Learning to Reason with a Restricted View

Roni Khardon* **Dan Roth†**

Aiken Computation Laboratory,
Harvard University,
Cambridge, MA 02138.
{roni,danr}@das.harvard.edu

Abstract

The current emphasis of the research in learning theory is on the study of inductive learning (from examples) of concepts (binary classifications of examples). The work in AI identifies other tasks, such as reasoning, as essential for intelligent agents, but those are not supported by the current learning models. The Learning to Reason framework was devised to reconcile inductive learning and efficient reasoning. The framework highlights the fact that new learning questions arise if we want to *learn in order to reason*. In this paper we study learning to reason problems in which the examples seen are only partially specified. These problems have not received much attention in learning theory to date. They become important, however, when learning in order to reason. We first show that the standard "learning to classify" approach to these problems does not necessarily facilitate efficient reasoning, even when the learning is successful. We then develop learning to reason algorithms in the presence of partial information in the interface with the environment. We also prove an impossibility result on learning to reason, for cases in which the class of queries asked are too expressive, and discuss the tradeoffs between the strength of the oracles used and that of the positive results that can be achieved.

*Research supported by Center for Intelligent Control Systems under ARO contract DAAL03-92-G-0115.

†Research supported by NSF grant CCR-92-00884 and by DARPA AFOSR-F4962-92-J-0466.

1 Introduction

The study of learning is motivated by the belief that a learning component must have a central role in any system capable of performing high level cognitive tasks. The current emphasis of the research in learning theory is on the study of inductive learning (from examples) of concepts (binary classifications of examples). Since in this framework the performance of the learner is measured by the correctness of classifying future examples, we call this the *Learning to Classify* approach. The theoretical research in this direction, originating from the seminal work of Valiant [Val84] has already proven useful in that it had contributed to our understanding of some of the main characteristics of the learning phenomenon and has contributed to applied research on classification tasks [DSS93].

However, the goal of developing an understanding of how learning supports other high level cognitive tasks is still not within reach. Consider, for example "high level" cognitive tasks such as language understanding, high level vision and planning, which are widely interpreted as tasks that rely on performing some sort of *inference*. A basic inference task considered in this context is that of *deductive inference*, which is usually modeled as follows: given a Boolean function W, represented as a conjunction of rules and assumed to capture our knowledge of the world, and a Boolean function α, a query that is supposed to capture the situation at hand, decide whether W logically implies α (denoted $W \models \alpha$).

The generally accepted framework for the study of reasoning in intelligent systems is the *knowledge-based systems approach*. The idea is to store the knowledge in some *representation language* with a well defined meaning assigned to its sentences. The sentences are stored in a Knowledge Base (KB) which is combined with a reasoning mechanism that can be used to determine what can be inferred from the sentences in the KB. Given a logical knowledge base, for example, reasoning can be abstracted as the deduction task mentioned above.

While the central role of learning in cognition is acknowledged by many, most lines of research nevertheless study reasoning phenomena separately from learning phenomena. The assumption is that "Learning

can be added later". That is, the separate study of reasoning and learning phenomena will eventually be combined to produce intelligent behavior (see discussion in [Kir91]).
However, as argued in [KR94a], this is not the case. Computational considerations show that the self contained approach to the study of reasoning is inadequate for common sense reasoning. This is true not only for the task of deduction, but for practically all the formalisms developed for reasoning within the knowledge based systems approach [Sel90, KS91, Pap91, Rot93]. Moreover, in the current frameworks, learning and reasoning algorithms cannot be combined in a straightforward way. For example, even if we could *learn* the set of rules W, we cannot use the outcome to efficiently perform deductive reasoning with the query α. This task requires solving satisfiability for $W \wedge \overline{\alpha}$, a task that is believed to be computationally intractable. Other examples for this phenomenon, which do not rely directly on the computational hardness of reasoning, can also be given.

The Learning to Reason framework was introduced in [KR94a] to address the problems mentioned above. On one hand, inspired by the pac-model of learning, it argues that a central question to consider is how the intelligent system acquires its knowledge and how this process, of interaction with its environment, influences the performance of the reasoning system. On the other hand, it highlights the fact that new learning questions arise, and should be addressed if we want to *learn in order to reason*.
In the study presented in this paper we concentrate on one reasoning task, deductive reasoning. Other reasoning tasks can be studied (e.g., [KR95, Rot95, KR94b]) and this paper should be seen as a precursor showing that other tasks should be pursued.
The Learning (in order) to Reason approach combines the interfaces to the world used by known learning models with the reasoning task and a performance criterion suitable for it. In this framework the intelligent agent is given access to her favorite learning interface, and is also given a grace period in which she can interact with this interface and construct her representation KB of the world W. Her reasoning performance is measured only after this period, when she is presented with queries α from some function class, relevant to the world, and has to answer whether W implies α. In [KR94a] (and also in this paper) it is shown that a learning algorithm that is aimed at providing good classification behavior does not, in general, facilitate reasoning tasks. The results there also validate the claim that there is a need to study *Learning in order to Reason*. Namely, learning process whose product can guarantee good reasoning performance rather than good classification performance. In particular, the following, somewhat surprising results, are proved in [KR94a]:

- There are classes of Boolean functions for which the deduction process, in its current formalization, is computationally hard, but for which there is an efficient Learning to Reason algorithm.
- There are classes of Boolean functions (e.g., DNF) for which no efficient solution for the Learning to Classify problem is known, but it is possible to efficiently Learn to Reason with these functions.

These results show that neither a traditional reasoning algorithm (from the CNF representation) nor a traditional learning algorithm (that can "classify" the world) is necessary for Learning to Reason.

In this paper we extend the Learning to Reason framework to study the case in which the interaction with environment is via partial observations. These interactions are more realistic, but received little attention in learning theory mainly since they become more important when learning in order to reason. (Valiant's paper [Val84] and some more recent papers discussed in the next section are notable exceptions.) When learning to reason one cannot assume that examples include an assignment of values to *all* the attributes in the world (as is reasonable in many learning to classify tasks, e.g., character recognition). Rather, the information perceived provides only partial information on the state of the world. For example, when sitting in a windowless lecture hall one's senses do not supply any information about the weather at that moment. Clearly, some, but possibly not all of the missing information may be relevant to the task at hand.
We concentrate on partial information in the learning phase. Namely, the learner is exposed to partial information in the form of examples which are not totally specified. The performance on deductive tasks is our measure of success. We discuss a few ways in which partial examples can be interpreted, and relate our learning results to these interpretations.
We first show that the standard "learning to classify" approach does not facilitate efficient reasoning, even when the learning is successful. We then develop learning to reason algorithms in the presence of partial information in the interface with the environment.
Our positive results are based on keeping a collection of partial examples as a knowledge base which is then used for reasoning. This raises combinatorial questions, as to how much information such a collection contains, which may be of independent interest. We show that such knowledge bases are useful for the deduction tasks considered here, and that they can be learned.
We also prove an impossibility result on learning to reason, for cases in which the class of queries asked are too expressive, and discuss the tradeoffs between the strength of the oracles used and that of the positive results that can be achieved.
We note that partial information may also exist in the form of partially specified questions. This issue is very related to an important phenomenon studied in reasoning, that of "non-monotonic reasoning" [Rei87]. While we do not address this problem here, a similar approach is developed in [Val94, Rot95] to study the non-monotonicity of reasoning.

