Logic, Knowledge Representation and Bayesian Decision Theory

David Poole

Department of Computer Science
University of British Columbia
2366 Main Mall
Vancouver, B.C., Canada V6T 1Z4
poole@cs.ubc.ca
http://www.cs.ubc.ca/spider/poole/

Abstract. In this paper I give a brief overview of recent work on uncertainty in AI, and relate it to logical representations. Bayesian decision theory and logic are both normative frameworks for reasoning that emphasize different aspects of intelligent reasoning. Belief networks (Bayesian networks) are representations of independence that form the basis for understanding much of the recent work on reasoning under uncertainty, evidential and causal reasoning, decision analysis, dynamical systems, optimal control, reinforcement learning and Bayesian learning. The independent choice logic provides a bridge between logical representations and belief networks that lets us understand these other representations and their relationship to logic and shows how they can extended to first-order rule-based representations. This paper discusses what the representations of uncertainty can bring to the computational logic community and what the computational logic community can bring to those studying reasoning under uncertainty.

"It is remarkable that a science which began with the consideration of games of chance should become the most important object of human knowledge...The most important questions of life are, for the most part, really only problems of probability."

"The theory of probabilities is at bottom nothing but common sense reduced to calculus."

— Pierre Simon de Laplace (1794–1827)

1 Introduction

There are good normative arguments for using logic to represent knowledge (Nilsson, 1991; Poole, Mackworth & Goebel, 1998). These arguments are usually based on reasoning with symbols with an explicit denotation, allowing relations amongst individuals, and permitting quantification over individuals. This is often translated as needing (at least) the first-order predicate calculus. Unfortunately, the first-order predicate calculus has very primitive mechanisms for handling uncertainty, namely, the use of disjunction and existential quantification.

There are also good normative reasons for using Bayesian decision theory for decision making under uncertainty (Von Neumann & Morgenstern, 1953; Savage, 1972). These arguments can be intuitively interpreted as seeing decision making as a form of gambling, and that probability and utility are the appropriate calculi for gambling. These arguments lead to the assignment of a single probability to a proposition; thus leading to the notion of probability of a measure of subjective belief. The probability of a proposition for an agent is a measure of the agent's belief in the truth of the proposition. This measure of belief is a function of what the agent knows. Probability theory can be seen as the study of how knowledge affects belief.

It is important to note that decision theory has nothing to say about representations. Adopting decision theory doesn't mean adopting any particular representation. While there are some representations that can be directly extracted from the theory, such as the explicit reasoning over the state space or the use of decision trees, these become intractable as the problem domains become large; it is like theorem proving by enumerating the interpretations. Adopting logic doesn't mean you have to enumerate interpretations or generate the semantic tree (Chang & Lee, 1973), nor does adopting decision theory mean you have to use analogous representations.

First, I will talk about knowledge representation, in which tradition this representation is built. Then I will introduce belief networks. The ICL will then be presented from three alternate viewpoints: as a semantic framework in terms of choices made by agents, in terms of first-order belief networks (Bayesian networks) and as a framework for a abduction and argumentation. I then discuss work on diagnosis, dynamical systems and learning from the uncertainty point of view and relate it to logical representations.

1.1 Knowledge Representation

In order to understand where this work fits in, Figure 1 (from (Poole et al., 1998)) shows the knowledge representation (KR) view. Given a problem we want a solution to, we find a representation for the problem; using this representation we can do computation to find an answer that can then be interpreted as a solution to the problem.

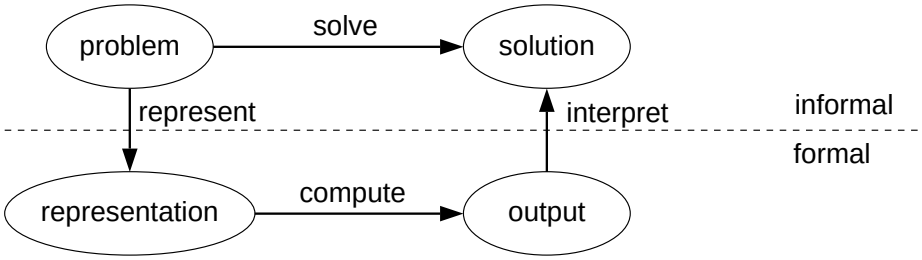

Fig. 1. Knowledge Representation Framework

When considering representations, there are a number of often competing considerations:

– The representation should be rich enough to be able to contain enough information to actually solve the problem.
– The representation should be as close to the problem as possible. We want the representation to be as "natural" as possible, so that a small changes in the problem result in small changes in the representation.
– We want the representation to be amenable to efficient computation. This does not necessarily mean that the representation needs to be efficient in the worst case (because that usually invalidates the first consideration). Rather we would like to be able to exploit features of the problem for computational gain. This means that the representation must be capable of expressing those features of the problem that can be exploited computationally.
– We want to be able to learn the representation from data and from past experiences in solving similar problems.

Belief networks (or Bayesian networks) (Pearl, 1988) are of interest because they provide a language that is represents the sort of knowledge a person may have about a domain, because they are rich enough for many applications, because features of the representation can be exploited for computational gain, and because they can be learned from data. Unfortunately, the underlying logic of belief networks is propositional. We cannot have relations amongst individuals as we can, for example, in the first-order predicate calculus.

2 Belief Networks

Probability specifies a semantic construction and not a representation of knowledge. A belief network (Pearl, 1988) is a way to represent probabilistic knowledge. The idea is to represent a domain in terms of random variables and to explicitly model the interdependence of the random variables in terms of a graph. This is useful when a random variable only depends on a few other random variables, as occurs in many domains. Belief networks form the foundation from which much of the work on uncertainty in AI is built.

