Linux Plan Corpus. In addition, we create a new, so-called IPD Plan Corpus, for strategies in the iterated Prisoner's Dilemma and show the experimental results for it.

1. Introduction

We propose a method for intention recognition in a dynamic, real-world environment. An important aspect of intentions is their pointing to the future, i.e. if we intend something now, we mean to execute a course of actions to achieve something in the future (Bratman 1987). Most actions may be executed only at a far distance in time. During that period, the world is changing, and the initial intention may be changed to a more appropriate one or even abandoned (Bratman 1992). An intention recognition method should take into account these changes, and may need to reevaluate the intention recognition model depending on some time limit; in addition, as a new action is observed, the model should be reconfigurable to incorporate new observed actions.

Generally, *intention recognition* (also called *goal recognition*) is defined as the process of becoming aware of the intention of another agent and, more technically, as the problem of inferring an agent's intention through its actions and their effects on the environment (Heinze 2003). *Plan recognition* is closely related to intention recognition, extending it to also recognize the plan the observed agent is following in order to achieve his intention (Sadri 2010).

Intention recognition is performed in domains in which it is preferable to have a fast detection of just the user goal or intention than a more precise but time consuming detection of the complete user plan, e.g. in the interface agents domain (Horvitz et al. 1998). Generally, the input of both intention and plan recognition systems are a set of conceivable intentions and a set of plans achieving each intention (plan library or plan corpus). Intention recognition is distinct from planning, as goals are not known a priori, and presumed goals are subject to defeasibility. There are also generative approaches based on planning algorithms, which do not require plan library/corpus (e.g., see (Ramírez and Geffner 2010)).

In this work, we use Bayesian Networks (BN) as the intention recognition model. The flexibility of BNs for representing probabilistic dependencies and the efficiency of inference methods for BN have made them an extremely powerful and natural tool for problem solving under uncertainty (Pearl 1988; Pearl 2000).

This paper presents a knowledge representation method to support incremental BN construction for performing intention recognition during runtime, from an initially given domain knowledge base. As more actions are observed, a new BN is constructed reinforcing some intentions while ruling out others. This incremental method allows domain experts to specify knowledge in terms of small and simple BN fragments, which can be easily maintained and changed. Alternatively, these fragments can be easily learned from data.

It is inspired by the fact that knowledge experts often consider a related set of variables together, and organize domain knowledge in larger chunks. An ability to represent conceptually meaningful groupings of variables and their interrelationships facilitates both knowledge elicitation and knowledge base maintenance (Natarajan et al. 2008; Laskey 2008). To this end, there have been several methods proposed for Bayesian network construction from small and easily maintained network fragments (Pearl 1988; Natarajan et al. 2008; Laskey 2008). Basically, a combination of BNs is a graph that includes all nodes and links of the networks, where nodes with the same name are combined into a common node. The main issue for a combination method is how the influence of different parents of the common node can be combined in the new network, given the partial influence of each parent in the corresponding fragment. The most extensively used and popular combination method is Noisy-Or, firstly proposed by (Pearl 1988) for Bayesian networks of Boolean variables, and generalized by (Srinivas 1993) for the general case of arbitrary domains. A set of conditions is needed to be satisfied for the Noisy-OR method to work

[*]HTA is supported by FCT Portugal (reference SFRH/BD/62373/2009).

properly. We will discuss in more detail in the third section where the method is applied in our work (Def.5).

In the next section we recall some background about BN. Then a general method for incremental BN model construction during runtime is presented. We next address a special well-known case and present experimental results for it, comparing with other systems.

2. Bayesian Networks

Definition 1 *A Bayes Network is a pair consisting of a directed acyclic graph (DAG) whose nodes represent variables and missing edges encode conditional independencies between the variables, and an associated probability distribution satisfying the Markov assumption of conditional independence, saying that variables are independent of non-descendants given their parents in the graph (Pearl 2000).*

In a BN, associated with each node of its DAG is a specification of the distribution of its variable, say A, conditioned on its parents in the graph (denoted by $pa(A)$)—i.e., $P(A|pa(A))$ is specified. If $pa(A) = \emptyset$ (A is called root node), its unconditional probability distribution, $P(A)$, is specified. These distributions are called Conditional Probability Distribution (CPD) of the BN.

The joint distribution of all node values can be determined as the product of conditional probabilities of the value of each node on its parents $P(X_1, ..., X_N) = \prod_{i=1}^{N} P(X_i|pa(X_i))$, where $V = \{X_i|1 \leq i \leq N\}$ is the set of nodes of the DAG.

Suppose there is a set of evidence nodes (i.e. their values are observed) in the DAG, say $O = \{O_1, ..., O_m\} \subset V$. We can determine the conditional probability distribution of a variable X given the observed value of evidence nodes by using the conditional probability formula

$$P(X|O) = \frac{P(X,O)}{P(O)} = \frac{P(X, O_1, ..., O_m)}{P(O_1, ..., O_m)} \tag{1}$$

where the numerator and denominator are computed by summing up the joint probabilities over all absent variables with respect to V.

3. Incremental Bayesian Network Construction for Intention Recognition

In (Pereira and Han 2009; Pereira and Han 2010), a general BN model for intention recognition is presented and justified based on Heinze's intentional model (Heinze 2003). Basically, the BN consists of three layers: cause/reason nodes in the first layer, connecting to intention nodes in the second one, in turn connecting to action nodes in the third. The interested readers are referred to those papers for more details. Han and Pereira (2010a) then presented a method for incrementally constructing such BN model for performing *incremental* intention recognition, including all the three layers.