The rest of the paper is organized as follows. In Section 2 we discuss the meaning of partial observations, and Section 3 formally defines the model. In Section 4 we discuss learning to classify when examples are partially specified, and motivate the discussion in the rest of the paper. In Section 5 we study the problem of reasoning from a knowledge base that consists of partial examples and in Section 6 we discuss the learning to reason problem. In Section 7 we conclude and suggest directions for future work.

2 Partial Observations

We consider a set $X = \{x_1, \ldots, x_n\}$ of Boolean variables, each of which can take the values 1 or 0 to indicate whether the associated world's attribute is true or false. Assignments in $\{0,1\}^n$ are denoted by x, y, z. We model the world as a Boolean function $W : \{0,1\}^n \to \{0,1\}$. The intention is that $W(x) = 1$ if and only if x corresponds to a combination of features which is possible in the world (in other words: x is a "possible world"). A *vector* (partial example, partial assignment) assigns to each of the n variables a value from $\{0,1,*\}$. The symbol $*$ denotes that the value of a variable is *unknown*. For example, $v = (1,*,0)$ means that x_1 is true, x_3 is false, and the value of x_2 is unknown. A vector is *total* (total example, total assignment) if the value of every variable is *known* (i.e., assigned value from $\{0,1\}^n$).
Consider a positive example $v \in \{0,1,*\}^n$ which is only partially specified. There are various ways to interpret the meaning that v conveys on W.

1. Universal interpretation: For all possible extensions of v to total vectors v', $W(v') = 1$.
2. Existential interpretation: There exists an extension of v to a total vector v', such that $W(v') = 1$.
3. Abbreviated interpretation: The partial vector v is just a short way to write the total vector v' defined by $v'_i = v_i$, for all i such that $v_i = 1$, and $v'_i = 0$, otherwise. Thus $W(v) = 1$ means that $W(v') = 1$, for the v' defined above.

The approach (1) is taken in [Val84], in the context of concept learning, to model an agent that observes all the attributes that are relevant for the classification of the learned concept. The approach (3) is useful when the total number of attributes n is much larger than the number of positive attributes any one example has, and an example is presented as a list of its positive attributes (E.g., [LC94]). Learning to classify in this model is studied in [Blu92]. For the task of learning a "world" representation in order to reason about it later, it seems that the agnostic approach, the existential interpretation (2), ought to be taken.
A motivating scenario is that of an agent who is wandering around in the world, but can perceive at any instance only a limited number of attributes. The agent has no control on the perceived attributes, nor can she tell if all the "important" attributes have been perceived. This situation can be modeled by having someone "hide" some of the attributes in an example.
In addition to the works mentioned above, other works [BDD93, SG94, HGR94] also discuss partial assignments, mostly in the context of Learning to Classify. The various approaches differ from ours in that they consider a different learning task, but also on the assumptions made on the interface of the agent with the environment. In [BDD93] it is assumed that the agent can select the attributes it perceives. In [HGR94] it is assumed that a helpful teacher blocks all the attributes which are irrelevant for classifying an example with respect to a fixed decision tree. In [SG94], a (probabilistic or arbitrary) "blocking process" hides some of the attributes of a randomly drawn example. This approach is closest to ours in the sense that the learning task is to learn "default rules", for the purpose of reasoning with them later. The reasoning stage, however, is not considered, and presumably is performed by a traditional reasoner, and is thus intractable.
In the following we would assume that one of the above interpretations has been chosen, and is fixed for the entire duration of the learning scenario. In such a case we can evaluate a Boolean function on a partial assignment according to this interpretation. We denote this by using a subscript to mark the interpretation chosen. Namely, f_u (f_e, f_a) means that the function f would be evaluated according to the Universal (Existential, Abbreviated) interpretation.
Next we discuss some properties of the interpretations of partial assignments. Given $y \in \{0,1,*\}^n$ define a corresponding term $t_y = \wedge_{i=1}^n x_i^{y_i}$ (where $x_i^0 = \overline{x_i}$, $x_i^1 = x_i$, $x_i^* = 1$). The following claims are immediate from the definitions:

Claim 1 *Let f_u, f_e be the universal and existential interpretations, respectively, of $f : \{0,1\}^n \to \{0,1\}$. Then*
(i) $f_u(y) = 1$ if and only if $t_y \wedge \overline{f}$ is not satisfiable.
(ii) $f_e(y) = 0$ if and only if $t_y \wedge f$ is not satisfiable.

The claim shows that evaluating f on a partial assignment may be hard. In particular, if we take the existential interpretation, evaluating $f_e(x)$ is co-NP-Complete for f given in a CNF representation.

3 The Learning to Reason Framework

3.1 The Reasoning Task

We consider Boolean functions $f, \alpha : \{0,1\}^n \to \{0,1\}$. An assignment $x \in \{0,1\}^n$ is a *model* (satisfying assignment) of f if $f(x) = 1$. By "f implies α", denoted $f \models \alpha$, we mean that every model of f is also a model of α. Throughout the paper, when no confusion can arise, we identify a Boolean function f with the set of its models, namely $f^{-1}(1)$. Observe that the connective "implies" ($\models$) used between Boolean functions is equivalent to the connective "subset or equal" ($\subseteq$) used for subsets of $\{0,1\}^n$. That is, $f \models g$ if and only if $f \subseteq g$. Let $\mathcal{F}, \mathcal{Q}$ be two propositional languages. (Note that, a propositional expression is just a representation for a Boolean function, and a propositional language is a class of representations for Boolean functions. These terms are used in the rea-

soning and learning literature accordingly, and we use them interchangeably. Namely, $\mathcal{F}$ and $\mathcal{Q}$ are classes of Boolean functions.)