Suppose we decide to represent some domain using the random variables[1] $x_1, \ldots, x_n$. Let's totally order the variables. It is straightforward to prove:

$$\begin{aligned} & P(x_1, \ldots, x_n) \\ & = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \cdots P(x_n|x_1 \cdots x_{n-1}) \\ & = \prod_{i=1}^{n} P(x_i|x_1, \ldots, x_{i-1}) \end{aligned}$$

For each variable x_i suppose there is some minimal set $\pi_{x_i} \subseteq \{x_1, \ldots, x_{i-1}\}$ such that

$$P(x_i|x_1, \ldots, x_{i-1}) = P(x_i|\pi_{x_i})$$

[1] Or in terms of propositions. A proposition is a random variable with two possible values *true* and *false* (these are called Boolean random variables). In examples, I will often write $x = true$ as x and $x = false$ as $\neg x$.

That is, once you know the values of the variables in π_{x_i}, knowing the values of other predecessors of x_i in the total ordering will not change your belief in x_i. The elements of the set π_{x_i} are known as the **parents** of variable x_i. We say x_i is **conditionally independent** of its predecessors given its parents. We can create a graph where there is an arc from each parent of a node into that node. Such a graph, together with the conditional probabilities for $P(x_i|\pi_{x_i})$ for each variable x_i is known as a **belief network** or a **Bayesian network** (Pearl, 1988; Jensen, 1996).

Example 1. An example belief network is given in Figure 2. The parents of *projector_lamp_on* are *power_in_projector* and *lamp_works*. Note that this graph does not specify how *power_in_projector* depends on *projector_lamp_on* and *lamp_works*. It does, however, specify that *power_in_projector* is independent of *power_in_building*, *alan_reading_book* and the other non-descendent given these parents. Separately we need a specification of how each variable depends on its parents.

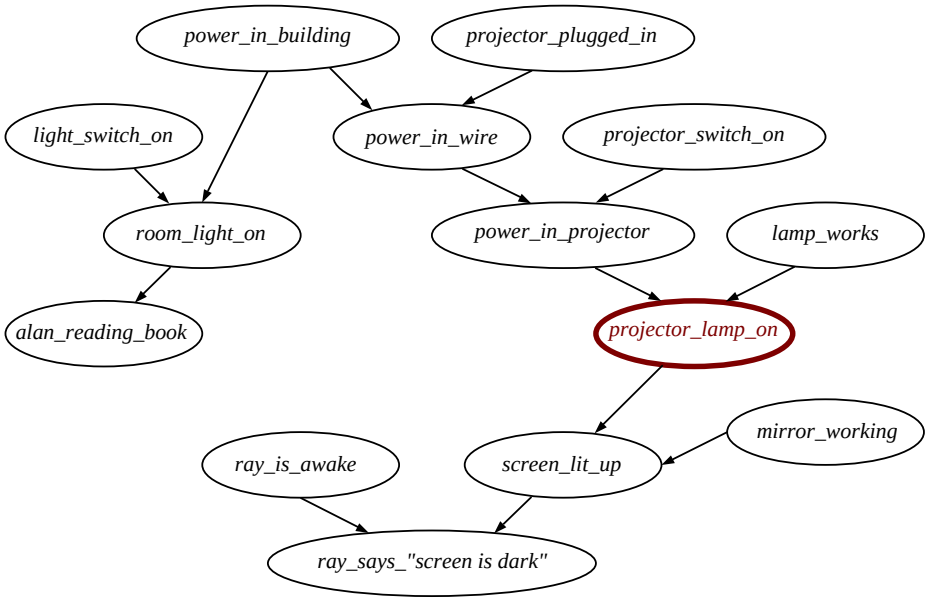

Fig. 2. A belief network for an overhead projector (we discuss the node in bold)

There are a few important points to notice about a Bayesian network:

- By construction, the graph defining a Bayesian network is acyclic.
- Different total orderings of the variables can result in different Bayesian networks for the same underlying distribution.
- The size of a conditional probability table for $P(x_i|\pi_{x_i})$ is exponential in the number of parents of x_i.

Typically we try to build belief networks so that the total ordering results in few parents and a sparse graph. Belief networks can be constructed taking into account just local information, the information that has to be specified is reasonably intuitive, and there are many domains that have concise representations as belief networks. There are algorithms that can exploit the sparseness of the graph for computational gain (Lauritzen & Spiegelhalter, 1988; Dechter, 1996; Zhang & Poole, 1996), exploit the skewness of distributions (Poole, 1996a), use the structure for stochastic simulation (Henrion, 1988; Pearl, 1987; Dagum & Luby, 1997) or exploit special features of the conditional probabilities (Zhang & Poole, 1996; Poole, 1997b; Jordan, Ghahramani, Jaakkola & Saul, 1997). They can be learned from data (Heckerman, 1995).

Notice that there is nothing *causal* about the definition of a belief network. However, there have been much work on relating belief networks and causality (Pearl, 1999; Pearl, 2000). There are a number of good reasons for this:

- If the direct clauses of a variable are its parents, one would expect that causation would follow the independence of belief networks. Thus if you wanted to represent causal knowledge a belief network would be appropriate.
- There is a conjecture that representing knowledge causally (with direct causes as parents) results in a sparser network that is more stable to changing contexts. This seems to be born out by experience of many people in building these networks.
- A causal network also lets us predict the effect of an intervention: what happens of we change the value of a variable. This is important when we want an agent to affect the value of a variable (e.g., to decide whether to smoke).

However, it must be emphasised that a belief network can represent non-causal relationships as well.

