Learning to Reason

RONI KHARDON AND DAN ROTH

Harvard University, Cambridge, Massachusetts

Abstract. We introduce a new framework for the study of reasoning. The Learning (in order) to Reason approach developed here views learning as an integral part of the inference process, and suggests that learning and reasoning should be studied together.

The Learning to Reason framework 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 its favorite learning interface, and is also given a grace period in which it can interact with this interface and construct a representation KB of the world W. The reasoning performance is measured only after this period, when the agent is presented with queries α from some query language, relevant to the world, and has to answer whether W implies α.

The approach is meant to overcome the main computational difficulties in the traditional treatment of reasoning which stem from its separation from the "world". Since the agent interacts with the world when constructing its knowledge representation it can choose a representation that is useful for the task at hand. Moreover, we can now make explicit the dependence of the reasoning performance on the environment the agent interacts with.

We show how previous results from learning theory and reasoning fit into this framework and illustrate the usefulness of the Learning to Reason approach by exhibiting new results that are not possible in the traditional setting. First, we give Learning to Reason algorithms for classes of propositional languages for which there are no efficient reasoning algorithms, when represented as a traditional (formula-based) knowledge base. Second, we exhibit a Learning to Reason algorithm for a class of propositional languages that is not known to be learnable in the traditional sense.

Categories and Subject Descriptors: I.2.3 [**Artificial Intelligence**]: Deduction and Theorem–*deduction*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; I.2.6 [**Artificial Intelligence**]: Learning–*Knowledge acquisition*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Common sense reasoning, computational learning, knowledge representation, model-based reasoning

---

An earlier version of this paper appeared in *Proceedings of the National Conference on Artificial Intelligence* (*AIII-94*).

The research of R. Khardon was supported by ARO under grant DAAL03-92-G-0115 and by the National Science Foundation (NSF) under grant CCR 95-04436.

The research of D. Roth was supported by NSF grant CCR 92-00884 and by DARPA AFOSR-F4962-92-J-0466.

Authors' present addresses: R. Khardon, Division of Applied Sciences, Harvard University, Cambridge, MA 02138, e-mail: roni@das.harvard.edu; D. Roth, Department of Computer Science, University of Illinois at Urbana-Champaign, 1304 West Springfield Avenue, Urbana, IL 61801, e-mail: danr@cs.winc.edu.

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery (ACM), Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.
© 1997 ACM 0004-5411/97/0900-0697 $03.50

1. *Introduction*

Consider a baby robot, starting out its life. If it were a human being, nature would have provided for the infant a safe environment in which it can spend an initial period of time. In this period, the robot adapts to its environment and learns about the structures, rules, meta-rules, superstitions and other information the environment provides. In the meantime, the environment protects it from fatal events. Only after this "grace period" is the robot expected to have "full functionality" in the environment, but naturally, its performance depends on this environment and reflects the amount of interaction it has had with it.

Computational learning theory, a subfield concerned with modeling and understanding learning phenomena [Valiant 1984], takes a similar view that the performance of the learner should be evaluated relative to the world, and only after a certain learning period. Early theories of intelligent systems, however, have assumed that cognition (namely, computational processes like reasoning, language recognition, object identification and other "high level" cognitive tasks) can be studied separately from learning, or as phrased by Kirsh [1991], that "learning can be added later."

In this paper, we develop a new framework for the study of Reasoning. In contrast to earlier approaches to reasoning, the *Learning to Reason* framework developed here, views learning as an integral part of the process, and suggests that learning and reasoning should be studied together. Thus, the Learning to Reason theory is concerned with studying the entire process of *learning* some knowledge representation and *reasoning* with it.

In this framework, the intelligent agent is given access to its favorite learning interface, and is also given a grace period in which it can interact with this interface and construct a representation KB of the world W. The reasoning performance is measured only after this period, when the agent is presented with queries α from some query language, relevant to the world, and has to answer whether W implies α.

Thus, we do not assume that the knowledge representation describing the "world" is given to the agent. Instead, the agent constructs the knowledge representation while interacting with the world. In this way, the reasoning task is no longer a "stand alone" process, and the agent does not need to reason from a previously defined "general purpose" knowledge representation. The integrated study of learning and inference offers several advantages, since the agent can choose a knowledge representation that facilitates the reasoning task at hand, and since its reasoning performance can be measured relative to the environment it interacts with. Moreover, we take the view that a reasoner need not answer efficiently *all* possible queries, but only those that are "relevant", or "common", in a well-defined sense. This relaxation can be used by the agent in selecting its knowledge representation. In the Learning to Reason framework, the knowledge representation used by the agent is chosen to be efficiently learnable and at the same time to facilitate efficient performance in the reasoning task, relative to the environment.

We prove the usefulness of the Learning to Reason approach by showing that through interaction with the world, the agent truly gains additional reasoning power, over what is possible in the traditional setting. Several results are presented to substantiate this claim, presenting cases where learning to reason

about the world is feasible but either (1) reasoning from a given representation of the world or (2) learning representations of the world do not have efficient solutions.

1.1. Overview of the Paper. We first motivate the ideas captured in the framework by developing a simple sampling approach to reasoning. The sampling approach exemplifies the power gained by giving the agent access to the "world" it is supposed to reason in later. We prove that Learning to Reason is possible for arbitrary worlds and query languages under some technical restriction on the queries asked. Observing several limitations of this simple approach we define the general framework in which the appropriate questions can be studied.

We consider some general questions regarding the relation of the Learning to Reason (L2R) framework to the two existing ones, the traditional reasoning framework and the traditional learning framework (which we call here *Learning to Classify* (*L2C*)). We first show that, when the class of queries is not restricted, L2R implies L2C. As demonstrated by the sampling approach, though, this property does not hold if the class of queries is restricted in some natural way. We then discuss how existing results from learning and reasoning can be used in the new framework. In particular, we investigate the straightforward approach that builds a L2R system by reasoning from the output of a (PAC or mistake bound) L2C algorithm, and exhibit the shortcomings of this approach.

The main technical results of the paper show that the new framework allows for efficient solutions even in cases where the separate learning and reasoning tasks are not tractable. In particular, we consider Learning to Reason algorithms that use a set of models (satisfying assignments) as their knowledge representation. We build on a characterization of reasoning with models developed in Khardon and Roth [1996] that is based on the monotone theory developed by Bshouty [1995] to show:

—Consider the reasoning problem $W \models \alpha$, where W is some CNF formula and α is a log nCNF (i.e., a CNF formula with at most log n literals in each clause). Then, when W has a polynomial size DNF[1] there is an exact and efficient Learning to Reason algorithm for this problem, while the traditional reasoning problem (with a CNF representation as the input) is NP-Hard.