A BN model for intention recognition consists of two layers: the layer of intentions and the layer of actions.

Definition 2 (Intention Recognition BN – IRBN) *A BN for intention recognition (IRBN) W is a triple* $\langle \{Is, As\}, pa, P_W \rangle$ *where*

- Is *and* As *are the sets of intention nodes and action nodes, respectively. They stand for binary random variables.*
- pa *is a mapping which maps a node to the set of its parent nodes such that:* $\emptyset \neq pa(A) \subseteq Is \;\; \forall A \in As$, *and* $pa(I) = \emptyset \;\; \forall I \in Is$. *This means there is no isolated action node and intentions are represented by top nodes.*
- *CPD tables of the action nodes and prior probabilities of the intention nodes are given by the probability distribution* P_W, *i.e.* $P_W(X|pa(X))$ *defines the probability of* X *conditional on* $pa(X)$ *in W, for all* $X \in V_W$ *where* $V_W = Is \cup As$.

The intention recognition method will be performed by incrementally constructing an IRBN as more actions are observed. The construction is based on a prior knowledge base of Unit BN Fragments consisting of a single intention connecting to a single action. We refer to them as the Unit Fragment (of BN) for intention recognition.

Definition 3 (Unit Fragment) *A Unit Fragment of BN for intention recognition consists of an intention* I *connecting to (i.e. causally affecting) an action* A, *and is denoted by* $UF(I, A)$. *Both nodes stand for binary random variables.*

Definition 4 (Knowledge Base) *A domain knowledge base KB consists of a set unit fragments.*

An intention I has the same *fixed* prior probability distribution in all the unit fragments it belongs to, denoted by $P_{KB}(I)$. The prior probability distribution of the top (intention) nodes also can be made situation-dependent by adding a pre-intentional layer of cause/reason nodes as in (Han and Pereira 2010a)—which would enable it to deal with and explain some important issues in intention/plan recognition such as intention change/abandonment (Geib and Goldman 2003). However, since the dataset we use later for evaluation has no such information available, we omit that layer to simplify the presentation.

The simple structure of unit fragments enables domain experts to easily construct and maintain the knowledge base. The BN fragments also can be learned from appropriate datasets, as we shall see later with the Linux plan corpus.

Before presenting the intention recognition algorithm, let us define some operators for handling CPD tables and IRBNs.

3.1 Operators for Constructing IRBN

As a new action A is observed, we need to incorporate it into the current IRBN. Firstly, the appropriate unit fragments for A are selected from the domain knowledge base. Han and Pereira (2010a) proposed some methods for selecting the appropriate fragments in a situation-sensitive manner. They are based on the intuition that whether an intention may give rise to an action depends on the situation in which the action is observed. That enables to reduce the size of the BN model, which otherwise could be very large.

We are not going to elaborate further on these methods here, and assume that the operator *select(A,SIT)* provides the (context-dependent) set of unit fragments for action A given the situation at hand SIT. If SIT is empty, *select(A,SIT)* is

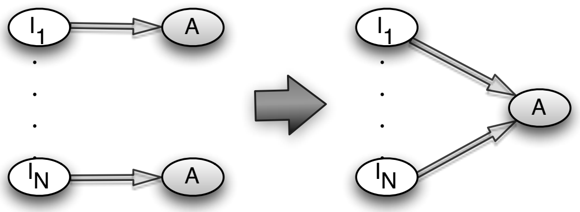


Figure 1: Noisy-OR Combination Method

the set of all unit fragments for action A from the knowledge base.

Secondly, after having obtained the appropriate fragments, we combine them using the Noisy-OR method (Pearl 1988; Srinivas 1993; Cozman 2004) and obtain an IRBN with a single action (Figure 1). It is called an Unit IRBN for action A in situation SIT, and denoted by *irBN(A,SIT)*.

Definition 5 (Unit IRBN via Noisy-OR) *The Unit IRBN for action* A *in a given situation* SIT*, $irBN(A, SIT)$, is obtained via* Noisy-OR *method as follows.*

Let $select(A, SIT) = \{UF(I_1, A), ..., UF(I_N, A)\}$ and for $1 \leq i \leq N$, $P(A = T|I_i = T) = p_i$ (defined in fragment $UF(I_i, A)$). Then, $\mathtt{irBN(A,SIT)}$ is the result of combining $UF(I_1, A), ..., UF(I_N, A)$ using Noisy-OR method, i.e. $p(A = T|I_1, ..., I_N) = 1 - \prod_{i:I_i=T}(1 - p_i)$. Note that the prior probability distribution of I_i, $1 \leq i \leq N$, in the new combined BN is the same as in its original fragment.

The rationale and appropriateness of the application of the Noisy-OR method here for combining unit fragments is based on the intuition that each intention I_i, $1 \leq i \leq N$, can be interpreted as a "*cause*" of action A; and action A occurs when one or more of the intentions are active. Detailed arguments for this can be found in (Cozman 2004).