3 The Independent Choice Logic

The independent choice logic (ICL) is a knowledge representation that can be seen in a number of different ways (see Figure 3):

- It is a way to add Bayesian probability to the predicate logic. In particular we want to have all uncertainty to be handled by probabilities (or for decision problems, as choices of various agents). So we start with logic programs, which can be seen as predicate logic with no uncertainty (no disjunctive assertions), and have independent choices that have associated probability distributions. A logic program specifies what follows from the choices made.
- It is a way to lift Bayesian networks into a first-order language. In particular a Bayesian network can be seen as a deterministic system with "noise" (independent stochastic) inputs (Pearl, 1999; Pearl, 2000). In the ICL, the deterministic system is modelled as a logic program. Thus we write the conditional probabilities in rule form. The noise inputs are given in terms of independent choices.
- It is a sound way to have probabilities over assumptions. Explaining observations means that we use abduction; we find the explanations (set of hypotheses) that imply the observations, and from these we make predictions. This reasoning is sound probabilistic inference in the ICL.

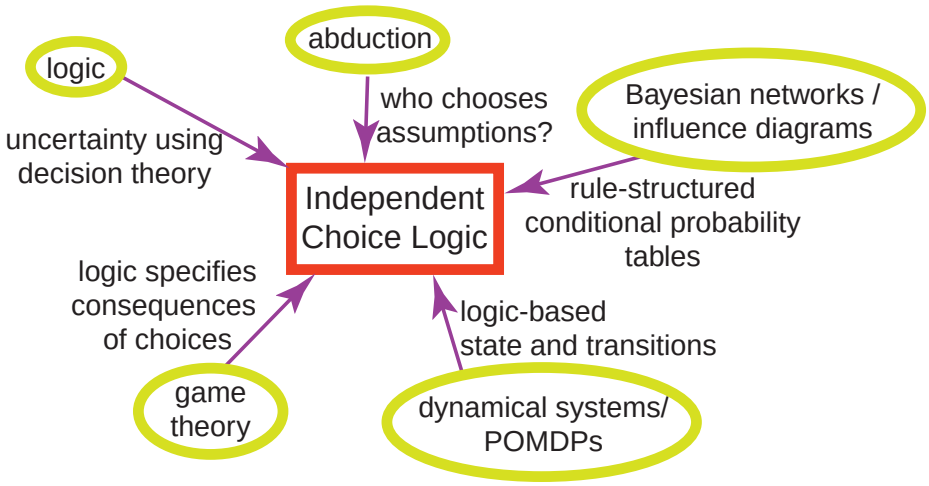

Fig. 3. ICL Influences

The ICL started off as Probabilistic Horn Abduction (Poole, 1991a; Poole, 1991b; Poole, 1993a; Poole, 1993b) (the first three had a slightly different language). The independent choice logic extends probabilistic Horn abduction in allowing for multiple agents making choices (Poole, 1997a) (where nature is a special agent who makes choices probabilistically) and in allowing negation as failure in the logic (Poole, 2000a).

3.1 The Language

In this section we give the language and the semantics of the ICL. This is simplified slightly; the general ICL allows for choices by various agents (Poole, 1997b) which lets us model decisions in a decision-theoretic (single agent) or game-theoretic (multiple agents) situation.

We assume that we have atomic formulae as in a normal logical language (Lloyd, 1987). We use the Prolog convention of having variables in upper case, and predicate symbol and function symbols in lower case.

A **clause** is either an atom or is of the form

$$h \leftarrow a_1 \wedge \cdots \wedge a_k$$

where h is an atom and each a_i is an atom or the negation of an atom.

A **logic program** is a set of clauses. We assume the logic program is acyclic[2] (Apt & Bezem, 1991).

An **atomic choice** is an atom that does not unify with the head of any clause. An **alternative** is a set of atomic choices. A **choice space** is a set of alternatives such that an atomic choice can be in at most one alternative.

[2] All recursions for variable-free queries eventually halt. We disallow programs such as $\{a \leftarrow \neg a\}$ and $\{a \leftarrow \neg b, b \leftarrow \neg a\}$. We want to ensure that there is a unique model for each logic program.

An ICL theory consists of

F the facts, an acyclic logic program
C a choice space
P_0 a probability distribution over the alternatives in C. That is $P_0 : \cup\mathbf{C} \rightarrow [0, 1]$ such that

$$\forall \chi \in \mathbf{C} \sum_{\alpha \in \chi} P_0(\alpha) = 1$$

Example 2. Here is a meaningless example:

$$\mathbf{C} = \{\{c_1, c_2, c_3\}, \{b_1, b_2\}\}$$

$$\begin{aligned}\mathbf{F} = \{ & f \leftarrow c_1 \wedge b_1, \; f \leftarrow c_3 \wedge b_2, \\ & d \leftarrow c_1, \qquad d \leftarrow \neg c_2 \wedge b_1, \\ & e \leftarrow f, \qquad e \leftarrow \neg d\}\end{aligned}$$

$$\begin{aligned}&P_0(c_1) = 0.5 \; P_0(c_2) = 0.3 \; P_0(c_3) = 0.2 \\ &P_0(b_1) = 0.9 \; P_0(b_2) = 0.1\end{aligned}$$

3.2 Semantics

The semantics is defined in terms of possible worlds. Here we present the semantics for the case of a finite choice space, where there are only finitely many possible worlds. The more general case is considered in other places (Poole, 1997b; Poole, 2000a).

A **total choice** for choice space C is a selection of exactly one atomic choice from each alternative in C.

There is a **possible world** for each total choice. What is **true** in a possible world is defined by the atoms chosen by the total choice together with the logic program. In particular an atom is true if it in in the (unique) stable model[3] of the total choice together with the logic program (Poole, 2000a). The measure of a possible world is the product of the values $P_0(\alpha)$ for each α selected by the total choice.

The probability of a proposition is the sum of the measures of the possible worlds in which the proposition is true.