—Consider the reasoning problem $W \models \alpha$, where W is any Boolean formula with a polynomial size DNF and α is a log nCNF. Then, there is an exact and efficient Learning to Reason algorithm for this problem, while the class of Boolean formulas with polynomial size DNF is not known to be learnable in the traditional (Learning to Classify) sense.

We also consider Learning to Reason algorithms that use formulas as their knowledge representation, and exhibit another instance of Learning to Reason in which the traditional reasoning task is computationally hard. In this case, the formula-based knowledge representation does not describe the world exactly but rather an approximation of it. Building on a result on learning (to classify) Horn expressions [Frazier and Pitt 1993], we show:

[1] The DNF representation is not given to the reasoner. Its existence is essential, since the algorithm is polynomial in its size.

—Consider the reasoning problem $W \models \alpha$, where W is any Boolean formula that has a Horn approximation of polynomial size, and α is a Horn expression. Then, there is an exact and efficient Learning to Reason algorithm for this problem, while the problem of learning W exactly is not known to be solvable, and the problem of reasoning from a representation of W is not tractable.

Of course, our algorithms do not solve NP-hard problems. In both cases, the additional reasoning power is gained through the interaction with the world. In the first instance, examples from the world are used to construct the model-based representation. In the second instance, the queries presented by the interface are used to construct the approximation of W. An additional crucial observation, used in two of the results, is that, in order to reason with respect to W, one need not learn W exactly. Instead, it is sufficient to use the least upper bound approximation of W. (The approximation, defined precisely later, is in some sense the function closest to W in the class of queries we reason about.) We show that these approximations are learnable in a form that supports the reasoning task efficiently, and use it to prove the Learning to Reason results.

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. Moreover, the results exemplify the phrase "intelligence is in the eye of the beholder" [Brooks 1991], since our agent seems to behave logically, even though its knowledge representation need not be a logical formula and it does not use any logic or "theorem proving."

To summarize, the new positive results are made possible by a combination of several features. First, we relax the inference problems by restricting the classes of queries considered,[2] while, at the same time, using different knowledge representations (that may not be in the traditional comprehensible form) in which this can be exploited. Second, we represent in our KB the least upper bounds of the "world" function rather than the exact representation. Third, and perhaps conceptually most important, our formal framework for the study of reasoning is different from previous ones since we allow the agent to interact with the world, and can therefore measure its performance relative to the world.

In this paper, we focus on presenting the framework and exhibit the results in the context of the most basic reasoning task, namely deductive reasoning. The framework, however, should be seen in a more general context and can be applied for a variety of tasks. In particular, similar results have been recently developed for other, related, reasoning tasks within this framework.[3] These include nonmonotonic reasoning, Learning to Reason with partial observations, Learning to Act in a dynamic world, and learning of active classifiers.

1.2. Comparison with Related Work. The generally accepted framework for the study of reasoning in intelligent systems is the knowledge-based system approach [McCarthy 1985; Nilsson 1991]. It is assumed that the knowledge is given to the system, stored in some *representation language* with a well-defined meaning assigned to its sentences. The sentences are stored in a Knowledge Base

[2] Notice that restricting the classes of queries considered does not change the intractability of the deduction problem, if the world is represented traditionally, as a CNF formula.

[3] See, for example, Khardon and Roth [1995; 1997], Roth [1995], Khardon [1996], and Greiner et al. [1996].

(KB) which is combined with a reasoning mechanism, used to determine what can be inferred from the sentences in the KB. The question of how this knowledge might be acquired and whether this should influence how the performance of the reasoning system is measured is normally not considered. The intuition behind this approach is based on the following observation:

OBSERVATION 1.2.1. *If there is a learning procedure that can learn an exact description of the world in representation R, and there is a procedure that can reason exactly using R, then there is a complete system that can learn to produce "intelligent behavior" using R.*

We believe that the separate study of learning and the rest of cognition is, at least partly, motivated by the assumption that the converse of the above observation also holds, namely, that if there is a system that can Learn to Reason, then there is a learning procedure that can learn a representation of the world, and a reasoning procedure that can reason with it.

Computational considerations, however, render the traditional self-contained reasoning approach as well as other variants of it not adequate for common-sense reasoning. This is true not only for the task of deduction, but also for many other forms of reasoning that have been developed, partly in order to avoid the computational difficulties in exact deduction and partly to meet some (psychological and other) plausibility requirements. All those were shown to be even harder to compute than the original formulation [Selman 1990; Papadimitriou 1991; Roth 1995]. As a consequence, a lot of recent work in reasoning aims at identifying classes of limited expressiveness, with which one can perform some sort of reasoning efficiently.[4] However, none of these works meet the strong tractability requirements for common-sense reasoning (as described, e.g., in Shastri [1993]), even though, (as argued, e.g., in Doyle and Patil [1991]), the inference deals with limited expressiveness and is sometimes restricted in implausible ways.

Very few works have considered the question of integrating theories of reasoning and learning in any formal way. In fact, results in these two fields are currently in a fairly disconnected state. Perhaps the most important open question in learning theory today is concerned with the learnability of DNF or CNF formulas (the problems are equivalent in the current framework). However, even if one had a positive result for the learnability of these classes, this would be relevant only for classification tasks, and cannot be used for reasoning. The reason is that if the output of the learning algorithm is a CNF expression, then it cannot be used for reasoning, since this problem is computationally hard. From a traditional reasoning point of view, on the other hand, learning a DNF is not considered interesting, since it does not relate easily to a rule based representation. Alternative representations studied in learning theory are also not geared towards supporting the reasoning task, and are thus not directly usable. Other problems that exist in the interface between a learning algorithm and a reasoning algorithm are discussed later in this paper. Some of these problems have been identified previously in Kearns [1992].

[4] See, for example, Levesque and Brachman [1985], Cadoli [1995], Levesque [1992], and Selman [1990].

In this work, therefore, while we build on the framework and some of the results of computational learning theory, we distinguish the traditional learning task which we call here *Learning to Classify* from the new learning task, *Learning to Reason.*

The Learning to Reason approach should also be contrasted with various knowledge compilation studies [Selman and Kautz 1996; Moses and Tennenholtz 1996]. There, a theory (KB) is given to the system designer who is trying to compile it, off line, into a more tractable knowledge representation, to facilitate the answering of future queries. In our approach, a world representation is not given to the agent, but instead, it is assumed that the agent can access the world itself via some reasonable interface and acquire information that, later on, will support query answering correctly and efficiently. It should also be noted that while a key issue in selecting a knowledge representation in the traditional knowledge base paradigm is the comprehensibility of the representation (cf. preferring a CNF over a DNF representation) this is not an issue in the current framework, where a representation is chosen based on its learnability and on whether it facilitates reasoning.