Definition 6 (Project of CPD Table) *Let T be a CPD table defining $P(X|V)$, the probability of a random variable X conditional on a set of random* binary *variables V, and $V' \subsetneq V$. The project of T on V', denoted by $\mathtt{proj(T,V')}$, is the part of T corresponding to all variables in $V \setminus V'$ being false.*

Now we need to combine the obtained unit IRBN, $\mathtt{irBN(A,SIT)}$, with the current IRBN. For that, in the sequel we define how to combine two IRBNs. Intuitively, we simply add up all the new nodes and links of the new IRBN to the current IRBN, keeping the CPD tables from the original IRBNs.

Definition 7 (Combination of IRBNs) *Let $W_1 = \langle\{Is_1, As_1\}, pa_1, P_1\rangle$ and $W_2 = \langle\{Is_2, As_2\}, pa_2, P_2\rangle$ be two IRBNs. The combination of these two IRBNs is an IRBN, denoted by $\mathtt{comb(W_1,W_2)} = \langle\{Is, As\}, pa, P_W\rangle$, defined as follows*

- $As = As_1 \cup As_2$; $Is = Is_1 \cup Is_2$;
- $pa(I) = \emptyset\ \forall I \in Is$; $pa(A) = pa_1(A) \cup pa_2(A)$;
- $P_W(I) = P_{KB}(I)\ \ \forall I \in Is$; *and for each* $A \in As$, $P_W(A|pa(A)) = P_{W_1}(A|pa(A))$ *if* $A \in As_1$ *and* $P_W(A|pa(A)) = P_{W_2}(A|pa(A))$ *if* $A \in As_2$.

Note that here it is allowed the possibility that the observed agent follows multiple intentions simultaneously. In (Han and Pereira 2010a) the authors dealt with the case of a single intention being pursued, where in the combined IRBN only the intersection (instead of union) of two intention sets, $Is_1 \cap Is_1$, is retained; which enables to reduce the size of the IRBN model.

When some intentions are found irrelevant—e.g. because they are much unlikely[1]—those intentions should be remove from the IRBN. This is enacted by considering them as completely false and employing a project operator.

Definition 8 (Remove Intentions from IRBN) *Let $W = \langle\{Is, As\}, pa, P_W\rangle$ be an IRBN and $R \subset Is$ a strict subset of Is. The result of removing the set of intentions R from W is an IRBN, denoted by $\mathtt{remove(W,R)} = \langle\{Is_R, As_R\}, pa_R, P_R\rangle$, and defined as follows*

- $As_R = \{A \in As \mid pa_R(A) \neq \emptyset\}$; $Is_R = Is \setminus R$;
- $pa_R(I) = \emptyset\ \forall I \in Is_R$; $pa_R(A) = pa(A) \setminus R$;
- $P_W(I) = P_{KB}(I)\ \ \forall I \in Is$; *and for each* $A \in As_R$, $P_R(A|pa_R(A))$ *is defined by the CPD table* $proj(T, Is_R)$ *where T is the CPD table for A in W, i.e. defined by* $P_W(A|pa(A))$.

Based on these operators, we now can describe an algorithm for incremental intention recognition in a real-time manner.

Incremental Intention Recognition Algorithm. Repeat the following step until some given time limit is reached; the most likely intention in previous cycle is the final result.

- Let A be a new observed action and SIT the current situation. Combine the current IRBN W with $\mathtt{irBN(A,SIT)}$ we obtain $\mathtt{W'} = \mathtt{comb(W,irBN(A,SIT))}$. If A is the initially observed action, let $\mathtt{W'} = \mathtt{irBN(A,SIT)}$.
- Compute the probability of each intention in W', conditional on the set of current observed actions in W'. Remove the intentions which are much less likely than the others (following Definition 8).

Note that if an observed action is not in the knowledge base, the action is considered irrelevant to the sought for intention, and discarded. Furthermore, at any cycle, if the likelihood of all the intentions are very small (say, smaller than a given threshold), one could say that the sought for intention is abandoned. This is because the causes and actions do not support or force the intending agent to keep pursuing his initial intention anymore.

4. Relations Amongst Intention Nodes

When considering the case in which the observed agent may pursue multiple intentions simultaneously, it is undoubtedly indispensable to take into account and express the relations amongst the intentions in the model.

Pursuing one intention may exclude the other intention to be pursued. It may be so because of resource limitation, e.g.

[1]One intention is much less likely than the other if the fraction of its likelihood and that of the most likely intention is less than some small threshold. It is up to the KB designer to provide it.

allowance time is not enough for accomplishing both intentions at the same time. It also may be because of the nature or restriction of the observed agent's task: the agent is restricted to pursuing a single intention (e.g. in constructing Linux plan corpus, a user is given one task at a time to complete. We shall discuss this case in more detail in the next sections).

We introduce a so-called *exclusive relation e*—a binary relation on the set of intention nodes—representing that if one intention is pursued, then the other intention cannot be pursued. It is usually, although perhaps not always, the case that intentions exclusiveness is symmetric. It holds for the resource limitation case: one intention excludes the other intention because there is not enough resource for accomplishing both, which in turn implies that the latter intention also excludes the former one. It also clearly holds for the case where the agent is restricted to pursuing a single intention. In this paper, we assume that the exclusive relation on intentions e is symmetric; it can be renamed *mutually exclusive relation*.