Example 3. In the ICL theory of example 2, there are six possible worlds:

$$\begin{aligned}
w_1 &\models c_1 \; b_1 \quad f \quad d \quad e \qquad P(w_1) = 0.45 \\
w_2 &\models c_2 \; b_1 \; \neg f \; \neg d \quad e \qquad P(w_2) = 0.27 \\
w_3 &\models c_3 \; b_1 \; \neg f \quad d \; \neg e \qquad P(w_3) = 0.18 \\
w_4 &\models c_1 \; b_2 \; \neg f \quad d \; \neg e \qquad P(w_4) = 0.05 \\
w_5 &\models c_2 \; b_2 \; \neg f \; \neg d \quad e \qquad P(w_5) = 0.03 \\
w_6 &\models c_3 \; b_2 \quad f \; \neg d \quad e \qquad P(w_6) = 0.02
\end{aligned}$$

The probability of any proposition can be computed by summing the measures of the worlds in which the proposition is true. For example

$$P(e) = 0.45 + 0.27 + 0.03 + 0.02 = 0.77$$

[3] The acyclicity of the logic program and the restriction that atomic choices don't unify with the head of clauses guarantees there there is a single model for each possible world.

3.3 ICL and Belief networks

It may seem that, with independent alternatives, that the ICL is restricted in what it can represent. This is not the case; in particular it can represent anything the is representable by a Belief network. Moreover the translation is local, and (if all variables and alternatives are binary) there is the same number of alternatives as there are free parameters in the belief network.

Example 4. If we had Boolean variables a, b and c, where b and c are the parents of a, we will have rules such as

$$a \leftarrow b \wedge \neg c \wedge aifbnc$$

where *aifbnc* is an atomic choice where $P_0(aifbnc)$ has the same value as the conditional probability as $P(a|b, \neg c)$ in the belief network. This generalizes to arbitrary discrete belief networks in the analogous way (Poole, 1993b).

This representation lets us naturally specify context-specific independence (Boutilier, Friedman, Goldszmidt & Koller, 1996; Poole, 1997b), where, for example, a may be independent of c when b is false but be dependent when b is true. Context-specific independence is often specified in terms of a tree for each variable; the tree has probabilities at the leaves and parents of the variable on the internal nodes. It is straightforward to translate these into the ICL.

Example 5. In the belief network of Figure 2, we can axiomatize how *power_in_projector* depends on *projector_lamp_on* and *lamp_works*:

$$\begin{aligned}
&projector_lamp_on \leftarrow \\
&\quad power_in_projector \wedge \\
&\quad lamp_works \wedge \\
&\quad projector_working_ok. \\
&projector_lamp_on \leftarrow \\
&\quad power_in_projector \wedge \\
&\quad \neg lamp_works \wedge \\
&\quad working_with_faulty_lamp.
\end{aligned}$$

We also have the alternatives:

$$\{projector\_working\_ok, projector\_broken\}$$
$$\{working\_with\_faulty\_lamp, not\_working\_with\_faulty\_lamp\}$$

The ICL lets us see the relationship of Belief networks to logical languages. The logic programs are standard logic programs (they can even have negation as failure (Poole, 2000a)). Viewing them as logic programs gives us a natural way to lift belief networks to the first-order case (i.e., with logical variables universally quantified over individuals).

3.4 ICL, Abduction and Logical Argumentation

The ICL can also be seen as a language for abduction. In particular, if all of the atomic choices are assumable (they are abducibles or possible hypotheses). An **explanation**[4] for g is a consistent set of assumables that implies g. A set of atomic choices is consistent if there is at most one element in any alternative.

An explanation can be seen as an argument based on explicit assumptions about what is true. Each of these explanations has an associated probability obtained by computing the product of the probabilities of the atomic choices that make up the explanation. The probability of g can be computed by summing[5] the probabilities of the explanations for g (Poole, 1993b; Poole, 2000a).

If we want to do evidential reasoning and observe obs, we compute

$$P(g|obs) = \frac{P(g \wedge obs)}{P(obs)}$$

In terms of explanations, we can first find the explanations for obs (which would give us $P(obs)$) and then try to extend these explanations to also explain g (this will give us $P(g \wedge obs)$). Intuitively, we explain all of the observations and see what these explanations also predict. This is similar to proposals in the nonmonotonic reasoning community to mix abduction and default reasoning (Poole, 1989; Shanahan, 1989; Poole, 1990).

We can also bound the prior and posterior probabilities by generating only a few of the most plausible explanations (either top-down (Poole, 1993a) or bottom-up (Poole, 1996b)). Thus we can use inference to the best explanations to do sound (approximate) probabilistic reasoning.

3.5 Reasoning in the ICL

To do reasoning in the ICL we can either do

- variable elimination (marginalization) to simplify the model (Poole, 1997b). We sum out variables to reduce the detail of the representation. This is similar to partial evaluation in logic programs.
- Generating some of the explanations to bound the probabilities (Poole, 1993a; Poole, 1996a). If we generated all of the explanations we could compute the probabilities exactly, but there are combinatorially many explanations.
- Stochastic simulation; generating the needed atomic choices stochastically, and estimating the probabilities by counting the resulting proportions.

---

[4] We need to extend the definition of explanation to account for negation as failure. The explanation of $\neg a$ are the duals of the explanations of a (Poole, 2000a).

[5] This assumes the bodies for the rules for each atom a are mutually exclusive. This is a common practice in logic programming and the rules obtained from the translation from belief networks have this property. We need to do something a bit more sophisticated if the rules are not disjoint (Poole, 2000a).

4 Relating Work in Other Fields

4.1 Reasoning about actions

In this section I will review some of the work about actions outside of the logic camp. See Shanahan (1997) for a review of the logicist approach to representing actions; I do not have the space to review this here.