Our work is similar in nature to the Neuroidal model developed by Valiant [1994]. The model developed there provides a more comprehensive approach to cognition, and akin to our approach it views learning as an integral and crucial part of the process. There, the agent reasons from a learned knowledge base, a complex circuit, and thus can be modeled by our framework. Indeed reasoning in the Neuroidal model shares many properties with the Learning to Reason framework. Our approach is different in that in an effort to give a more formal treatment of a reasoner that has learned its knowledge base, we currently restrict our discussion to a fixed, consistent world.

To summarize, the main contributions of this work are:

—Defining a framework that integrates theories of learning and reasoning. The approach developed suggests an "operational" approach to the study of reasoning, that is nevertheless rigorous and amenable to analysis. At the same time, we show that this approach is more expressive and efficiently supports "more reasoning" than traditional approaches.

—From the Machine Learning point of view, we exhibit the limitations of the current approach in supporting reasoning tasks. As a result we suggest a new set of questions, oriented towards "learning a functionality" rather than learning a representation of the world. For the task of logical deduction, we show cases where this is indeed useful. As mentioned above, similar results for other tasks have been recently obtained.[3]

—From the Knowledge Representation and Reasoning point of view, the main contribution is the change of paradigm. We suggest that the reasoner interacts with the world and constructs a suitable knowledge representation. Consequently, the effectiveness of the knowledge representation depends on its learnability and on how well it supports inference. This contrasts with the traditional approach where a major issue is the comprehensibility of the knowledge representation. In addition, in terms of expressiveness, we consider here restrictions on the set of queries the agent may need to respond to, rather than restrictions on the world the agent can reason about. Given that the class of queries is restricted, one need not keep in the KB an exact representation of

the world but rather an approximation of it. As a consequence, we can support efficient reasoning for larger classes of world description functions. A third difference draws on the fact that in this framework the performance of the reasoning process can be measured with respect to the environment in which the agent functions (i.e., learns and reasons). This enables us to consider (distributional) restrictions that are not expressible in terms of the knowledge representation alone, but depend on the environment.

The rest of this paper is organized as follows: In Section 2, we give the necessary background on learning and reasoning. In Section 3, we define the new Learning to Reason framework. In Section 4, we discuss the relation between Learning to Reason and Learning to Classify. In Section 5, we discuss how the traditional learning results and reasoning results can be used to get Learning to Reason algorithms. In Section 6, we discuss how model-based reasoning can be used for Learning to Reason. In Section 7, we consider Learning to Reason with Horn queries. We conclude with some reference to future work.

2. *Preliminaries*

We consider a Boolean function f: $\{0, 1\}^n \rightarrow \{0, 1\}$. The elements in the set $\{x_1, \ldots, x_n\}$ are called variables. Assignments in $\{0, 1\}^n$ are denoted by x, y, z, and $weight(x)$ denotes the number of 1 bits in the assignment x. A literal is either a variable x_i (called a positive literal) or its negation $\overline{x_i}$ (a negative literal). A clause is a disjunction of literals and a CNF formula is a conjunction of clauses. For example, $(x_1 \vee \overline{x_2}) \wedge (x_3 \vee \overline{x_1} \vee x_4)$ is a CNF formula with two clauses. A term is a conjunction of literals and a DNF formula is a disjunction of terms. For example, $(x_1 \wedge \overline{x_2}) \vee (x_3 \wedge \overline{x_1} \wedge x_4)$ is a DNF formula with two terms. A CNF formula is Horn if every clause in it has at most one positive literal. We note that every Boolean function has many possible representations and in particular, both a CNF representation and a DNF representation. The size of CNF and DNF representation are, respectively, the number of clauses and the number of terms in the representation.

An assignment $x \in \{0, 1\}^n$ satisfies f if $f(x) = 1$. Such an assignment x is also called a model of f. By "f implies g", denoted $f \models g$, we mean that every model of f is also a model of g. 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 $f \subseteq g$.

2.1. Reasoning.

A widely accepted framework for reasoning in intelligent systems is the knowledge-based system approach [McCarthy 1985]. Knowledge, in some *representation language* is stored in a *Knowledge Base* (KB) that is combined with a reasoning mechanism. Reasoning is abstracted as a deduction task of determining whether a *query* α, assumed to capture the situation at hand, is implied from KB (denoted KB $\models \alpha$). The discussion in this paper is restricted

to propositional knowledge bases.[5] Let $\mathcal{F}$, $\mathcal{Q}$ be two arbitrary classes of representations for Boolean functions.

Definition 2.1.1. An algorithm A is an *exact reasoning* algorithm for the reasoning problem $(\mathcal{F}, \mathcal{Q})$, if for all $f \in \mathcal{F}$ and for all $\alpha \in \mathcal{Q}$, 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 $f \models \alpha$.

In the definition above the complexity of the algorithm depends on the size of the input, the formulas f and α. For this reason we usually restrict the discussion to a polynomial size representation of the functions considered. In particular, the class $\mathcal{F} = \text{CNF}$ denotes those Boolean functions with a polynomial size CNF, and $\mathcal{F} = \text{CNF} \cap \text{DNF}$ denotes those Boolean functions with a polynomial size CNF, and a polynomial size DNF.

Answering the question $\text{KB} \models \alpha$ is equivalent to solving satisfiability for the formula $\text{KB} \wedge \bar{\alpha}$. This implies that if KB is given as a CNF or α is given as a DNF the problem is NP-Hard. It is also known that if KB is given as a DNF and α is given as a CNF the problem can be solved in polynomial time, but this representation was less favored in the formal study of knowledge representation than the CNF one, possibly because of the belief that the KB should be represented a set of rules, for comprehensibility reasons, and that easily translates to a CNF representation but not to a DNF expression.[6] Thus, when KB is given as a CNF, exact reasoning can be done efficiently only when satisfiability can be solved efficiently (e.g., Horn theories, CNF with clauses of length two).

2.2. The Learning Interface. Computational learning theory [Valiant 1984; Haussler 1987; Angluin 1992] abstracts the problem of inductively learning a concept as the problem of learning a Boolean function, given some access to an oracle that is familiar to some degree with the function. The interpretation is that the function's value is 1 when the input belongs to the target concept and 0 otherwise. The oracles are used to model the type of interface the learner may have to the world and they vary between learning models according to the amount of information the learner is assumed to receive about the concept. We next describe some of the standard oracles that are used in learning theory.

Definition 2.2.1. A *Membership Query Oracle* for a function f, denoted $MQ(f)$, is an oracle that when given an input $x \in \{0, 1\}^n$ returns $f(x)$.

Definition 2.2.2. An *Equivalence Query Oracle* for a function f, denoted $EQ(f)$, is an oracle that when given as input a function g, answer "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.