Intentions I_1 and I_2 are mutually exclusive iff they cannot be pursued simultaneously, i.e. $P(I_1 = T, I_2 = T) = 0$. Thus, for any action A, if $I_1, I_2 \in pa(A)$ then the CPD table for A is undefined. Hence, the BN needs to be restructured. The mutually exclusive intentions must be combined into a single node since they cannot co-exist as parents of a node. Each intention represents a possible value of the new combined node. Namely, let $I_1, ..., I_t$ be such that $e(I_i, I_j),\ \forall i, j : 1 \le i < j \le t$. The new combined node, I, stands for a random variable whose possible outcomes are either $I_i,\ 1 \le i \le t$, or $\tilde{I}$—the outcome corresponding to the state that none of $I_i = T$. Note that if the intentions $I_1, ..., I_t$ are exhaustive, $\tilde{I}$ can be omitted. Next, I is linked to all the action nodes that has a link from one of $I_i,\ 1 \le i \le t$.

It remains to re-define CPD tables in the new BN. They are kept the same for action A where $I \notin pa(A)$. For A such that $I \in pa(A)$, the new CPD table at $I = I_i$ corresponds to the CPD table in the original BN at $I_i = T$ and $I_j = F\ \forall j \neq i, 1 \le j \le t$, i.e. $P(A|I = I_i, ...) = P(A|I_0 = F, ..., I_{i-1} = F, \mathbf{I_i} = \mathbf{T}, I_{i+1} = F, ..., I_t = F, ...)$. Note that the left hand side is defined in the new BN, and the right hand side is defined in the original BN. Similarly, the new CPD table at $I = \tilde{I}$ corresponds to $I_i = F\ \forall 1 \le i \le t$. In addition, prior probability $P(I = I_i) = P(I_i = T)$ and $P(I = \tilde{I}) = \prod_{i=1}^{t} P(I_i = F)$ (and then being normalized).

In the next section we will look at a special case where the observed agent pursues a single intention. Thus, all intentions are mutually exclusive, and they can be combined into a single node. We then evaluate the method using the Linux Plan Corpus. After that, in Section 6., we present a new plan corpus, called IPD, and also present experimental results for it.

5. Single Intention Being Pursued

5.1 The Model

Suppose the observed agent pursues a single intention. In this case, all intentions are mutually exclusive, and they can be combined into a single node. The IRBN then consists of a single intention node, linking to all action nodes.

Let $I_1, ..., I_n$ be the intentions in the original IRBN. As usual, they are assumed to be exhaustive, i.e. the observed agent is assigned an intention from them. The combined node I thus has n possible outcomes $I_i,\ 1 \le i \le n$. Let $A_1, ..., A_m$ be the current observed actions. Applying Equation 1, we easily obtain the probability of each intention conditional on the current observed actions as follows, for $1 \le j \le n$,

$$P(I = I_j|A_1, ..., A_m) = \frac{P(I_j)\prod_{i=1}^{m} P(A_i|I_j)}{\sum_{j=1}^{n} P(I_j)\prod_{i=1}^{m} P(A_i|I_j)}$$

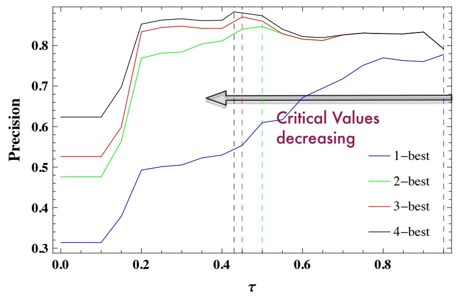
This implies our intention recognizer has a linear complexity $O(|Is|)$, where Is is the set of intentions being modeled.

5.2 Experimental Evaluation

The Linux Plan Corpus Plan corpus is the term used to describe a set of plan sessions and consists of a list of goals/intentions and the actions a user executed to achieve them (Armentano and Amandi 2009). Although there are many corpora available for testing machine learning algorithms in other domains, just a few are available for training and testing plan/intention recognizers; furthermore, each of the plan/intention recognizers using plan corpora usually has its own datasets—which leads to a difficult comparison amongst each other. For that important reason, we chose Linux plan corpus (Blaylock and Allen 2004)—one of the rare regularly used plan corpora—which was kindly made publicly available by Nate Blaylock—in order to test our system. It also enables a better comparison with other systems using this corpus (Blaylock and Allen 2005; Blaylock and Allen 2004; Armentano and Amandi 2009).

The Linux plan corpus is modeled after Lesh's Unix plan corpus (Lesh 1998). It was gathered from 56 human users (graduate and undergraduate students, faculty, and staff) from the University of Rochester Department of Computer Science. The users have different levels of expertise in the use of Linux, and they were allowed to perform as many times as they wished, in order to contribute more plan sessions. The sessions, consisting in sequences of commands performed by the users to achieve a given goal/intention, were automatically recorded. At the end of each session, the users were asked to indicate whether they succeeded in achieving their goal/intention. In total, there are 547 sessions, 457 of which were indicated as successfully completing the goal, 19 goal schemas and 43 action schemas. More details can be found in (Blaylock and Allen 2004) or (Linux-Plan-Corpus).

Learning Unit Fragments from Data For unit fragment $UF(I, A)$, the conditional probability of A given I is defined by the frequency of A in a plan session for achieving the goal/intention I divided by the frequency of any action for achieving I: $P(A = T|I = T) = \frac{freq(A_I)}{freq(I)}$. For better understanding, in the plan corpus each action is marked with the intention which the action is aiming at. Then, $freq(A_I)$ is the frequency of A being marked by I, and $freq(I)$ is the frequency of seeing the mark I.