Much work in AI, dynamical systems, stochastic control, and operations research is built on the motion of a Markov process (see for example (Luenberger, 1979; Bertsekas, 1995; Boutilier, Dean & Hanks, 1999)), where there is a state variable that depends on the previous state and the action being carried out. In general, we don't observe the state, but only get to observe what our sensors provide. When an agent makes a decision the only information available is the history of observations and actions.

One case with no control is the hidden Markov model (HMM); this can be seen as a simple belief network as in Figure 4. In this figure s_t is random variable representing the

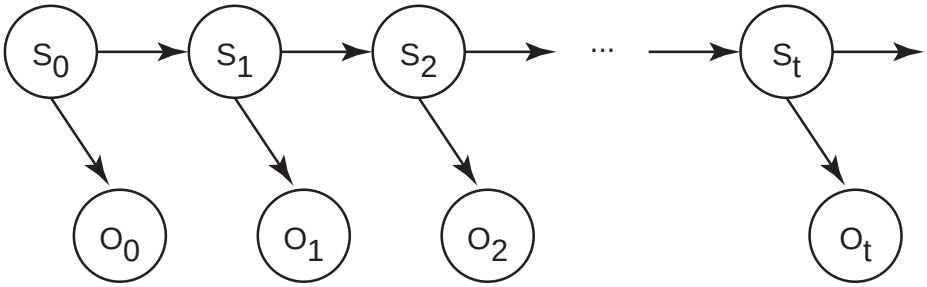

Fig. 4. Belief network corresponding to a hidden Markov model

state at time[6] t and o_t is a random variable representing the observation at time t. The probabilities we need to specify are $P(s_0)$, $P(s_t|s_{t-1})$ and $P(o_t|s_t)$. These represent the initial state, the system dynamics and the observation model respectively.

We can use the general mechanism to convert this to a logic program. The result looks like:

$$state(S,T) \leftarrow T > 0 \wedge state(S1, T-1) \wedge trans(S1, S)$$

where there is an alternative for each state si

$$\{trans(si, s0), trans(si, s1), \ldots, trans(si, sn)\}$$

where the states are $s0, s1, \ldots, sn$. We only need to include those transitions that have a non-zero probability. Omitting the zero probabilities can be exploited in sparse matrix computations.

[6] This is either fixed time steps or is based on the times of interesting events. In the latter case $T+1$ is the time of the next interesting event (or the state that results from the action). There is also a large body of work on continuous time dynamical systems that I won't review.

We don't want to specify each state by name, but would rather describe the properties of states. That is we describe the states in terms of random variables (or propositions). In the probabilistic literature this is known as dynamic belief networks (or dynamic Bayes networks) (Dean & Kanazawa, 1989; Dean & Wellman, 1991). In a dynamic belief network we divide the state into a number of random variables and then specify how each variable depends on values at the same[7] and previous times.

In the ICL, the direct translation results in rules like:

$$a(T) \leftarrow a_1(T-1) \wedge \ldots \wedge a_k(T-1) \wedge b_1(T) \wedge \ldots b_r(T) \wedge n(T)$$

where the a_i and b_i are literal fluents and $n(T)$ is an atomic choice (there is a different atomic choice for each combinations of the a_i and b_j).

When we have a control problem, (such as in Markov decision processes) we have to choose the actions based on the information available (the history of actions of and observations). In this case, using the same representation as we used for conditional probabilities, a policy in the ICL is represented as a logic program that specifies what an agent will do based on its history (Poole, 1997a). We can also use conditional plans to represent policies (Poole, 1998; Bacchus, Halpern & Levesque, 1999).

There are many dimensions on which to compare different representations for dynamics:

- deterministic versus stochastic dynamics; whether an action from a state results in a known state or results in a distribution over states.
- goal versus values; whether we can only say that some goal needs to be achieved, or we give a cardinal rating of all of the resulting states, (for example rating how bad a possible undesirable state is).
- finite stage versus infinite stage; whether we plan for a specific given number of future actions or for an indeterminate number of future actions.
- fully observable versus partial observability; whether the agent gets to observe (or knows) the actual state it is in when it has to decide what to do, or whether it has only limited and noisy sensors of the state.
- explicit state space versus states described in terms of properties (using random variables or propositions); whether there is a single state variable or the state is factored into a number of random variables.
- zeroth-order versus first-order; whether we can quantify over individuals or not.
- given dynamics and rewards versus dynamics and rewards acquired through interaction with the world; whether we must learn through trial and error the dynamics and the value or whether the dynamics is provided.
- single agent versus multiple agents
- perfect rationality versus bounded rationality; whether we can assume that the agent has unbounded computation or whether it must act within time and space limitations (Simon, 1996; Horvitz, 1989; Russell & Subramanian, 1995; Russell, 1997).

[7] We need to be able to specify how variables depend on other variables at the same time to account for correlated action effects. This could also be achieved by inventing new variables (that represent a common cause that makes two effects correlated). Of course, we still must maintain the acyclicity of the resulting belief network.

For each of these choices, the left-hand alternative is simpler than the right-hand one. We know how to build agents that only have a few of the right-hand sides. However, when we have more of the right-hand sides, we know that the problems are much more computationally difficult.

For example, when there stochastic dynamics, values, infinite stages and partially observable, we get partially observable Markov decision processes (POMDPs) (Cassandra, Kaelbling & Littman, 1994). Even the most efficient exact algorithms known (Cassandra, Littman & Zhang, 1997) can only work for a few hundred states[8]. Interestingly, these exact algorithms are essentially backward conditional planning algorithms, where multiple conditional plans are maintained. The difficult problem is to determine which plans stochastically dominate others (see Poole, 1998, for a review).

Similarly, where there are multiple agents, determining locally optimal solutions for each agent (Nash equilibria) is exponentially more difficult than the corresponding single-agent case (Koller & Megiddo, 1992).