[5] A propositional expression is just a Boolean function, and a propositional language is a class of Boolean functions. These are the terms used in the reasoning and learning literature respectively, and we use them here interchangeably.

[6] Notice that both CNF and DNF are universal representations and are symmetric as far as the size of the representation; therefore, there is a need to resort to arguments outside the theory of Boolean functions to prefer one over the other.

Definition 2.2.3. An *Example Oracle* for a function f, with respect to the probability distribution D, denoted $EX_D(f)$, is an oracle that when accessed, returns $(x, f(x))$, where x is drawn at random according to D.

The following oracles, introduced in Frazier and Pitt [1993] can be viewed as oracles that use reasoning as part of the interface, and are thus suitable for learning in the current context. In some sense, the situation modeled by these oracles, as well as by the RQ oracle defined later, models a scenario in which an agent is told (or is programmed to have) some facts or rules about the world. This is reminiscent of McCarthy's advice taker [McCarthy 1985] and the "ask" and "tell" operations discussed by Levesque [1984].

Definition 2.2.4. An *Entailment Membership Query Oracle* for a function f, denoted $EnMQ(f)$, is an oracle that when given as input a function g answers "yes" if $f \models g$ and "no" otherwise.

Definition 2.2.5. An *Entailment Equivalence Query Oracle* for a function f, denoted $EnEQ(f)$, 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 either a negative counterexample, namely, a function h such that $f \not\models h$ but $g \models h$, or a positive counterexample h such that $f \models h$ but $g \not\models h$. The function h is restricted to be a Horn disjunction.

Several other oracles have been discussed in the literature. For example, the oracles "incomplete membership queries" [Angluin and Slonim 1994], "faulty example oracles" [Angluin and Laird 1988], "malicious example oracles" [Valiant 1985; Kearns and Li 1988], and "partial assignments' oracles" [Khardon and Roth 1995] were studied and can be used here as well. In the following, any subset of these oracles may be available to the learner.

Definition 2.2.6. We denote by $I(f)$ the *interface* available to the learner when learning f. This can be any subset of the oracles defined above, and might depend on some fixed but arbitrary and unknown distribution D over the instance space $\{0, 1\}^n$.

2.3. Learning to Classify. The standard definitions for Learning to Classify [Valiant 1984; Angluin 1988; Littlestone 1988] distinguish between a "batch" type learning scenario, and an "on-line" scenario. In the "batch" scenario, the learning algorithm interacts with the environment, via $I(f)$, in order to acquire the skill of labeling future instances. The performance of the algorithm is measured only after some "grace period", that must be of length polynomial in the size of the concept learned (and also in the quality of its performance), when it is required to predict the value of f on some other example drawn according to D. The dependence of the algorithm on the size of the concept is not made explicit in the following definitions, assuming that it has some polynomial (in n) representation. We denote by $h(x)$ the prediction[7] of the algorithm on the example $x \in \{0, 1\}^n$.

[7] Sometimes an equivalent definition is used, in which the learning algorithm outputs an hypothesis h and its performance is measured with respect to it. In order not to make any assumptions on how this hypothesis is represented we assume here that it is internal to the algorithm.

Definition 2.3.1. An algorithm A is an *Exact Learn to Classify* (*E-L2C*) algorithm for a class of functions $\mathcal{F}$, if there exists a polynomial $p(\ )$ such that for all $f \in \mathcal{F}$, when given access to $I(f)$, A runs in time $p(n)$ and then, given any $x \in \{0, 1\}^n$, takes time $p(n)$ to predict σ such that $\sigma = f(x)$.

Definition 2.3.2. An algorithm A is a *Probably Approximately Correct Learn to Classify* (*PAC-L2C*) algorithm for a class of functions $\mathcal{F}$, if there exists a polynomial $p(,,)$ such that for all $f \in \mathcal{F}$, for any probability distribution D, on input ϵ, δ and given access to $I(f)$, A runs in time $p(n, 1/\epsilon, 1/\delta)$ and then, given any $x \in \{0, 1\}^n$, predicts $h(x)$ in time $p(n, 1/\epsilon, 1/\delta)$. A's predictions have the property that with probability at least $1 - \delta$, $\Pr_{x \in D}[f(x) \neq h(x)] < \epsilon$. The parameter δ is called the *confidence* of the algorithm A and ϵ its *accuracy.*

In the on-line (or, mistake-bound) scenario [Littlestone 1988], algorithm A is presented with a sequence of examples in $\{0, 1\}^n$. At each stage, the algorithm is asked to predict $f(x)$ and is then told whether its prediction was correct. Each time the learning algorithm makes an incorrect prediction, we charge it one *mistake.*

Definition 2.3.3. An algorithm A is a *Mistake Bound Learn to Classify* (*MB-L2C*) algorithm for a class of functions $\mathcal{F}$, if there exists a polynomial $p(\ )$ such that for all $f \in \mathcal{F}$, for every (arbitrary infinite) sequence of instances, A runs in time $p(n)$ (on each example) and makes no more than $p(n)$ mistakes.

It is known [Angluin 1988; Littlestone 1988] that a Mistake bound learning algorithm can be transformed into a PAC learning algorithm, and that an E-L2C (or PAC-L2C) algorithm that uses the $EQ(f)$ oracle can be transformed into an algorithm that achieves PAC-L2C using an Example Oracle and instead of using $EQ(f)$.

3. *Learning to Reason*

In this section, we motivate and introduce the definitions of the *Learning to Reason* model.

Let $W \in \mathcal{F}$ be a Boolean function that describes the world exactly. Let $\mathcal{Q}$ be the class of queries considered, α be some Boolean function (a query) and let D be some fixed but arbitrary and unknown probability distribution over the instance space $\{0, 1\}^n$. As in the Learning to Classify framework, we assume that D governs the occurrences of instances in the world.

Definition 3.1. The query α is called *legal* if $\alpha \in \mathcal{Q}$.

Definition 3.2. The query α is called (W, ϵ)-*fair* if either $W \subseteq \alpha$ or $\Pr_D[W \setminus \alpha] > \epsilon$.

The intuition behind this definition is that in case that $W \not\models \alpha$ but the weight of W outside α is very small, we may allow the algorithm to err (and answer that $W \models \alpha$). Along with ϵ, the *accuracy* parameter, we use a *confidence* parameter, δ, and sometimes might allow the reasoning algorithm to err, with small probability, less than δ.

3.1. A Sampling Approach. To motivate our approach, we first consider the following simple approach to reasoning: whenever presented with a query α, first

use the Example Oracle $EX_D(W)$ and take a sample of size $m = (1/\epsilon)\ln(1/\delta)$, where δ and ϵ are the required confidence and accuracy parameters. Then, perform the following model-based test: for all the examples $(x, 1)$ sampled from $EX_D(W)$ (note that we ignore the samples labeled 0), check whether $\alpha(x) = 1$. If for some x, $\alpha(x) = 0$ say $W \not\models \alpha$; otherwise say $W \models \alpha$.