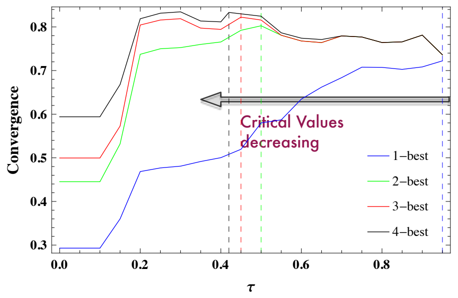


Figure 2: Precision and convergence for $\tau \in [0, 1]$ and for different values of N (N = 1, 2, 3, 4) on Linux Plan corpus.

Note that prior probabilities of all the intentions in the corpus are given initially, and used for generating tasks for users (Linux-Plan-Corpus ; Blaylock and Allen 2004).

Making Predictions Similar to (Blaylock and Allen 2004), instead of letting the recognizer make a prediction after each observed action, we set a *confidence* threshold τ $(0 \le \tau \le 1)$, which allows the recognizer to decide whether or not it is confident enough to make a prediction; the recognizer only makes a prediction if the likelihood of the most likely intention in the model is greater than τ. Otherwise, it predicts "don't know". In addition, instead of only predicting the most likely intention, the recognizer provides a set of N most likely ones (*N-best prediction*).

Evaluation Metrics For evaluating our system and comparing with the previous ones (Blaylock and Allen 2004; Armentano and Amandi 2009), we use three different metrics. *Precision* and *recall* report the number of correct predictions divided by total predictions and total prediction opportunities, respectively. More formally (also see (Armentano and Amandi 2009)), let $Seq = a_1, ..., a_n$ be a sequence of actions (plan session) achieving intention I. Considering N-best prediction case, let $correct(A) = 1$ if I is one of N most likely intentions, and 0 otherwise. Then, precision and recall for *Seq* are defined as: $precision(Seq) = (\sum_{i=1}^{n} correct(a_i))/z$; $recall(Seq) = (\sum_{i=1}^{n} correct(a_i))/Z$, where z and Z are the number of predictions made (when the recognizer is confident enough) and the total number of prediction opportunities (i.e. when $\tau = 0$), respectively.

On the other hand, *convergence* is a metric that indicates how much time the recognizer took to converge on what the current user goal/intention was. Let t be such that $correct_i = 0$ for $0 \le i \le t-1$ and 1 for $t \le i \le n$ (i.e. t is the first time point which from there on the system always correctly predicts), *convergence* for *Seq* is defined as: $convergence(Seq) = (z - t + 1)/z$.

Finally, the *overall* precision, recall and convergence are obtained by taking averages over all testing sessions.

Experiments and Results Because of the small size of the Linux corpus, similar to previous works, we ran experiments using the one-out cross validation method (Armentano and Amandi 2009). Just one at a time, one plan session in the whole corpus is left out. The rest of the corpus is used for training the model, which is then evaluated against the left out plan session. We study the effect of confidence level τ w.r.t. precision and convergence (for recall, it clearly is a decreasing function of τ) (Figure 2). The greater N, the better precision and convergence. The difference in precision and convergence between two different values of N is large when τ is small, and gets smaller for greater τ. Most interestingly, we observe that precision and convergence are not increasingly monotonic on τ. There are *critical values* of τ at which the measures have maximal value, and those values are smaller for greater N. This observation suggests that in plan/intention recognition task, the more precise (i.e. the smaller N) the decision is needed to make, the greater confidence level the recognizer should gain to make a good (enough) decision. On the other hand, the recognizer should not be too cautious, leading to refuse to make a prediction when it would have been able to make a correct one. In short, this experimentation suggests an important need to study (experimentally) the confidence threshold τ carefully for particular application domains, and for particular values of N. Using the same τ for all values of N could decrease the recognizer's performance.

Table 1: Intention Recognition Results on the Linux Plan Corpus

N-best	1-best	2-best	3-best	4-best
τ	0.95	0.5	0.45	0.42
Precision	0.786	0.847	0.870	0.883
Recall	0.308	0.469	0.518	0.612
Converg.	0.722	0.799	0.822	0.824

Table 1 shows some of the results for different values of N (and the corresponding value of τ). Similar to the previous works on the same Linux corpus (Blaylock and Allen 2004; Armentano and Amandi 2009), we keep the best results of each case w.r.t. τ for the comparison. For example, we obtained a precision of 78.6% for 1-best that is increased to 87.0% for 3-best prediction and 88.3% for 4-best one. Convergence is increased from 72.2% for 1-best to 82.2% for 3-best and 82.4% 4-best prediction.

The best performance on the Linux corpus (namely, in terms of precision and convergence) so far was reported in (Armentano and Amandi 2009), where the authors use variable Markov model with exponential moving average. Here we got an increment of 14% better precision and 13.3% better convergence for 1-best prediction, 8.2% better precision and 9.3% better convergence for 2-best prediction, and 7.5% better precision and 7.7% better convergence for 3-

best prediction. We also obtained better recalls comparing with (Blaylock and Allen 2004) in all cases.

Note that in (Armentano and Amandi 2009), the authors use a more fine-grained preprocessing method for their work, but we suspect it would have increased their performance. To fairly compare with both works, we use the original corpus.