Definition 2 *An algorithm A is an* exact reasoning *algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q})$, if for all $f \in \mathcal{F}$ and $\alpha \in \mathcal{Q}$, when A is presented with input (f, α), A runs in time polynomial in n and the size of f and α, and answers "yes" if and only if $f \models \alpha$.*

Answering the question $f \models \alpha$ is equivalent to solving unsatisfiability for the formula $f \wedge \overline{\alpha}$. Thus, when f is given as a CNF, exact reasoning can be done efficiently only when satisfiability can be solved efficiently (e.g., Horn expressions). We note that, a DNF representation for f, although better on computational grounds, is less favored mainly for comprehensibility reasons. (Since a representation as a set of rules easily translates to a CNF but not to a DNF expression.)
The following class of queries plays an important role in our results:

Definition 3 *The class $\mathcal{Q}_C$ of* common queries *consists of Boolean functions with the following property: Every $\alpha \in \mathcal{Q}_C$ has a CNF representation, in which every clause is either (1) of size $\leq \log n$ or (2) a Horn-clause (contains at most one positive literal) or (3) a k-quasi-Horn clause (at most k positive literals).*

3.2 The Interface

We define an interface to the world, in the spirit of the known learning models, adapted to deal with partial assignments. As mentioned above we would assume that one of the interpretations for partial assignments has been chosen, and will be fixed for the duration of the learning scenario. Since most of the results hold for all interpretations, we would sometimes omit the subscript denoting which interpretation has been chosen. When we write $f(x)$, where f is a Boolean function and $x \in \{0,1,*\}^n$ is a partial assignment, we mean the value of f_e, f_u, or f_a on x, according to the interpretation chosen. We start by presenting the standard oracles:
An *Example Oracle* for a function f, with respect to the probability distribution D over $\{0,1,*\}^n$, denoted $EX_D(f)$, is an oracle that when accessed, returns $(x, f(x))$, where x is drawn at random according to D. A *Membership Query Oracle* for a function f, denoted $PMQ(f)$, is an oracle that when given an input $x \in \{0,1,*\}^n$ returns $f(x)$. Note that this is a stronger oracle than the membership oracle usually used, since it answers all possible queries on total vectors and, in addition, all queries on partial vectors. We denote the standard membership oracle, which answers only on total vectors, by $MQ(f)$. An *Equivalence Query Oracle* for a function f, denoted $EQ(f)$, is an oracle that when given as input a function g, answers "yes" if and only if $f \equiv g$. If it answers "no" it supplies a counterexample, namely, an $x \in \{0,1,*\}^n$ such that $f(x) \neq g(x)$. A counterexample x satisfying $f(x) = 1$ ($f(x) = 0$) is called a positive (negative) counterexample. The next oracles use the reasoning process itself as a source for examples. The first is used in an on-line scenario. The other two were first used in [FP93]:

Definition 4 *A* Reasoning Query Oracle *for a function f and a propositional language $\mathcal{Q}$, denoted $RQ(f, \mathcal{Q})$, is an oracle that when accessed performs the following protocol with a learning agent A. (1) The oracle picks an arbitrary query $\alpha \in \mathcal{Q}$ and returns it to A. (2) The agent A answers "yes" or "no" according to her belief with regard to the truth of the statement $f \models \alpha$. (3) If A's answer is correct then the oracle says "Correct". If the answer is wrong the oracle answers "Wrong". We call the oracle a* Reasoning Query Oracle with Counterexamples, *(denoted $RQC(f, \mathcal{Q})$) if, when $f \not\models \alpha$ and a reasoning mistake is made, it supplies a counterexample (i.e., $x \in f \setminus \alpha$ where $x \in \{0,1,*\}^n$).*

Definition 5 *An* Entailment Membership Query Oracle *for a function f, denoted $EnMQ(f, \mathcal{Q})$, is an oracle that when given as input a function $g \in \mathcal{Q}$ answers "Yes" if $f \models g$ and "No" otherwise.*

Definition 6 *An* Entailment Equivalence Query Oracle *for a function f, denoted $EnEQ(f, \mathcal{Q})$, is an oracle that when given as input a function g, answers "yes" if and only if $f \equiv g$. If it answers "no" it supplies either a negative counterexample, namely, a function $h \in \mathcal{Q}$ such that $f \not\models h$ but $g \models h$, or a positive counterexample $h \in \mathcal{Q}$ such that $f \models h$ but $g \not\models h$.*

A similar definition can be given for an "entailment example oracle" which draws a query according to some probability distribution, and provides the right classification for it [GS92].

3.3 Learning to Classify

We now define the standard learning to classify problem, in the pac model. To avoid cumbersome notation we assume from now on that all the functions discussed can be represented, in the corresponding representation class, with size polynomial in n (the number of variables), for some fixed polynomial.
Let $I(f)$ denote any subset of the oracles defined above, excluding the RQ oracle[1].

Definition 7 *An algorithm A is a* Probably Approximately Correct Learn to Classify (PAC-L2C) *algorithm for the class of functions $\mathcal{F}$, if there exists a polynomial $p(,,)$ such that for any probability distribution D over $\{0,1,*\}^n$, for all $f \in \mathcal{F}$, on input ϵ, δ, given access to $I(f)$, A runs in time $p(n, 1/\epsilon, 1/\delta)$ and then with probability at least $1-\delta$, outputs a hypothesis h such $Prob_D[h(x) \neq f(x)] < \epsilon$.*

3.4 Learning to Reason

As in the known learning models we distinguish between Learning to Reason in a "batch" type scenario, and "on-line" Learning to Reason.

[1] We could add any other oracle to $I(f)$, e.g., noisy oracles, as long as it represents a reasonable interaction of the agent with the environment. The reason for excluding RQ is to prevent cumbersome notations in the on-line learning to reason definition.

Definition 8 *An algorithm A is an* Exact Learn to Reason (E-L2R) *algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q})$, if there exists a polynomial $p()$ such that for all $f \in \mathcal{F}$, given access to $I(f)$, A runs in time $p(n)$ and then, when presented with any query $\alpha \in \mathcal{Q}$, A runs in time $p(n)$, does not access $I(f)$, and answers "yes" if and only if $f \models \alpha$.*

When a probability distribution governs the occurrence of instances in the world we somewhat relax the requirements using the following restriction: The query α is called (W, ϵ)-*fair* with respect to D if either (1) $W \models \alpha$, or (2) $Prob_D\{x \in \{0,1,*\}^n | W(x) = 1, \alpha(x) = 0\} > \epsilon$. The intuition behind this definition is that the algorithm is allowed to err in case $W \not\models \alpha$, but the weight of W outside α is very small. When W and D are clear from the context we will say that α is ϵ-fair.