	(a) CP	(b) DTP	(c) IDs	(d) RL	(e) HMM	(f) GT
Stochastic dynamics		✔	✔	✔	✔	✔
Values		✔	✔	✔		✔
infinite stage	✔	✔		✔	✔	
partially observable			✔		✔	✔
random variables	✔	✔	✔	✔		✔
first-order	✔					
dynamics not given				✔	✔	
multiple agents						✔
bounded rationality						

- (a) classical planning (e.g., Strips (Fikes & Nilsson, 1971) or the Situation Calculus (McCarthy & Hayes, 1969))
- (b) decision-theoretic planning (Boutilier, Dearden & Goldszmidt, 1995; Boutilier et al., 1999)
- (c) influence diagrams (Howard & Matheson, 1984)
- (d) reinforcement learning (Sutton & Barto, 1998; Kaelbling, Littman & Moore, 1996; Bertsekas & Tsitsiklis, 1996)
- (e) hidden Markov models (Jurafsky & Martin, 2000; Rabiner, 1989)
- (f) game theory: the extensive form of a game (Von Neumann & Morgenstern, 1953; Ordeshook, 1986; Myerson, 1991; Fudenberg & Tirole, 1992)

Fig. 5. Comparing Models of Dynamics

Figure 5 shows various representations and how they differ on the dimensions above. What is important to notice is that they share the same underlying notion of dynamics and the translation into belief networks (and ICL) is like that of the HMMs).

[8] There are excellent online resources on POMDPs by Tony Cassandra (`http://www.cs.brown.edu/research/ai/pomdp/index.html`) and Michael Littman (`http://www.cs.duke.edu/~mlittman/topics/pomdp-page.html`).

Reinforcement learning (Sutton & Barto, 1998; Kaelbling et al., 1996; Bertsekas & Tsitsiklis, 1996) is an interesting case of the general paradigm of understanding dynamics under uncertainty. While there has been much work with states described in terms of properties, virtually all of this learns the the value function (or the state transition function and the reward function) in terms of neural networks. There is one notable exception; Chapman & Kaelbling (1991) use decision trees (which can easily be converted into rules) to represent value functions (Q-functions).

One other interesting comparison is with hidden Markov models that have been used extensively in speech recognition (Rabiner, 1989; Jurafsky & Martin, 2000). In other work, Hobbs, Stickel, Appelt & Martin (1993) use a language similar to the independent choice logic (but with "costs" that are added; these costs can be seen a log-probabilities) to represent a way to combine syntax, semantic and pragmatic preferences into a coherent framework. The ICL show a way how these two, seemingly unrelated pieces of work can be combined into a coherent framework.

4.2 Model-based diagnosis

There is a large body of work on model-based diagnosis using belief networks and decision analysis tools based on these such as influence diagrams (Henrion, Breese & Horvitz, 1991). Essentially we write a forward simulation of the system, making explicit the possible faults and the uncertainty involved in the working of normal and faulty components. In terms of the ICL, we write a logic program that implies the outputs from the inputs, the status of the components and the stochastic mechanisms. There is a strong relationship between the search methods for belief networks and the traditional methods for model-based diagnosis (Poole, 1996a).

4.3 Bayesian Leaning

There is a large body of work on learning and belief networks. This means either:

- Using the belief network as a representation for the problem of Bayesian learning of models (Buntine, 1994). In Bayesian learning, we want the posterior distribution of hypotheses (models) given the data. To handle multiple cases, Buntine uses the notion of plates that corresponds to the use of logical variables in the ICL (Poole, 2000b). Poole (2000b) shows the tight integration of abduction and induction. These papers use belief networks to learn various representations including decision trees and neural networks, as well us unsupervised learning.
- Learning the structure and probabilities of belief networks (Heckerman, 1995). We can use Bayesian learning or other learning techniques to learn belief networks. One of the most successful methods is to learn a decision tree for each variable given its predecessors in a total ordering (Friedman & Goldszmidt, 1996; Chickering, Heckerman & Meek, 1997), and then search over different total orderings. It is straightforward to translate from these decision trees to the ICL.

The ICL can also be compared to the stochastic logic programs of Muggleton (1995). Stochastic logic programs allow for annotated logic programs of the form:

$$p : h \leftarrow a_1 \wedge \ldots \wedge a_k$$

This can be seen as similar to the ICL rule:

$$h \leftarrow a_1 \wedge \ldots \wedge a_k \wedge n_p$$

where n_p is an atomic choice with $P_0(n_p) = p$. The definition of stochastic logic programs has problems with programs such as:

$$1.0 : a \leftarrow b \wedge c$$
$$0.5 : b$$
$$1.0 : c \leftarrow b$$

Intuitively a should have probability one half (as it is true whenever b is true, and b is true half the time). Stochastic logic programs double-count b, which is used in the proof for a twice. The use of atomic choices lets us not double count, as we keep track of the assumptions used (and only use them once in the set of assumptions for a goal). The semantics of the ICL is simpler than the semantics for stochastic logic programs; all of the clauses in the ICL have their standard meaning.

The ICL has the potential to form the basis for an integration of inductive logic programming (Muggleton & De Raedt, 1994; Quinlan & Cameron-Jones, 1995; Muggleton, 1995) with reinforcement learning and leaning of belief networks.

5 Conclusion

This paper has provided a too-brief sketch of work in uncertainty in AI. I aimed to show that belief networks provide a way to understand much of the current work in stochastic dynamical systems, diagnosis and learning under uncertainty. The ICL provides a bridge between that work and the work in the logic community. Eventually we will need to build systems with first-order representations and reason about uncertainty, dynamics and learning. Hopefully I have provided some idea of how this could be achieved. There is still much work to be done.

Acknowledgements

This work was supported by Institute for Robotics and Intelligent Systems and the Natural Sciences and Engineering Research Council of Canada Operating Grant OGPOO44121.