The following analysis shows that if α is (W, ϵ)-fair then with probability at least $1 - \delta$ the algorithm is correct.

Clearly, if $W \models \alpha$ the algorithm never errs. The algorithm makes a mistake only if $W \not\models \alpha$ but an instance x in $W \cap \bar{\alpha}$ is never sampled. An instance x, picked randomly according by $EX_D(W)$ is in $W \cap \bar{\alpha}$ with probability greater than ϵ. Therefore, the probability that an instance $x \in W \cap \bar{\alpha}$ is missed in $m = (1/\epsilon)\ln(1/\delta)$ trials is less than

$$(1 - \epsilon)^m = (1 - \epsilon)^{1/\epsilon \ln(1/\delta)} < \delta. \tag{1}$$

Therefore, the algorithm errs on a (W, ϵ)-fair query with probability less than δ.

This analysis depends on the fact that the samples are independent of the query α, and therefore a different sample has to be taken for every query α. We call this a *repeated sampling* approach.[8] However, repeated sampling is not a plausible approach to reasoning in intelligent systems. When presented with a query, an agent cannot allow itself further interactions with the world before answering the query. Especially if the query is "A lion is approaching $\Rightarrow$ I have to run away".

For a more plausible approach, we now modify the above analysis and show that a *one time sampling* approach can also guarantee reasoning with respect to (W, ϵ)-fair queries, with confidence $1 - \delta$.

From Eq. (1), we have that the probability that *any* (W, ϵ)-fair query $\alpha \in \mathcal{Q}$, that is not implied by W is answered by "yes" after m samples is less than

$$|\mathcal{Q}|(1 - \epsilon)^m, \tag{2}$$

where $|\mathcal{Q}|$ is the size of the class $\mathcal{Q}$ of queries. Therefore, by taking more than

$$m = \frac{1}{\epsilon}\left(\ln\left|\mathcal{Q}\right| + \ln\frac{1}{\delta}\right) \tag{3}$$

samples from $EX_D(f)$, we get that the model-based test described above errs on any (W, ϵ)-fair query with probability less than δ. Since all the queries in $\mathcal{Q}$ are propositional formulas of polynomial size, the number m of samples required to guarantee this performance is polynomial. This approach is therefore feasible.

It is important to note that the example oracle suggests a somewhat nonstandard view of the reasoning problem. When using the example oracle we are, in some sense, assuming an "objective"[9] world reflected through the oracle; each positive example corresponds to a combination of features that occurs in the world and is measured by the agent's attributes. The distribution D on the

[8] A similar (though more sophisticated) approach was developed in Kearns [1992] for the case in which both the knowledge base and the queries are learned concepts in the PAC sense. It is implicit there that for each possible query one needs a new sample.

[9] We note that the notions "objective" and "subjective" have been loaded with a variety of ideas in the literature, and our work relates to these only in the narrow sense discussed here.

occurrence of examples is assumed to be independent of the agent, and reflects the natural distribution of scenarios occurring in the world. This is in contrast with the standard "possible worlds approach" (e.g., [Fagin et al. 1995]) where the knowledge of the agent is characterized using formulas and the possible worlds are exactly the assignments that agree with this knowledge (regardless of whether they can actually occur in the "real world"), and can thus be thought of as reflecting the "subjective" view of the agent. A similar contrast holds with works that considered distributions over possible worlds as a tool for formalizing the notion of belief [Bacchus et al. 1993; 1997]. There, some form of uniform distribution over all the "subjective" possible worlds is used in order to measure the plausibility of a statement (corresponding to the belief in it). The corresponding computational problems (e.g., Nilsson [1986] and Henrion [1988]) when given as input a formula W, involve sampling from $\{0, 1\}^n$, typically according to the uniform distribution, in order to estimate the support for a given belief. This process, however, is intractable in a very strong sense [Roth 1996]. The computational advantage in our approach is due to the fact that we have access to the distribution over the "objective" world. That is, we observe and take into account only instances that actually occur in the world with some non-negligible probability. We note that other oracles we use later in the paper are altogether oblivious to this issue. That is, they are consistent with both the "objective" and the "subjective" views. The following example illustrates these notions and the sampling approach:

The Seminar Room World. Consider an agent-camera at the ceiling of a seminar room in some academic institute. Further assume that there are 5 people of interest to the agent, 10 chairs in the room, and that it can observe which of these people is in the room and on which chair they are sitting. The camera can also observe several other attributes, like whether the light is on, whether the overhead projector is used, etc. All these together constitute the set of measured variables $x_1, \ldots, x_n$.

Various types of seminars take place in this department, and people attend seminars of interest to them. This induces a set of possible combinations of attribute values that the camera might ever observe. This set of possibilities is the set of possible worlds, and the "world" refers to the general scenario of the seminar room. Notice that in this situation several general rules may apply. For example, the rule "if the overhead projector is used then the light is off" may hold. Other restrictions may arise from rules regarding the behaviors of the people involved like, "person A attends all the seminars which person B attends". The task of reasoning, discussed above, requires that the camera will be able to assess the validity of assertions of this kind, given the information it has on the world. The sampling result implies that this can be done reliably and efficiently by observing previous instances, and deciding accordingly. On the other hand, assessing the validity of a rule without observing the world through the camera may force us to go through (or sample) all the combinations of the attributes, even those that may never occur in the world, and evaluate the rule and the "theory" of the world on them.

In summary, the sampling approach exemplifies the power gained by giving the reasoner access to the "world" it is supposed to reason in later. Learning to Reason is possible for arbitrary world and query languages given that the queries

are fair. However, the one-time sampling approach is not the ultimate solution for reasoning and there are several reasons not to be satisfied with this approach as the sole solution for the reasoning problem. We briefly discuss some of these reasons as a motivation for the approach taken in the rest of the paper.

Exact reasoning: In many cases, it is natural to require the reasoning to be *exact* with respect to the world. Unlike the sampling approach other representations might support it.

Active learning: Certain events we want to reason about might be too rare to have a significant weight under D, the distribution that governs the occurrences of instances in the world (i.e., they might not be "fair"). To facilitate this, we might want to supply the reasoner with a richer class of interfaces with the world. Oracles that seem to be relevant in this context are partial assignments oracles [Valiant 1985] more selective oracles such as Amsterdam's [1988] experiment oracle and Valiant's [1995] actions in the rationality model, and perhaps the use of declarative information. The availability and plausibility of various oracles as well as the technical investigation into algorithms and knowledge representations that can make use of these oracles are the main questions here. Preliminary progress in this direction is done in Khardon and Roth [1995].