Definition 9 *An algorithm A is a* Probably Approximately Correct Learn to Reason (PAC-L2R) *algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q})$, if there exists a polynomial $p(,,)$ such that for any probability distribution D over $\{0,1,*\}^n$, for all $f \in \mathcal{F}$, on input ϵ, δ, given access to $I(f)$, A runs in time $p(n, 1/\epsilon, 1/\delta)$ and then with probability at least $1-\delta$, when presented with any (f, ϵ)-fair query $\alpha \in \mathcal{Q}$, A runs in time $p(n, 1/\epsilon, 1/\delta)$, does not access $I(f)$, and answers "yes" if and only if $f \models \alpha$.*

In the batch scenario above we did not allow access to $I(f)$ while in the query answering phase. In the on-line version however, we consider a query α given to the algorithm as if given by the reasoning oracle $RQ(f, \mathcal{Q})$ defined above. Thus, a reasoning error may supply the algorithm with a counterexample which in turn can be used to improve its future reasoning behavior. We allow the L2R algorithm to access $I(f)$ during this update, but *not* while answering a query. In this on-line (or, mistake-bound) scenario, the L2R algorithm is charged one mistake each time the reasoning query is answered incorrectly.

Definition 10 *An algorithm A is a* Mistake Bound Learn to Reason (MB-L2R) *algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q})$, if A interacts with the reasoning oracle $RQ(f, \mathcal{Q})$, and there exists a polynomial $p()$ such that for all $f \in \mathcal{F}$, (1) A runs in time $p(n)$ (on each query) and answers "yes" or "no" according to its belief with regard to the truth of the statement $f \models \alpha$, without accessing $I(f)$, (2) then runs in time $p(n)$ before it is ready for the next query (possibly, with accessing $I(f)$), and (3) for every (arbitrary infinite) sequence of queries, A makes no more than $p(n)$ mistakes.*

4 Learning to Classify with Partial Assignments

As a motivation for our decision to use model-based representation for reasoning, as discussed in the next section, consider first the problem of PAC learning of concepts from examples, which are partial assignments; namely, learning to classify. The concept class is k-CNF, the hidden function is W, and the learner is given access to $EX_D(W)$, where D is a distribution over $\{0,1,*\}^n$.

Valiant [Val84] shows that the elimination algorithm, that starts with all $3^k\binom{n}{k}$ disjunctions, and eliminates disjunctions which are falsified by positive examples, is a PAC-L2C algorithm for the universal interpretation. It is not hard to see that the same holds for the abbreviated interpretation.

For the existential interpretation, where evaluating a CNF on a partial assignment is computationally hard, the above algorithm is not efficient. If, however, in the learning phase we evaluate the hypothesis on partial examples in a "lazy" manner, the algorithm works for the existential interpretation as well.

According to the "lazy" evaluation, $h(v) = 0$ if and only if one of the clauses in h is falsified by v. The algorithm, when given a positive partial example x, evaluates each of the disjunctions on x, and eliminates the disjunction d from the list if and only if $d(x) = 0$. Negative examples are ignored.

Using Claim 1, we observe that if a disjunction d is eliminated then $f \not\models d$. Thus, d is not in any CNF representation of f. Therefore, the CNF expression h satisfies $h \models f$. This implies that if we took a large enough sample, we could get a hypothesis h for which $Pr_D(h \neq f)$ is small[2]. We have: (A similar result, under different assumptions, is proved in [BDD93].)

Theorem 11 *For all partial example interpretations, there exists an algorithm that PAC-L2C the class of k-CNF functions from partial assignments.*

Suppose, however, that the purpose of learning was to perform deductive reasoning with the hypothesis. In this case, given any query α it is intractable to answer $W \models \alpha$. This mismatch between learning and reasoning, hinted upon in the introduction, is further discussed in [KR94a]. As we discuss in the next section, we can avoid this difficulty by using partial assignments as our knowledge representation.

5 Reasoning and Partial Assignments

Using models as a knowledge representation is straightforward when they are total, but becomes more complicated when dealing with partial models. We start with studying implication with respect to partial assignments.

The definition of a model can be extended to $\{0,1,*\}^n$, using the interpretations given in Section 2. A partial assignment $x \in \{0,1,*\}^n$ is a *p-model* of W if and only if $W_p(x) = 1$, for $p \in \{a, e, u\}$, depending on the interpretation we favor. Likewise we define implication with respect to partial assignments: Let α

[2] Note, however that there is a caveat here. While the algorithm described above is polynomial, the outcome h is "not useful" in the following sense: Evaluating h on a partial assignment is computationally hard. Evaluating h using the "lazy" evaluation, on the other hand, is not guaranteed to be correct on partial assignments y such that $f(y) = 0$.

be a Boolean function. We say that W p-implies α ($W \models_p \alpha$) if every p-model of W is also a p-model of α (where $p \in \{a, e, u\}$). As the following theorem shows the connectives $\models$ and $\models_p$ are equivalent. We therefore use $\models$ in the rest of the paper.

Theorem 12 *Let W, α be Boolean functions and $p \in \{a, e, u\}$. Then, $W \models \alpha$ if and only if $W \models_p \alpha$.*

Proof: Assume first that $W \models \alpha$. Let $x \in \{0,1,*\}^n$ such that $W_e(x) = 1$. Then, there exists an extension $x' \in \{0,1\}^n$ of x such that $W(x') = 1$. From the assumption, $\alpha(x') = 1$ and therefore $\alpha_e(x) = 1$. Similarly, let $x \in \{0,1,*\}^n$ such that $W_u(x) = 1$. Then, for all extensions $x' \in \{0,1\}^n$ of x, $W(x') = 1$. Therefore, all those extensions satisfy $\alpha(x') = 1$ and we have that $\alpha_u(x) = 1$. Finally, let $x \in \{0,1,*\}^n$ such that $W_a(x) = 1$. Then, the unique 0-padded extension $x' \in \{0,1\}^n$ of x satisfies $W(x') = 1$. From the assumption, $\alpha(x') = 1$ and therefore $\alpha_a(x) = 1$. We have shown that $W \models_p \alpha$ for all $p \in \{a, e, u\}$.
For the other direction, assume first that $W \models_e \alpha$. Then, given $x \in \{0,1\}^n$ such that $W(x) = 1$, we can treat x as an element of $\{0,1,*\}^n$ and deduce, from the assumption, that $\alpha(x) = 1$. That is, we have that $W \models \alpha$. The same argument holds when we assume that $W \models_u \alpha$. Assume now that we are given that $W \models_a \alpha$. Then, given $x \in \{0,1\}^n$ such that $W(x) = 1$, we define $x' \in \{0,1,*\}^n$ by replacing all the 0 entries in x by $*$'s. By definition, $W_a(x') = 1$ and therefore $\alpha_a(x') = 1$, and we get that $\alpha(x) = 1$, that is, $W \models \alpha$. ■

5.1 Using Partial-Assignment-Based Representations

We first review some results from [KR94b] on reasoning with total model based representations, and then discuss reasoning from partial assignment based representations. Recall that the reasoning task is to decide, given a "world" function f and a query α, whether f implies α. Consider the following model-based algorithm to the problem $f \models \alpha$:

$\mathcal{A}_{MBR}$:

Test Set: A set Γ of assignments.