References

Apt, K. R. & Bezem, M. (1991). Acyclic programs, *New Generation Computing* **9**(3-4): 335–363.

Bacchus, F., Halpern, J. Y. & Levesque, H. J. (1999). Reasoning about noisy sensors and effectors in the situation calculus, *Artificial Intelligence* **111**(1–2): 171–208.
URL: *http://www.lpaig.uwaterloo.ca/˜fbacchus/on-line.html*

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*, Athena Scientific, Belmont, Massachusetts. Two volumes.

Bertsekas, D. P. & Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*, Athena Scientific, Belmont, Massachusetts.

Boutilier, C., Dean, T. & Hanks, S. (1999). Decision-theoretic planning: Structual assumptions and computational leverage, *Journal of Artificial Intelligence Research* **11**: 1–94.

Boutilier, C., Dearden, R. & Goldszmidt, M. (1995). Exploiting structure in policy construction, *Proc. 14th International Joint Conf. on Artificial Intelligence (IJCAI-95)*, Montréal, Québec, pp. 1104–1111.

Boutilier, C., Friedman, N., Goldszmidt, M. & Koller, D. (1996). Context-specific independence in Bayesian networks, *in* E. Horvitz & F. Jensen (eds), *Proc. Twelfth Conf. on Uncertainty in Artificial Intelligence (UAI-96)*, Portland, OR, pp. 115–123.

Buntine, W. L. (1994). Operations for learning with graphical models, *Journal of Artificial Intelligence Research* **2**: 159–225.

Cassandra, A., Littman, M. & Zhang, N. (1997). Incermental pruning: A simple, fast, exact mehtod for partially observable markov decision processes, *in* D. Geiger & P. Shenoy (eds), *Proc. Thirteenth Conf. on Uncertainty in Artificial Intelligence (UAI-97)*, pp. ??–??

Cassandra, A. R., Kaelbling, L. P. & Littman, M. L. (1994). Acting optimally in partially observable stochastic domains, *Proc. 12th National Conference on Artificial Intelligence*, Seattle, pp. 1023–1028.

Chang, C. L. & Lee, R. C. T. (1973). *Symbolic Logical and Mechanical Theorem Proving*, Academic Press, New York.

Chapman, D. & Kaelbling, L. P. (1991). Input generlization in delayed reinforcement learning: An algorithm and performance comparisons, *Proc. 12th International Joint Conf. on Artificial Intelligence (IJCAI-91)*, Sydney, Australia.

Chickering, D. M., Heckerman, D. & Meek, C. (1997). A bayesian approach to learning bayesian networks with local structure, *Proc. Thirteenth Conf. on Uncertainty in Artificial Intelligence (UAI-97)*, pp. 80–89.

Dagum, P. & Luby, M. (1997). An optimal approximation algorithm for Bayesian inference, *Artificial Intelligence* **93**(1–2): 1–27.

Dean, T. & Kanazawa, K. (1989). A model for reasoning about persistence and causation, *Computational Intelligence* **5**(3): 142–150.

Dean, T. L. & Wellman, M. P. (1991). *Planning and Control*, Morgan Kaufmann, San Mateo, CA.

Dechter, R. (1996). Bucket elimination: A unifying framework for probabilistic inference, *in* E. Horvitz & F. Jensen (eds), *Proc. Twelfth Conf. on Uncertainty in Artificial Intelligence (UAI-96)*, Portland, OR, pp. 211–219.

Fikes, R. E. & Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving, *Artificial Intelligence* **2**(3-4): 189–208.

Friedman, N. & Goldszmidt, M. (1996). Learning Bayesian networks with local structure, *Proc. Twelfth Conf. on Uncertainty in Artificial Intelligence (UAI-96)*, pp. 252–262.
URL: *http://www2.sis.pitt.edu/ dsl/UAI/UAI96/Friedman1.UAI96.html*

Fudenberg, D. & Tirole, J. (1992). *Game Theory*, MIT Press, Cambridge, MA.

Heckerman, D. (1995). A tutorial on learning with Bayesian networks, *Technical Report MSR-TR-95-06*, Microsoft Research. (Revised November 1996).
URL: *http://www.research.microsoft.com/research/dtg/heckerma/heckerma.html*

Henrion, M. (1988). Propagating uncertainty in Bayesian networks by probabilistic logic sampling, *in* J. F. Lemmer & L. N. Kanal (eds), *Uncertainty in Artificial Intelligence 2*, Elsevier Science Publishers B.V., pp. 149–163.

Henrion, M., Breese, J. & Horvitz, E. (1991). Decision analysis and expert systems, *AI Magazine* **12**(4): 61–94.

Hobbs, J. R., Stickel, M. E., Appelt, D. E. & Martin, P. (1993). Interpretation as abduction, *Artificial Intelligence* **63**(1–2): 69–142.

Horvitz, E. J. (1989). Reasoning about beliefs and actions under computational resource constraints, *in* L. Kanal, T. Levitt & J. Lemmer (eds), *Uncertainty in Artificial Intelligence 3*, Elsevier, New York, pp. 301–324.

Howard, R. A. & Matheson, J. E. (1984). Influence diagrams, *in* R. A. Howard & J. E. Matheson (eds), *The Principles and Applications of Decision Analysis*, Strategic Decisions Group, Menlo Park, CA.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks*, Springer Verlag, New York.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S. & Saul, L. K. (1997). An introduction to variational methods for graphical models, *Technical report*, MIT Computational Cognitive Science. **URL:** *http://www.ai.mit.edu/projects/jordan.html*

Jurafsky, D. & Martin, J. (2000). *Speech and Language Processing*, Prentice Hall.

Kaelbling, L. P., Littman, M. L. & Moore, A. W. (1996). Reinforcement learning: A survey, *Journal of Artificial Intelligence Research* **4**: 237–285.