6. IPD Plan Corpus

We present a new plan corpus in the context of Iterated Prisoner's Dilemma (IPD)[2] and show the experimental results for it. The intentions/goals to be recognized are the (known) strategies in IPD (see below) and plan sessions are the sequences of moves these strategies play with other players.

6.1 Iterated Prisoner's Dilemma

Prisoner's Dilemma is a symmetric two-player non-zero game defined by the payoff matrix (for row player)

$$\begin{array}{c c} & \begin{array}{cc} C & D \end{array} \\ \begin{array}{c} C \\ D \end{array} & \begin{pmatrix} R & S \\ T & P \end{pmatrix} \end{array}$$

Each player have two options in each round, cooperates (C) or defects (D). A player who chooses to cooperate with someone who defects receives the sucker's payoff S, whereas the defecting player gains the temptation to defect, T. Mutual cooperation (resp., defection) yields the reward R (resp., punishment P) for both players. In PD, it satisfies that $T > R > P > S$. Thus, in a single round, it is always best to defect, but cooperation may be rewarded if the game is iterated. In IPD, it is also required that mutual cooperation is preferred over an equal probability of unilateral cooperation and defection ($2R > T + S$); otherwise alternating between cooperation and defection would lead to a higher payoff than mutual cooperation.

IPD is usually known as a story of tit-for-tat (TFT), which won both Axelrod's tournaments (Axelrod 1984). *TFT* starts by cooperating, and does whatever the opponent did in the previous round. It will cooperate if the opponent cooperated, and will defect if the opponent defected. But if there are erroneous moves (i.e. an intended move is wrongly performed with a given execution error), the performance of *TFT* declines: it cannot correct errors or mistakes. Tit-for-tat is then replaced by generous tit-for-tat (GTFT), a strategy that cooperates if the opponent cooperated in the previous round, but sometimes cooperates even if the opponent defected (with a fixed "forgiveness" probability $p > 0$) (Sigmund 2010). *GTFT* can correct mistakes.

Subsequently, *TFT* and *GTFT* were replaced by win-stay-lose-shift (WSLS) as the winning strategy chosen by evolution (Sigmund 2010). *WSLS* repeats the previous move whenever it did well, but changes otherwise.

Some other less famous strategies (which we are going to use later) are GRIM – a grim version of TFT, prescribing to defect except after a round of mutual cooperation, and Firm-But-Fair (FBF) – known as a tolerant brother of TFT, prescribing to defect only if getting a sucker's payoff S in previous round. Details of all strategies considered here can be found in (Sigmund 2010) (Chapter 3).

[2]It also applies for other famous social dilemmas such as Snow Drift and Stag Hunt (Sigmund 2010).

Next, we describe how training and testing plan corpora are created employing these strategies. Abusing notations, R, S, T and P are also referred to as game states (in a single round or interaction). We too use E (standing for *empty*) to refer to the game state having had no interaction.

6.2 IPD Plan Corpus Description

We made an assumption that all strategies to be recognized have the memory size bounded-up by M ($M \geq 0$)—i.e. their decision at the current round is independent of the past rounds that are at a time distance greater than M. The strategies described above have memory $M = 1$.

An action in the corpus is of the form $s_1 ... s_M \xi$, where $s_i \in \{E, R, T, S, P\}$, $1 \leq i \leq M$, are the states of the M last interactions, and $\xi \in \{C, D\}$ is the current move. We denote by Σ_M the set of all possible types of action. E.g, $\Sigma_1 = \{EC, RC, TC, SC, PC, ED, RD, TD, SD, PD\}$. This encoding method enables to save the game states without having to save the co-player's moves, thus simplifying the corpus representation, described below.

Suppose we have a set of strategies to be recognized. The plan corpus for this set consists of a set of plan sessions generated for each strategy in the set. A plan session of a strategy is a sequence of actions played by that strategy (more precisely, a player using that strategy) against an arbitrary player. As an example, let us consider *TFT* and the following sequence of its interactions with some other player (denoted by X), in the presence of noise

round :	0	1	2	3	4	5
TFT :	–	C	C	D	D	D
X :	–	C	D	D	C	D
TFT-states :	E	R	S	P	T	P

The corresponding plan session for *TFT* is $[EC, RC, SD, PD, TD]$. At 0-th round, there is no interaction, thus the state is E. *TFT* starts by cooperating (1-st round), hence the first action of the plan session is EC. Since player X also cooperates in the 1-st round, the game state at this round is R. *TFT* reciprocates in the 2-nd round by cooperating, hence the second action of the plan session is RC. Similarly for the third and the fourth actions. Now, at the 5-th round, *TFT* should cooperate since X cooperated in 4-th round, but because of noise, it makes an error to defect. Therefore, the 5-th action is TD.

6.3 Plan Corpora Generation

Let us start by generating a plan corpus of seven strategies within the IPD framework: *AllC* (always cooperate), $AllD$ (always defect), *TFT*, *GTFT* (probability of forgiveness a defect is $p = 0.5$), *WSLS*, GRIM and FBF.