Test: If there is an element $x \in \Gamma$ which satisfies f, but does not satisfy α, deduce that $f \not\models \alpha$; Otherwise, $f \models \alpha$.

Clearly, (since $f \models \alpha$ iff every model of f is also a model of α) this approach solves the inference problem if Γ is the set of *all* models of f. A model-based approach is feasible if there exists a small set Γ of models which supports correct reasoning. Such a result is proved in [KR94b] for the set $\mathcal{Q}_C$ of common queries, defined above.

Theorem 13 ([KR94b]) *For any knowledge base f there exists a set Γ_f of models (the set of characteristic models of f), whose size is polynomially related to the DNF size of f, such that model based reasoning with Γ_f is correct for all common queries, $\alpha \in \mathcal{Q}_C$.*

We next extend this result for representations which consist of partial assignments. For any fixed k, there are $\binom{n}{k}$ subsets of size k of the n variables. Given an element $x \in \{0,1\}^n$ and a subset I of k variables, the projection of x on I is the partial vector v defined by: $v_i = x_i$, for all $x_i \in I$, and $v_i = *$ otherwise. Let Γ_f be the (polynomial size) set of characteristic models discussed above. Projecting all the elements of Γ_f on each of these subsets we get a set of $|\Gamma|\binom{n}{k}$ partial vectors.

Definition 14 *Let Γ_f be the set of characteristic models of f. Then the set of all projections of elements of Γ_f on subsets of size k is denoted by Γ_f^k.*

We now define a version of the model-based reasoning algorithm, that is useful when working with Γ_f^k.

$\mathcal{A}_{LAZY-MBR}$

Test Set: A set Γ of partial satisfying assignments.

Test: Given a CNF query α, if there is an element $x \in \Gamma$ which falsifies one of the clauses in α, deduce that $f \not\models \alpha$; Otherwise, $f \models \alpha$.

Note that the algorithm is slightly different from a normal model-based algorithm $\mathcal{A}_{MBR}$. The latter, when given a query and an assignment, will try to evaluate the query on this assignment to test whether the assignment satisfies the query. Since this may be a hard task for partial assignments, the lazy algorithm only tests for direct falsification of clauses in the query, and otherwise gives up.

Theorem 15 *The lazy model-based reasoning algorithm, when using the set Γ_f^k, is correct for all queries $\alpha \in k$-CNF.*

Proof: Clearly, if $f \models \alpha$, model-based reasoning answers correctly. Assume therefore that $f \not\models \alpha$. Theorem 13 implies that there exists a total model $z \in \Gamma_f$ such that $\alpha(z) = 0$. In particular, since α is a k-CNF, one of its clauses C_z must be falsified by z. That is $C_z(z) = 0$. Now consider the element z' which is the projection of z on the variables in C_z. Clearly z' is in Γ_f^k and $C_z(z') = 0$. So the lazy model-based algorithm will answer "No" due to z', and is therefore correct. ■

The previous theorem shows that given Γ_f we can generate a new model based representation, Γ_f^k, by projecting on all possible subsets of size k of the variables. It is interesting to ask whether the projection retains *all* of the information in Γ_f, so that one can reconstruct it from Γ_f^k. If true, this may enable us to answer a wider class of queries, in case Γ_f originally supported deduction also with queries from outside k-CNF, e.g., all common queries. The following claim shows this is not the case.

Claim 16 *The set Γ cannot be reconstructed from Γ_f^k.*

Proof: Using the projections in the example depicted in the following table it is easy to see that it is impossible to distinguish between $\{\gamma_1, \gamma_2, \gamma_3\}$ and $\{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$.

	x_1x_2	x_1x_3	x_1x_4	x_2x_3	x_2x_4	x_3x_4
$\gamma_1 = 0000$	0 0	0 0	**00**	0 0	0 0	**00**
$\gamma_2 = 0101$	**01**	**00**	0 1	**10**	1 1	0 1
$\gamma_3 = 1110$	1 1	1 1	1 0	1 1	**10**	1 0
$\gamma_4 = 0100$	0 1	0 0	0 0	1 0	1 0	0 0

■

Notice that the additional knowledge that Γ is in fact a set consisting of "characteristic models" (minimal models with respect to some order relation [KR94b]) does not help. Moreover, using Γ_f^k we cannot answer more general queries. These two issues can be illustrated as follows: Consider the function $f = \{\gamma_1\} \cup \{\gamma_2\} \cup \{\gamma_3\}$ whose model-based representation is $\{\gamma_1, \gamma_2, \gamma_3\}$. Using Γ_f^2 we cannot tell whether the query $(x_1 \vee \overline{x_2} \vee x_3 \vee x_4) \in$ 4-CNF is implied from the function or not (since this set of models could also be Γ_g^2, for the function $g = \{\gamma_1\} \cup \{\gamma_2\} \cup \{\gamma_3\} \cup \{\gamma_4\}$). So Γ_f^2 does not support correct reasoning with queries in 4-CNF. Theorem 15 provides a polynomial procedure, as long as the original set Γ_f is of polynomial size and k is a constant. It would be desirable to relax the restriction on k, so that some dependency on n is possible. For example, when Γ_f can answer $\log n$-CNF queries, we wish to find projections of Γ_f that will allow for correct reasoning. Clearly, in order to answer all the queries that are disjunctions of size k we need to have, or be able to generate, all projections of the elements in Γ_f on subsets of this size. We call these sets of variables, on which we project the elements of Γ_f, *windows* of variables. Unfortunately, as we show below, small windows are too restrictive.

We say that a window r_1 covers window r_2 if $r_2 \subseteq r_1$. Notice that in this case, we can answer queries on variables from the window r_2 using the projection over r_1. Let R be a set of windows, and let L_k denote the set of $\binom{n}{k}$ windows of size k.