Convergence: Reasoning with an inductively learned knowledge base yields a desirable nonmonotonic behavior: every time the algorithm makes a reasoning mistake, it changes its mind (i.e., the underlying learning algorithm does), learns something about the world, and would not make the same type of mistake again. More importantly, unlike reasoning from a large set of randomly selected examples, the algorithms we present later in this paper for reasoning from an inductively learned knowledge base, have the property that while exhibiting a nonmonotonic behavior, they also converge (after polynomially many reasoning mistakes) to yield an exact reasoning behavior. We would like to investigate which algorithms and knowledge base representations yield this behavior.

Space consideration: The one-time sampling approach suggested above might require maintaining a very large (though polynomial) knowledge base, and accordingly, a long period of interaction with the world, before it can guarantee acceptable performance. There might be other, much smaller representations that perform as well.

3.2. Learning to Reason: Definitions. We now provide the definitions that allow us to formally study the questions discussed above. As pointed out earlier, our definitions do not allow a *repeated sampling* approach, but do allow *one-time sampling.*

As in the case of Learning to Classify we distinguish between Learning to Reason in a "batch" type scenario, and an "on-line" Learning to Reason. In the batch scenario, the algorithm interacts with the environment, via $I(f)$, in order to acquire the *reasoning skill* of answering future queries. The performance of the algorithm is measured only after some "grace period", when it is required to decide, without interacting the world again, if some query is entailed by f. The "grace period" must be of length polynomial in the size of the world description

and in the quality of its performance. As before, we define an exact version and a nonexact version.

Definition 3.2.1. 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 $f \models \alpha$.

Definition 3.2.2. 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 all $f \in \mathcal{F}$, for any probability distribution D, on input ϵ, δ, and 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 oracle. Thus, a reasoning error may supply the algorithm 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. The following oracle is used to model the learning interface for the on-line scenario.

Definition 3.2.3. A *Reasoning Query Oracle* for a function f and a query 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 its 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" and in case $f \not\models \alpha$ it also supplies a counterexample (i.e., $x \in f \setminus \alpha$).

When learning, the algorithm is charged one mistake each time the reasoning query is answered incorrectly, and a successful learner should make a small number of mistakes.

Definition 3.2.4. 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.

We note that the seminar world example presented earlier in the context of the sampling approach can be extended to cover a variety of learning scenarios captured by the above definitions. For example, the MQ oracle corresponds to the situation where the agent constructs a set of feature values and asks whether it represents a "valid" seminar situation. The RQ oracle corresponds to someone (perhaps the agent itself) teaching the agent about the seminar world by

presenting questions to the agent. The answers are then confirmed or disproved in "real" seminars. These two oracles can be used together as the interface $I(f)$, and as we demonstrate later (see Theorem 6.5.1.1), the interface is sufficient to guarantee successful learning as in Definition 3.2.1.

4. *The Relation between L2R and L2C*

In this section, we investigate the relation between Learning to Reason and Learning to Classify. Clearly, given an Exact-L2C algorithm A, the ability to reason efficiently with the hypothesis that A keeps is sufficient to yield an efficient Exact-L2R algorithm. In this section, we consider the other direction of this relation. That is, given an algorithm that can Learn to Reason, is it necessarily the case that there is an algorithm that can Learn to Classify?

Intuitively, the classification task seems to be easier than the reasoning task. In the former we need to evaluate correctly a function on a single point, while in the latter we need to know if *all the models* of the function are also models of the query. It is not surprising therefore, that if *any* subset of $\{0, 1\}^n$ is a legal query, the ability to L2R implies the ability to L2C. We formalize this in the following theorem.

Let DISJ be the class of all disjunctions over n variables.

THEOREM 4.1. *If there is an Exact-L2R algorithm for the reasoning problem* ($\mathcal{F}$, *DISJ*), *then there is an Exact-L2C algorithm for the class* $\mathcal{F}$.

PROOF. Observe that for a Boolean function f and an assignment $z \in \{0, 1\}^n$,

$$f(z) = 0 \Leftrightarrow f \models \{0, 1\}^n \setminus \{z\}. \tag{4}$$

Denote by d_z the disjunction define by $d_z = \bigvee x_i^{z_i}$ (where $x_i^0 = x_i$, $x_i^1 = \overline{x_i}$). For example, if $z = (1, 0, 0)$, then $d_z = \overline{x_1} \vee x_2 \vee x_3$. Clearly d_z satisfies $d_z(y) = 0$ if and only if $y = z$, and therefore Eq. (4) can be written as

$$f(z) = 0 \Leftrightarrow f \models d_z. \tag{5}$$

Now, given an Exact-L2R algorithm A for the reasoning problem ($\mathcal{F}$, DISJ) we use it to construct an Exact-L2C algorithm B for $\mathcal{F}$ as follows: B first runs A (enough time to guarantee A's performance), and when given an assignment $z \in \{0, 1\}^n$ for classification by f, it first computes the disjunction d_z and gives it to A. B returns "1" if and only if A returns "no" on d_z. The correctness of the algorithm is clear from Eq. (5). □

We note that reasoning with DISJ is as hard as reasoning with general CNF queries, since in general $f \models (\alpha \wedge \beta)$ if and only if $f \models \alpha$ and $f \models \beta$.

The proof of the above theorem does not go through if the class of queries $\mathcal{Q}$ does not include all of DISJ, and does not hold also in the non-exact L2R formulation as exemplified by the sampling result. Strictly speaking, we leave open the question of whether there is a nontrivial case, in which there exists an exact Learning to Reason algorithm for the problem ($\mathcal{F}$, $\mathcal{Q}$) but there is no Learning to Classify algorithm for $\mathcal{F}$. The reason is that in general, it is hard to prove that there *does not exist* a L2C algorithm, when there are no restrictions on the output representation of the learning algorithm. We later show, however,

that there are interesting classes $\mathcal{F}$ and $\mathcal{Q}$ of Boolean functions, such that there exists a Learning to Reason algorithm for the problem $(\mathcal{F}, \mathcal{Q})$ but *it is not known* how to Learn to Classify $\mathcal{F}$.

5. *L2R by Combining Learning and Reasoning*

In this section, we consider straightforward combinations of learning and reasoning algorithms. Namely, we consider using the output of an existing learning (to classify) algorithm for Learning to Reason. We show that under some conditions this can be done successfully. However, as we discuss below, the significance of the results presented in this section is that they exhibit the *limitations* of L2R by combining separate reasoning and learning algorithms.

The case of exact learning algorithms that, in addition, produce as output a representation that allows for efficient reasoning has already been discussed above. Here we consider the relaxation of these requirements.