We collect plan sessions of each strategy by playing a random choice (C or D) in each round with it. To be more thorough, we can also play all the possible combinations for each given number of rounds to be played. For example, if it is 10, there will be 1024 (2^{10}) combinations—C or D in each

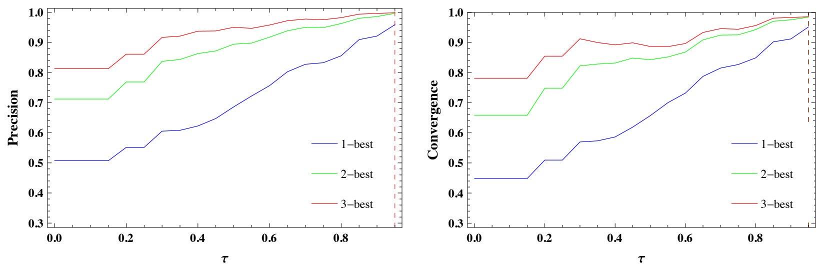


Figure 3: Plot of our method's precision and convergence for $\tau \in [0, 1]$ and for different values of N (N = 1, 2, 3) in IPD Plan Corpus.

round. When noise is present, each combination is played repeatedly several times.

The training corpus is generated by playing with each strategy all the possible combinations 10 times, and for each number of rounds from 5 to 10. The testing dataset is generated by playing a random choice with each strategy in each round, and also for each number of rounds from 5 to 10. We continue until obtaining the same number of sessions as in the training dataset (corpus). Both datasets are generated in presence of noise (namely, an intended move is wrongly performed with probability 0.05).

6.4 Results

The intention recognition model is acquired using the training corpus. Figure 3 shows the precision and convergence of the model with respect to the testing dataset. Given that the training as well as the testing datasets are generated in presence of noise, the achieved intention recognition performance is quite good. Namely, for big enough τ, both precision and convergence scores are greater than 0.9, even for the 1-best case.

7. Related Work

Bayesian networks have been one of the most successful models applied for intention/plan recognition problem (most importantly, see, (Charniak and Goldman 1993; Geib and Goldman 2009)). Depending on the structure of plan libraries, they employed some knowledge-based model construction to build BNs from the library, and then infer the posterior probability of explanations (for the set of observed actions). These works address a number of issues in intention/plan recognition, e.g. the observed agent follows multiple intentions or interleaved plans simultaneously; fails to observe actions; addresses partially ordered plans. However, they made several assumptions for the sake of computational efficiency. First, the prior probabilities of intentions are assumed to be fixed. This assumption is not reasonable because those prior probabilities should depend on the situation at hand, and be captured by the causes/reasons of intentions (see (Pereira and Han 2010) for several examples). In (Pynadath and Wellman 1995), a similar context-dependent approach was used, although the model is not incremental. Second, intentions are assumed to be independent of each other. This is not generally the case since the intentions may support or exclude one another. Those works hence do not appropriately address multiple intention recognition.

This latter assumption must always, explicitly or implicitly, be made by the approaches based on (Hidden) Markov Models, e.g. (Armentano and Amandi 2009; Bui 2003), or statistical corpus-based machine learning (Blaylock and Allen 2004; Blaylock and Allen 2005). Generally, in those approaches, a separate model is built for each intention; thus no relations amongst the intentions are expressed or can be expressed. These works were restricted to the single intention case.

8. Conclusion and Future Work

We have presented a method for incremental intention recognition. The method is performed by dynamically constructing a BN model for intention recognition from a prior domain knowledge base consisting of easily maintained fragments of BN. A fragment consists of a single intention connecting to a single action. This simple network structure allows easy maintenance by domain experts as well as automatically building from available plan corpora.

The main contribution of this paper is our efficient, yet very simple, method for incremental intention recognition. In general, its performance is better than all existent ones that make use of the Linux corpus. Given that our method also has the same linear complexity as other (best) existent ones, and that our model learning process is much simpler, we believe to have achieved significant improvements. In addition, the good performance of the method with respect to the Linux corpus shows its applicability to the important interface-agents domain (Horvitz et al. 1998).

The other contribution, though perhaps only minor, is our method for multiple intention recognition. We have proposed how to represent relationships amongst intentions in the intention recognition model. This aspect is indispensable in multiple intention recognition, but always omitted in previous works. Our next step is to evaluate the method experimentally. Note that although elsewhere reported a capability of dealing with the case where multiple intentions are being followed (e.g. (Geib and Goldman 2009)), to the best of our knowledge that capability has never been evaluated experimentally, partly due to unavailability of appropriate plan corpora or benchmarks. Thus, for the evaluation, we must gather an appropriate plan corpus allowing for the possibility that users pursue multiple intentions simultaneously.

In addition, for the intention recognition community, given the rich set of strategies in the literature (Hofbauer and Sigmund 1998; Sigmund 2010), we have provided here an important, easily extendable benchmark for evaluating intention recognition methods. Given that IPD and other social dilemmas are regularly found in everyday life, and the strategies studied within the framework of those dilemmas actually reflect human behaviors, we believe that game theory (and more generally, evolutionary game theory (Hofbauer and Sigmund 1998)) is a highly promising framework for creating benchmarks for intention recognition. Methods applicable for this benchmark can be used for a wide range of application domains, as diverse as in economics, psychology and biology (Sigmund 2010). For example, our intention recognition model has been successfully used to study the role of intention recognition in the evolution of cooperation, one of the most important issues actively studied in those fields (Han, Pereira, and Santos 2011; Han, Pereira, and Santos).