Theorem 17 *If the largest window in R is smaller than $\sqrt{n}$ then either $|R| \geq n^{k/2}$ or R does not cover all the windows in L_k.*

Proof: If the largest window in R is of size m, then every window in R covers at most $\binom{m}{k}$ windows of size k. Since $m < \sqrt{n}$, the number of windows we need in order to cover all the $\binom{n}{k}$ windows of size k is at least:

$$N = \frac{\binom{n}{k}}{\binom{m}{k}} > \sqrt{n}^k = n^{k/2}.$$

■

6 Learning to Reason with Partial Assignments

As we have seen in previous sections, a CNF formula representation is hard to reason with, while model based representations allow for efficient reasoning. We now discuss algorithms that learn model based representations in order to reason.

6.1 A Sampling Approach

The power of access to the distribution D that governs the occurrences of instances in the world is evident from the next argument. We show that given access to $EX_D(f)$, where D is a probability distribution over $\{0, 1, *\}^n$, a simple algorithm can answer correctly a large set of queries. Let $\mathcal{Q}_E$ be a class of functions that can be evaluated in polynomial time on partial assignments. (Notice that this class depends on the interpretation of the partial assignments.) The algorithm *Sample and Reason* first takes a sample of size $m = \frac{1}{\epsilon}(\ln|\mathcal{Q}_E| + \ln\frac{1}{\delta})$ examples from $EX_D(W)$. Let Γ be the set of positive example sampled. Then, whenever presented with a query, the algorithm uses Γ and performs model-based reasoning to answer the query. Let $\alpha \in \mathcal{Q}_E$ be an ϵ-fair query, and assume that $W \not\models \alpha$, then the probability that no example x in Γ is such that $\alpha(x) = 0$ is at most $(1-\epsilon)^m \leq e^{-(\ln|\mathcal{Q}_E| + \ln\frac{1}{\delta})} \leq \frac{\delta}{|\mathcal{Q}_E|}$. The probability that such an event happens for any query in $\mathcal{Q}_E$ is therefore at most δ. Observing that if $W \models \alpha$ a model-based reasoning algorithm cannot make a mistake on α (since it cannot find a counterexample which does not exist) we get the following theorem:

Theorem 18 *The algorithm* Sample and Reason *is a PAC-L2R algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q}_E)$ for any class $\mathcal{F}$.*

The above algorithm highlights another difference between learning to reason and learning to classify. In the learning to classify approach the above argument yields the result on "Occam" algorithms. Namely, if you take a big enough sample and find a small enough consistent hypothesis then you have learned to classify. Unfortunately, though, the problem of finding a consistent hypothesis is in many cases hard. For the learning to reason approach, the above argument shows that this hard task is not necessary.

6.2 Using the *RQ* Oracle

The sampling result presented above implies that we can learn to reason for arbitrary world functions and ϵ-fair queries. In the full version we discuss the limitation of this result. Next we discuss algorithms that do not have this restriction.

We present two on-line algorithms that can learn to reason with k-CNF queries. The first Algorithm, $\mathcal{A}$, interacts with $RQC(\mathcal{F}, \mathcal{Q})$ (see Def. 4) and answers queries, by performing model-based reasoning. Initially, the model-based representation, G, is empty. Following a reasoning mistake, the counterexample supplied by the RQC oracle is added to G. In some sense, this algorithm can be seen as an on-line version of the sampling algorithm.

Theorem 19 *For the universal and abbreviated interpretations, and for any class $\mathcal{F}$ of Boolean functions, the algorithm $\mathcal{A}$, when given access to $RQC(\mathcal{F}, k\text{-}CNF)$, is a MB-L2R algorithm for the reasoning problem $(\mathcal{F}, k\text{-}CNF)$. The number of mistakes $\mathcal{A}$ makes is bounded by $2^{k+1}\binom{n}{k}$.*

Proof: It is sufficient to keep in G a counterexample for every disjunction d with up to k variables such that $f \not\models d$. (Since $f \not\models d_1 \wedge d_2$ implies that either $f \not\models d_1$ or $f \not\models d_2$.) Observe that the number of these disjunctions is bounded by $2^{k+1}\binom{n}{k}$. Since every counterexample received falsifies a new disjunction this bounds also the number of mistakes made by $\mathcal{A}$. ■

Observe that the above algorithm depends on that the queries are presented as CNF expressions. Otherwise, evaluating the queries on partial assignments might be computationally hard. The same restriction applies for the next theorem, though for a different reason.
The above theorem does not hold when using the existential interpretation. In this case, one must use the lazy evaluation, and therefore it can be that $x \in f \setminus (d_1 \wedge d_2)$, but neither d_1 nor d_2 is falsified by x. Therefore, a counterexample supplied by an adversary may not be useful, and the algorithm may make repeated mistakes on the same query.
The following algorithm, $\mathcal{B}$, interacts with $RQ(\mathcal{F}, \mathcal{Q})$ and finds the counterexamples on its own. This results in a slightly higher mistake bound.
The algorithm $\mathcal{B}$ maintains a model-based representation G, initially empty, and uses the lazy evaluation algorithm $\mathcal{A}_{LAZY-MBR}$ to respond to queries presented by RQ. Let $\alpha = d_1 \wedge d_2 \ldots \wedge d_m$ be the CNF representation of a query supplied by RQ, and assume that the algorithm makes a mistake on α. When $f \not\models \alpha$ and the algorithm responded "yes", the algorithm produces a set of assignments and adds them to G. For each $d_i \in \alpha$, the algorithm produces an assignment z_{d_i} as follows: the jth bit in z_{d_i} is set to 1 if $\overline{x_j} \in d_i$, to 0 if $x_j \in d_i$, and to $*$ otherwise. (For example, if $d = x_1 \vee \overline{x_3}$ then $z_d = (0 * 1)$.) When $f \models \alpha$ and the algorithm responded "no", the algorithm removes from G the assignments that caused the mistake.
It is important to notice that the following theorem is independent of the interpretation of the partial assignments. The reason is that it does not receive any examples from the oracles; those are produced internally by the algorithm. Namely, the oracle used, RQ, does not depend on the interpretation. (This is in contrast to RQC, which returns a counterexample and therefore depends on the interpretation.) A similar phenomenon occurs in Section 6.4, where entailment oracles are discussed.