Koller, D. & Megiddo, N. (1992). The complexity of two-person zero-sum games in extensive form, *Games and Economic Behavior* **4**: 528–552.

Lauritzen, S. L. & Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems, *Journal of the Royal Statistical Society, Series B* **50**(2): 157–224.

Lloyd, J. W. (1987). *Foundations of Logic Programming*, Symbolic Computation Series, second edn, Springer-Verlag, Berlin.

Luenberger, D. G. (1979). *Introduction to Dynamic Systems: Theory, Models and Applications*, Wiley, New York.

McCarthy, J. & Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence, *in* M. Meltzer & D. Michie (eds), *Machine Intelligence 4*, Edinburgh University Press, pp. 463–502.

Muggleton, S. (1995). Inverse entailment and Progol, *New Generation Computing* **13**(3,4): 245–286.

Muggleton, S. & De Raedt, L. (1994). Inductive logic programming: Theory and methods, *Journal of Logic Programming* **19,20**: 629–679.

Myerson, R. B. (1991). *Game Theory: Analysis of Conflict*, Harvard University Press, Cambridge, MA.

Nilsson, N. J. (1991). Logic and artificial intelligence, *Artificial Intelligence* **47**: 31–56.

Ordeshook, P. C. (1986). *Game theory and political theory: An introduction*, Cambridge University Press, New York.

Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models, *Artificial Intelligence* **32**(2): 245–257.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, CA.

Pearl, J. (1999). Reasoning with cause and effect, *Proc. 16th International Joint Conf. on Artificial Intelligence (IJCAI-99)*, pp. 1437–1449.

Pearl, J. (2000). *Causality: Models, Reasoning and Inference*, Cambridge University Press.

Poole, D. (1989). Explanation and prediction: An architecture for default and abductive reasoning, *Computational Intelligence* **5**(2): 97–110.

Poole, D. (1990). A methodology for using a default and abductive reasoning system, *International Journal of Intelligent Systems* **5**(5): 521–548.

Poole, D. (1991a). Representing diagnostic knowledge for probabilistic Horn abduction, *Proc. 12th International Joint Conf. on Artificial Intelligence (IJCAI-91)*, Sydney, pp. 1129–1135.

Poole, D. (1991b). Search-based implementations of probabilistic Horn abduction, *Technical report*, Department of Computer Science, University of British Columbia, Vancouver, B.C., Canada.

Poole, D. (1993a). Logic programming, abduction and probability: A top-down anytime algorithm for computing prior and posterior probabilities, *New Generation Computing* **11**(3–4): 377–400.

Poole, D. (1993b). Probabilistic Horn abduction and Bayesian networks, *Artificial Intelligence* **64**(1): 81–129.

Poole, D. (1996a). Probabilistic conflicts in a search algorithm for estimating posterior probabilities in Bayesian networks, *Artificial Intelligence* **88**: 69–100.

Poole, D. (1996b). Probabilistic conflicts in a search algorithm for estimating posterior probabilities in Bayesian networks, *Artificial Intelligence* **88**: 69–100.

Poole, D. (1997a). The independent choice logic for modelling multiple agents under uncertainty, *Artificial Intelligence* **94**: 7–56. special issue on economic principles of multi-agent systems.
URL: *http://www.cs.ubc.ca/spider/poole/abstracts/icl.html*

Poole, D. (1997b). Probabilistic partial evaluation: Exploiting rule structure in probabilistic inference, *Proc. 15th International Joint Conf. on Artificial Intelligence (IJCAI-97)*, Nagoya, Japan, pp. 1284–1291.
URL: *http://www.cs.ubc.ca/spider/poole/abstracts/pro-pa.html*

Poole, D. (1998). Decision theory, the situation calculus and conditional plans, *Electronic Transactions on Artificial Intelligence* **2**(1–2).
URL: *http://www.etaij.org*

Poole, D. (2000a). Abducing through negation as failure: stable models in the Independent Choice Logic, *Journal of Logic Programming* **44**(1–3): 5–35.
URL: *http://www.cs.ubc.ca/spider/poole/abstracts/abnaf.html*

Poole, D. (2000b). Learning, bayesian probability, graphical models, and abduction, *in* P. Flach & A. Kakas (eds), *Abduction and Induction: essays on their relation and integration*, Kluwer.

Poole, D., Mackworth, A. & Goebel, R. (1998). *Computational Intelligence: A Logical Approach*, Oxford University Press, New York.

Quinlan, J. R. & Cameron-Jones, R. M. (1995). Induction of logic programs: FOIL and related systems, *New Generation Computing* **13**(3,4): 287–312.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition, *Proceedings of the IEEE* **77**(2): 257–286.

Russell, S. (1997). Rationality and intelligence, *Artificial Intelligence* **94**: 57–77.

Russell, S. J. & Subramanian, D. (1995). Provably bounded-optimal agents, *Journal of Artificial Intelligence Research* **2**: 575–609.

Savage, L. J. (1972). *The Foundation of Statistics*, 2nd edn, Dover, New York.

Shanahan, M. (1989). Prediction is deduction, but explanation is abduction, *Proc. 11th International Joint Conf. on Artificial Intelligence (IJCAI-89)*, Detroit, MI, pp. 1055–1060.

Shanahan, M. (1997). *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*, MIT Press, Cambridge, MA.

Simon, H. (1996). *The Sciences of the Artificial*, third edn, MIT Press, Cambridge, MA.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*, MIT Press, Canbridge, MA.

Von Neumann, J. & Morgenstern, O. (1953). *Theory of Games and Economic Behavior*, third edn, Princeton University Press, Princeton, NJ.

Zhang, N. & Poole, D. (1996). Exploiting causal independence in Bayesian network inference, *Journal of Artificial Intelligence Research* **5**: 301–328.