5.1. Learning to Reason via PAC Learning.

Assume that the world description W is in $\mathcal{F}$ and there is a PAC-L2C algorithm A for $\mathcal{F}$. We first show how a PAC learning algorithm, if it has an additional property, can be combined with a reasoning algorithm to yield a PAC-Learn to Reason algorithm.

Definition 5.1.1. An algorithm that PAC Learns to Classify $\mathcal{F}$ is said to *learn* $f \in \mathcal{F}$ *from below* if, when learning f, the algorithm never makes mistakes on instances outside of f. (I.e., if h is the hypothesis the algorithm keeps then it satisfies $h \subseteq f$.)

Theorem 5.1.2. *Let A be a PAC-Learn to Classify algorithm for the function class $\mathcal{F}$ and assume that A uses the class of representations $\mathcal{H}$ as its hypotheses. Then, if A learns $\mathcal{F}$ from below, and there is an exact reasoning algorithm B for the reasoning problem $(\mathcal{H}, \mathcal{Q})$, then there is a PAC-Learn to Reason algorithm C for the reasoning problem $(\mathcal{F}, \mathcal{Q})$.*

Proof. The learning algorithm C simply runs A and then uses B in order to reason with the hypothesis $h \in \mathcal{H}$ of A. Let α be a (W, ϵ)-fair legal query. Assume first that $W \models \alpha$. Then, since the algorithm A learns W from below its hypothesis h satisfies $h \subseteq W \subseteq \alpha$ and therefore $h \models \alpha$ and C answers correctly. When $W \not\models \alpha$, since α is (W, ϵ)-fair, we know that $\Pr_D[W \setminus \alpha] > \epsilon$. Together with the fact that $\Pr_D[W \setminus h] < \epsilon$, we have $h \setminus \alpha \neq \emptyset$ and therefore $h \not\models \alpha$, so again, C answers correctly. □

The performance guaranteed by the above theorem is no better than the one-time sampling approach, while using a possibly more complicated algorithm. The result is significant, however, for the following reasons; (1) It shows the limitations of L2R by combining reasoning and learning algorithms: the theorem cannot be extended to the case where the learning algorithm has also error from above (i.e., where it makes prediction mistakes also on instances of W). The reason is that if $h \setminus W \neq \emptyset$, then it might be the case that $W \models \alpha$ but $h \not\models \alpha$ exactly because of those additional assignments in h. (2) The theorem allows for the PAC learning algorithms to use any set of oracles (and in particular, other, more plausible interfaces to specific applications) while the sampling result holds only for the example oracle. (3) The theorem allows for a system that performs

classification as well as reasoning from the same knowledge base, and finally, (4) The theorem is useful in explaining the behavior of mistake bound algorithms discussed below.

The above theorem limits the type of errors the algorithm may make. For a similar result to hold for an algorithm with two sided error one would have to use a somewhat strong version of an "approximate deduction", where the reasoning algorithm answer "yes" if and only if $\Pr_D[W \setminus \alpha] < \epsilon$ (rather than $\Pr_D[W \setminus \alpha] = 0$ in exact deduction). Notice that the reasoning algorithm must be wrong on all queries which are not ϵ-fair. Under this restriction, the hypothesis h of the learning algorithm can be used to reason with 2ϵ-fair queries.

5.2. Learning to Reason via Mistake Bound Learning. Consider a Mistake Bound Learn to Classify algorithm that keeps a hypothesis that allows for efficient reasoning. We observe that such an algorithm can be used to construct a Learning to Reason algorithm. Furthermore, the type of reasoning exhibited by this algorithm is reminiscent of the phenomenon of nonmonotonic reasoning.

Let A be a Mistake Bound algorithm and assume it has been used long enough to guarantee PAC performance [Littlestone 1988]. In the case it has used up all of its mistakes on negative examples (i.e., on assignments outside of W), the hypothesis it uses is a "learn from below" hypothesis and, using Theorem 5.1.2, we can reason with it and succeed on all (W, ϵ)-fair queries.

Unfortunately, we cannot force the algorithm (or rather the interface) to make all these mistakes within the grace period. If we use an initial grace period to ensure its PAC properties, then after the algorithm is ready to answer queries it may still make (a limited number of) mistakes. If the reasoning mistakes can be used as a source for negative counterexamples (i.e., assignments in $h \setminus W$), then after making a polynomial number of reasoning mistakes the algorithm would hold a hypothesis that approximates W from below, and thus support exact reasoning. This is the motivation behind the definition for the oracle $RQ(W, \mathbb{2})$ (Definition 3.2.3), where a similar behavior is described.

It is interesting to note that reasoning with this type of an algorithm yields a nonmonotonic reasoning behavior. Every time the algorithm makes a reasoning mistake, it changes its mind, learns something about the world, and would not make the same mistake again. This is an inherent feature of the Learning to Reason approach, and it captures some of the features of nonmonotonic reasoning that are otherwise hard to formalize.

6. *Learning to Reason via Model-Based Reasoning*

In this section, we develop the main technical results of this paper and exhibit the computational advantages of the Learning to Reason approach. We deviate from the traditional setting of "first learn to classify, then reason with the hypothesis": A learning algorithm is used first, but rather than learning a "classifying hypothesis", it constructs a knowledge representation that allows for efficient reasoning.

We utilize two recent results, one on learning via monotone theory [Bshouty 1995] and the other on reasoning with models [Khardon and Roth 1996] to show that this process yields two interesting outcomes that were not possible in the traditional setting. First, we give a L2R algorithm for the class CNF $\cap$ DNF for

which there are no efficient reasoning algorithms, when represented as a traditional (formula-based) knowledge base. Second, we develop a L2R algorithm for the class of functions with polynomial size DNF, for which a L2C algorithm is not known.

6.1. MONOTONE THEORY. The monotone theory was introduced by Bshouty [1995]. We briefly describe the necessary details; a longer description can also be found in Khardon and Roth [1996].

Definition 6.1.1 (*Order*). We denote by $\leq$ the *usual partial order* on $\{0, 1\}^n$, the one induced by the order $0 < 1$. That is, for $x, y \in \{0, 1\}^n$, $x \leq y$ if and only if $\forall i$, $x_i \leq y_i$. For an assignment $b \in \{0, 1\}^n$, we define $x \leq_b y$ if and only if $x \oplus b \leq y \oplus b$ (Here $\oplus$ is the bitwise addition modulo 2).