Theorem 20 *For any class $\mathcal{F}$ of Boolean functions, the algorithm $\mathcal{B}$, when given access to $RQ(\mathcal{F}, k\text{-}CNF)$, is a MB-L2R algorithm for the reasoning problem $(\mathcal{F}, k\text{-}CNF)$. The number of mistakes $\mathcal{B}$ makes is bounded by $m \cdot 2^{k+1}\binom{n}{k}$, where m is the maximal number of clauses in a query presented to $\mathcal{B}$.*

Proof: As in the proof of Theorem 19, it is sufficient to keep in G a counterexample for every disjunction d with up to k variables such that $f \not\models d$.
For every mistake the algorithm makes when $f \not\models \alpha$, there exists $d \in \alpha$ such that $f \not\models d$. Therefore, if z_d falsifies a disjunction d', then $f \not\models d'$. We call such an assignment a "good" counterexample. The way the assignments are generated guarantees that the algorithm will not make a mistake on any query which contains this clause.
For every mistake the algorithm makes when $f \models \alpha$, at least one assignment which falsifies a disjunction in α is removed. By definition, this assignment is not one of the "good" counterexamples added previously. Thus we get the claimed mistake bound. ■

6.3 Using Stronger Oracles

The results of Section 5.1 suggest that the only way to improve the results we have shown so far, and reason with classes of queries that are wider than k-CNF, is to use total-model based representation. On the other hand, since the oracle RQ can insist on supplying short counterexamples, and reconstruction is not possible, this seems to be impossible. In order to learn to reason with more expressive queries, we will have to use stronger oracles, which would help us collect Γ. Recall that by $\mathcal{Q}_C$ we denote the class of all common queries, which includes, in particular, Horn CNF formulas and $\log n$-CNF formulas. To present the result we need the following theorem (which is taken from [KR94a], based on results from [Bsh93]):

Theorem 21 *There is a mistake bound algorithm that when given access to the oracles $RQC(f, \mathcal{Q}_C)$ and $MQ(f)$, which are restricted to use total assignments, outputs Γ_f. The bound on the number of mistakes is polynomial in n and the DNF size of f.*
The algorithm maintains a hypothesis h such that, at any time during its execution, $h \models f$.

Theorem 22 *There is a MB-L2R algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q}_C)$, for any class $\mathcal{F}$ and for all interpretations of partial assignments.*
(1) For the abbreviated and universal interpretations, the algorithm uses the oracles $RQC(f, \mathcal{Q}_C)$ and $MQ(f)$.
(2) For the existential interpretation the algorithm uses the oracles $RQ(f, \mathcal{Q}_C)$ and $PMQ(f)$.
The algorithm is polynomial in the DNF size of f and the size of the queries presented to it.

Proof: We show that the partial assignments oracles available here are as powerful as the total assignments oracles used in Theorem 21. This would imply that we can use the algorithm from Theorem 21 to collect Γ_f and then use model-based reasoning. Theorem 13 implies that the reasoning will be correct.
Clearly, a membership query oracle for partial assignments can answer all membership queries on total assignments. To simulate a reasoning query oracle, we have to find a *total* counterexample.
Let α be a query on which the algorithm makes a mistake (when interacting with RQ). First note that since the hypothesis h of the algorithm satisfies $h \models f$, it must be that $h \models \alpha$ and $f \not\models \alpha$.
We argue by cases, according to the interpretation used. For the universal and abbreviated interpretations, we have access to RQC. That is, whenever a mistake is made, RQC returns a partial assignment $v \in f \setminus \alpha$ as a counterexample. In the abbreviated

case, the unique 0-padded extension is the total counterexample needed. In the universal case, v must have an extension which falsifies a disjunction $d \in \alpha$. Such an extension is easy to find by finding a disjunction in which non of the literals is satisfied by v. (By definition, this extension satisfies f.)
For the existential interpretation, we have access to RQ, and therefore need to generate a total counterexample on our own. As in Theorem 20, we generate the counterexamples using the structure of the query α. Given $\alpha = \wedge d_i$, we generate the partial examples z_{d_i}. At least one of these examples, $v = z_{d_j}$, for some j, is positive for f. That is, the partial vector v is a counterexample. Next we show how to use these partial examples to generate a total counterexample. This is done using PMQ. First, using PMQ we find $v = z_{d_j}$ which is positive for f. Then, we extend v to a total vector which is still a counterexample. This can be done in a greedy manner, one bit at a time, using the oracle PMQ. Let v be the counterexample from above. Then, by definition, there exists some total example which is an extension of v and is positive for f. If there is a positive extension with 0 assigned to some bit, which is currently assigned $*$, (use a PMQ query to test this) then we can assign 0 to it, otherwise we can assign 1 to it. ■

6.4 Using Entailment Queries

In this section we present Learning to Reason results in which the agent uses entailment oracles as the interface with the environment. As in Theorem 20 the results in this section are independent of the interpretation of the partial assignments.
Entailment oracles were used before in [FP93, GS92]. The algorithm presented in [FP93] learns a Horn theory given access to $EnEQ$ oracle and to $EnMQ$ oracle. As argued in [KR94a], this algorithm can be shown to learn the Horn least upper bound of any theory (see there for definitions and a discussion) and can thus yield a Learning to Reason algorithm that can reason with any Horn query. Let $H\text{-}disj$ denote the class of Horn disjunctions. The result in [FP93] can be stated as follows:

Theorem 23 *The algorithm in [FP93], when given access to $EnEQ(f, H\text{-}disj)$ and $EnMQ(f, H\text{-}disj)$, is an E-L2R algorithm for the reasoning problem $(\mathcal{F}, Horn)$, for any class $\mathcal{F}$. The algorithm is polynomial in n and the size of the least upper bound of f in the class of Horn functions (in its Horn representation).*

The following theorem shows that a similar result can be achieved for a larger set of queries. (However, the results in Theorems 23 and 24 are incomparable. Both the complexity of the algorithms and the strength of the oracles used are incomparable.)

Theorem 24 *There is an E-L2R algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q}_C)$, for any class $\mathcal{F}$, when given access to $EnEQ(f, CNF)$ and $EnMQ(f, disjunctions)$. The algorithm is polynomial in n, the size of the DNF representation of f and the size of the queries presented to it.*

Proof: The proof relies on Theorem 21. We show how to use the entailment oracles to simulate the oracles MQ and RQC used there.
Let us first show how using $EnMQ(f, disjunctions)$, we can simulate membership queries. Let v be a total vector presented to MQ, and let $t_v = \wedge_{i=1}^{n} x_i^{v_i}$ and $d_v = \overline{t_v} = \vee_{i=1}^{n} x_i^{1-v_i}$ (where $x_i^1 = x_i$ and $x_i^0 = \overline{x_i}$). To answer MQ on v, we call $EnMQ(f, disjunctions)$ with d_v, and invert the (yes/no) answer received. The answer is correct since the only total assignment not satisfied by d_v is v.
Next we show that using $EnMQ(f, disjunctions)$ one can simulate $PMQ(f)$, in the case of the existential interpretation. (This is used later in the proof). Let v be a partial vector presented to PMQ, and let t_v and d_v be defined as above, where $1 - * = *$ and $x_i^* = 1$. To answer PMQ on v, we call $EnMQ(f, disjunctions)$ with d_v, and invert the (yes/no) answer received. The answer is correct since, for the existential interpretation, $f(v) = 0$ if and only if $f \models d_v$ (Claim 1).
We now show that $EnEQ$ can replace the oracle RQC. When the algorithm tries to access RQC, we instead call $EnEQ(f, CNF)$ with the algorithm's hypothesis h, and receive a counterexample α. As argued in Theorem 22, it must be that $h \models \alpha$, and $f \not\models \alpha$. We present α to the algorithm, as if coming from RQC. The algorithm will make a mistake on α. We then compute a counterexample and return it to the algorithm, along with the "no" response. Computing the counterexample is done exactly in the same manner as in the proof of Theorem 22, where PMQ for the existential interpretation is used. ■