We also aim at a real deployment for some application domains such as Elder Care (Pereira and Han 2010) and Ambient Intelligence (Han and Pereira 2010b). The simple structure of our BN fragments would enable an easy data collection process.

References

[Armentano and Amandi 2009] Armentano, M. G., and Amandi, A. 2009. Goal recognition with variable-order Markov models. In *IJCAI'09*, 1635–1640.

[Axelrod 1984] Axelrod, R. 1984. *The Evolution of Cooperation*. Basic Books, ISBN 0-465-02122-2.

[Blaylock and Allen 2004] Blaylock, N., and Allen, J. 2004. Statistical goal parameter recognition. In *ICAPS04*, 297–304. AAAI.

[Blaylock and Allen 2005] Blaylock, N., and Allen, J. 2005. Recognizing instantiated goals using statistical methods. In *IJCAI Workshop on Modeling Others from Observations (MOO-2005)*, 79–86.

[Bratman 1987] Bratman, M. E. 1987. *Intention, Plans, and Practical Reason*. The David Hume Series, CSLI.

[Bratman 1992] Bratman, M. E. 1992. Planning and the stability of intention. *Minds and Machines* 2(1):1–16.

[Bui 2003] Bui, H. H. 2003. A general model for online probabilistic plan recognition. In *IJCAI'03*, 1309–1318.

[Charniak and Goldman 1993] Charniak, E., and Goldman, R. 1993. A Bayesian model of plan recognition. *Artificial Intelligence* 64(1):53–79.

[Cozman 2004] Cozman, F. G. 2004. Axiomatizing noisy-or. In *ECAI-04*, 979–980.

[Geib and Goldman 2003] Geib, C. W., and Goldman, R. P. 2003. Recognizing plan/goal abandonment. In *IJCAI'03*.

[Geib and Goldman 2009] Geib, C. W., and Goldman, R. P. 2009. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence* 173(2009):1101–1132.

[Han and Pereira 2010a] Han, T. A., and Pereira, L. M. 2010a. Anytime intention recognition via incremental Bayesian network reconstruction. In *AAAI 2010 Fall Symp. on Proactive Assistant Agents*, 20–25. AAAI.

[Han and Pereira 2010b] Han, T. A., and Pereira, L. M. 2010b. Proactive intention recognition for home ambient intelligence. In *IE Workshop on AI Techniques for Ambient Intelligence*, 91–100. IOS Press.

[Han, Pereira, and Santos] Han, T. A.; Pereira, L. M.; and Santos, F. C. Intention recognition promotes the emergence of cooperation. *Adaptive Behavior*. to appear 2011.

[Han, Pereira, and Santos 2011] Han, T. A.; Pereira, L. M.; and Santos, F. C. 2011. The role of intention recognition in the evolution of cooperative behavior. In *IJCAI'2011*. to appear.

[Heinze 2003] Heinze, C. 2003. *Modeling Intention Recognition for Intelligent Agent Systems*. Ph.D. Dissertation, The University of Melbourne, Australia.

[Hofbauer and Sigmund 1998] Hofbauer, J., and Sigmund, K. 1998. *Evolutionary Games and Population Dynamics*. Cambridge University Press.

[Horvitz et al. 1998] Horvitz, E.; Breese, J.; Heckerman, D.; Hovel, D.; and Rommelse, K. 1998. The Lumiere project: Bayesian user modeling for inferring the goals and needs of software users. In *UAI'98*, 256–265.

[Laskey 2008] Laskey, K. B. 2008. Mebn: A language for first-order Bayesian knowledge bases. *Artificial Intelligence* 172(2-3):140 – 178.

[Lesh 1998] Lesh, N. 1998. *Scalable and Adaptive Goal Recognition*. Ph.D. Dissertation, U. of Washington.

[Linux-Plan-Corpus] Linux-Plan-Corpus. http://www.cs.rochester.edu/research/cisd/resources/linux-plan/. Last access: November 21 2010.

[Natarajan et al. 2008] Natarajan, S.; Tadepalli, P.; Dietterich, T. G.; and Fern, A. 2008. Learning first-order probabilistic models with combining rules. *Annals of Mathematics and Artificial Intelligence* 54:223–256.

[Pearl 1988] Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

[Pearl 2000] Pearl, J. 2000. *Causality: Models, Reasoning, and Inference*. Cambridge U.P.

[Pereira and Han 2009] Pereira, L. M., and Han, T. A. 2009. Intention recognition via causal Bayes networks plus plan generation. In *Procs. of 14th Portuguese Intl. Conf. on Artificial Intelligence (EPIA'09)*, 138–149. LNAI 5816.

[Pereira and Han 2010] Pereira, L. M., and Han, T. A. 2010. Intention recognition with evolution prospection and causal Bayesian networks. In *Computational Intelligence for Engineering Systems*. Springer. 1–33.

[Pynadath and Wellman 1995] Pynadath, D. V., and Wellman, M. P. 1995. Accounting for context in plan recognition, with application to traffic monitoring. In *UAI*.

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

[Sadri 2010] Sadri, F. 2010. Logic-based approaches to intention recognition. In *Handbook of Research on Ambient Intelligence: Trends and Perspectives*.

[Sigmund 2010] Sigmund, K. 2010. *The Calculus of Selfishness*. Princeton U. Press.

[Srinivas 1993] Srinivas, S. 1993. A generalization of the noisy-or model. In *UAI'93*.