Note that when using $EnMQ$, we used a "strong" version of it, since we allowed ourselves to ask queries that are arbitrary disjunctions. On the other hand we used a "weak" version of $EnEQ$ since it was allowed to return any counterexamples in CNF form.

7 Conclusions

We have argued that the current framework used in the study of reasoning is inadequate for common sense reasoning. In doing so we rely on computational considerations and the observation that in the current framework it is hard to reconcile efficient learning and reasoning.
The *Learning to Reason* framework, developed in [KR94a] and extended here, exhibits the role of inductive learning in achieving efficient reasoning, and the importance of studying reasoning and learning phenomena together. In this paper we concentrated on a single reasoning task, logical deduction, and considered the problem of Learning to Reason when the interaction with the world supplies only partial information. Several natural interpretations of partial assignments were considered, and it was shown that logical deduction retains the same meaning with respect to those. Most reasoning algorithms considered here use models-based-representation, that is, the knowledge base consists of a collection of partial examples

which are used for reasoning. We have shown that such knowledge bases support correct and efficient reasoning, and that they can be learned.

Even when the interface of the agent with its environment allows for only k variables at a time, some interesting positive results can be achieved. In particular, we have shown that it is possible to learn to reason from *unrestricted* worlds, provided that the queries are restricted to k-CNF queries. This should be contrasted with the case in which a world description is given to the reasoner in the usual CNF representation; there, restricting the queries will not facilitate tractable reasoning. On the other hand, we have shown that for the larger class of $\log n$-CNF, using restricted oracles cannot yield positive results. Using stronger oracles, though, one can get the same performance as in the case of total information.

This work highlights the fact that the study of learning can be incorporated in the study of other high level cognitive tasks and can contribute to the understanding of those tasks. Moreover, it shows that many interesting learning problems arise from this approach. The work can be continued in various directions, of which we mention a few. First, one can consider other reasoning tasks in which to study the question of learning to reason. For example, probabilistic reasoning is a major area in AI in which most formalizations turn out to be computationally hard. Working within the Learning to Reason framework might shed some light on this problem and allow for positive results. Another major area in AI is that of developing theories for agents that plan and act in a dynamic environment. Extending the Learning to Reason framework to this domain could contribute to this field.

Acknowledgments

We are grateful to Moti Frances, Wolfgang Maass, Les Valiant and the anonymous referees for their useful comments.

References

[BDD93] S. Ben-David and E. Dichterman. Learning with restricted focus of attention. In *Proc. 6th Annu. Workshop on Comput. Learning Theory*, pages 287–296. ACM Press, New York, NY, 1993.

[Blu92] A. Blum. Learning boolean functions in an infinite attribute space. *Machine Learning*, 9(4):373–386, October 1992.

[Bsh93] N. H. Bshouty. Exact learning via the monotone theory. In *Proceedings of the IEEE Symp. on Foundation of Computer Science*, pages 302–311, Palo Alto, CA., 1993.

[DSS93] H. Druker, R. Schapire, and P. Simard. Improving performance in neural networks using a boosting algorithm. In *Neural Information Processing Systems 5*, pages 42–49. Morgan Kaufmann, 1993.

[FP93] M. Frazier and L. Pitt. Learning from entailment: An application to propositional Horn sentences. In *Proceedings of the International Conference on Machine Learning*. Morgan Kaufmann, 1993.

[GS92] R. Greiner and D. Schuurmans. Learning useful Horn approximations. In *Proceedings of the International Conference on the Principles of Knowledge Representation and Reasoning*, pages 383–392, 1992.

[HGR94] T. Hancock, R. Greiner, and R. B. Rao. Exploiting the absence of irrelevant information. In *AAAI Fall Symposium on Relevance*, pages 178–183, 1994.

[Kir91] D. Kirsh. Foundations of AI: the big issues. *Artificial Intelligence*, 47:3–30, 1991.

[KR94a] R. Khardon and D. Roth. Learning to reason. In *Proceedings of the National Conference on Artificial Intelligence*, pages 682–687, 1994. Full version: Technical Report TR-02-94, Aiken Computation Lab., Harvard University, January 1994.

[KR94b] R. Khardon and D. Roth. Reasoning with models. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1148–1153, 1994. Full version: Technical Report TR-01-94, Aiken Computation Lab., Harvard University, January 1994.

[KR95] R. Khardon and D. Roth. Default-reasoning with models. In *Proceedings of the International Joint Conference of Artificial Intelligence*, August 1995.

[KS91] H. Kautz and B. Selman. Hard problems for simple default logics. *Artificial Intelligence*, 49:243–279, 1991.

[LC94] D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Workshop on Machine Learning*, pages 148–156, 1994.

[Pap91] C. H. Papadimitriou. On selecting a satisfying truth assignment. In *Proc. 32nd Ann. IEEE Symp. on Foundations of Computer Science*, pages 163–169, 1991.

[Rei87] R. Reiter. Nonmonotonic reasoning. In *Annual Reviews of Computer Science*, pages 147–188. 1987.

[Rot93] D. Roth. On the hardness of approximate reasoning. In *Proceedings of the International Joint Conference of Artificial Intelligence*, pages 613–618, August 1993. To Appear in Artificial Intelligence Journal, 1995.

[Rot95] D. Roth. Learning to reason with incomplete information. In *Proceedings of the International Joint Conference of Artificial Intelligence*, August 1995.

[Sel90] B. Selman. *Tractable Default Reasoning*. PhD thesis, Department of Computer Science, University of Toronto, 1990.

[SG94] D. Schuurmans and R. Greiner. Learning default concepts. In *Proceedings of the Tenth Canadian Conference on Artificial Intelligence (CSCSI-94)*, 1994.

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

[Val94] L. G. Valiant. Rationality. Technical Report TR-32-94, Aiken Computation Lab., Harvard University, November 1994.