Intuitively, if $b_i = 0$, then the order relation on the ith bit is the normal order; if $b_i = 1$, the order relation is reversed and we have that $1 <_{b_i} 0$. We now define:
The *monotone extension of* $z \in \{0, 1\}^n$ with respect to b:

$$\mathcal{M}_b(z) = \{x \mid x \geq_b z\}.$$

The *monotone extension of* f with respect to b:

$$\mathcal{M}_b(f) = \{x \mid x \geq_b z, \text{ for some } z \in f\}.$$

Notice that throughout we treat the function f as the set of its satisfying assignments, and therefore the above notation is natural. The set of *minimal assignments of* f with respect to b:

$$\min_b(f) = \{z \mid z \in f, \text{ such that } \forall y \in f, z \not\succ_b y\}.$$

Every Boolean function f can be represented in the following form:

$$f = \bigwedge_{b \in B} \mathcal{M}_b(f) = \bigwedge_{b \in B} \bigvee_{z \in \min_b(f)} \mathcal{M}_b(z) \tag{6}$$

for some set B. In this representation the set $B \subseteq \{0, 1\}^n$ is called a *monotone basis* for f. It is known that the size of the basis is at most the CNF size of f.

The set of *floor* assignments of an assignment x, with respect to the order relation b, denoted $\lfloor x \rfloor_b$, is the set of all elements $z <_b x$ such that there does not exist y for which $z <_b y <_b x$ (i.e., z is strictly smaller than x relative to b and is different from x in exactly one bit).

The set of *local minimal assignments of* f with respect to b is:

$$\min_b^*(f) = \{x \mid x \in f, \text{ and } \forall y \in \lfloor x \rfloor_b, y \notin f\}.$$

Clearly, we have that $\min_b(f) \subseteq \min_b^*(f)$. It is known that the size of $\min_b^*(f)$ and therefore the size of $\min_b(f)$, is bounded by the DNF size of f.

The representation in Eq. (6) yields the following necessary and sufficient condition for $x \in \{0, 1\}^n$ to be positive for f:

COROLLARY 6.1.2. *Let B be a basis for f, $x \in \{0, 1\}^n$. Then, $x \in f$ (i.e., $f(x) = 1$) if and only if for every basis element $b \in B$ there exists $z \in min_b(f)$ such that $x \geq_b z$.*

6.2. RESTRICTED QUERIES. A monotone basis can be used to characterize a class of Boolean functions: all those which have the same basis. Thus, we can use the notion of a basis to characterize classes of propositional queries:

Definition 6.2.1. Let B be a monotone basis for the Boolean function f. A class $\mathcal{Q}$ of Boolean functions is *relevant to* f if B is also a monotone basis for all the functions in $\mathcal{Q}$.

Definition 6.2.2. A class $\mathcal{Q}$ of Boolean functions is *common* if $\mathcal{Q}$ has a fixed, polynomial size monotone basis.

It is known [Bshouty 1995; Khardon and Roth 1995] that the class of Horn CNF functions has a basis of size $n + 1$, and that the class of $\log n$ CNF functions (CNF in which the clauses contain at most $O(\log n)$ literals) has a basis of size less than n^3. Other important examples of common languages are: k-quasi-Horn queries (a generalization of Horn theories in which there are at most k positive literals in each clause), reverse-k-quasi-Horn queries and others. Clearly, the union of common classes is also common.

6.3. REASONING WITH MODELS. Model based representations of knowledge are related to Levesque's [1986] notion of vivid representations. These representations have been studied in the context of Horn expressions in Kautz et al. [1995], and a general theory of reasoning with models has been developed in Khardon and Roth [1996], using the monotone theory. We briefly describe the results that are used here; for a more thorough discussion of reasoning with models, see Kharon and Roth [1996; 1997].

Reasoning with models is a very simple and highly parallelizable procedure. Let $\Gamma \subseteq f \subseteq \{0, 1\}^n$ be a set of models. To decide whether $f \models \alpha$ use the model-based approach to deduction: for all the models $z \in \Gamma$ check whether $\alpha(z) = 1$. If for some z, $\alpha(z) = 0$ say "no"; otherwise, say "yes". By definition, if $\Gamma = f$ this approach yields correct deduction, but representing f by explicitly holding *all* the possible models of f is not plausible. A model-based approach becomes feasible if Γ supports correct deduction and is small. In the following, we characterize a model-based knowledge base that provides for correct reasoning.

Definition 6.3.1. Let $\mathcal{F}$ be a class of functions, and let B be a basis for $\mathcal{F}$. For a knowledge base $f \in \mathcal{F}$ we define the set $\Gamma = \Gamma_f^B$ of *characteristic models* to be the set of all minimal assignments of f with respect to the basis B. Formally,

$$\Gamma_f^B = \cup_{b \in B} \{z \in \min_b(f)\}.$$

The following is the basic theorem of the theory of reasoning with models:

THEOREM 6.3.2. *Let $f, \alpha \in \mathcal{F}$ and let B be a basis for $\mathcal{F}$. Then $f \models \alpha$ if and only if for every $u \in \Gamma_f^B$, $\alpha(u) = 1$.*

Next we discuss the notion of a least upper bound of a Boolean function, its relation to the monotone theory and its usage in model-based reasoning.

Definition 6.3.3 (*Least Upper-bound*). Let $\mathcal{F}$, $\mathcal{G}$ be families of propositional languages. Given $f \in \mathcal{F}$ we say that $f_{lub} \in \mathcal{G}$ is a $\mathcal{G}$-least upper bound of f if and only if $f \subseteq f_{lub}$ and there is no $f' \in \mathcal{G}$ such that $f \subset f' \subset f_{lub}$.

THEOREM 6.3.4. *Let f be any propositional theory and $\mathcal{G}$ a class of all propositional theories with basis B. Then*

$$f_{lub} = \bigwedge_{b \in B} \mathcal{M}_b(f).$$

The above theorem shows that the logical function represented by the set Γ_f^B, where B is a basis for $\mathcal{G}$, is the LUB of f in $\mathcal{G}$. Nevertheless, this representation is sufficient to support exact deduction with respect to queries in $\mathcal{G}$:

THEOREM 6.3.5. *Let $f \in \mathcal{F}$, $\alpha \in \mathcal{G}$ and let B be a basis for $\mathcal{G}$. Then $f \models \alpha$ if and only if for every $u \in \Gamma_f^B$, $\alpha(u) = 1$.*

The complexity of model-based reasoning is directly related to the number of models in the representation. It is therefore important to compare this size with the size of other representations of the same function.

THEOREM 6.3.6. *Let f be any Boolean function, and B a basis. Then, the size of the model-based representation of f is*

$$|\Gamma_f^B| \leq \sum_{b \in B} |min_b(f)| \leq |B| \cdot |DNF(f)|.$$

We note that this bound is tight in the sense that for some functions the size of the DNF is indeed needed. It does however allow for an exponential gap in other cases. Namely, there are functions with an exponential size DNF and a linear size model-based representation. For a discussion of these issues see Khardon and Roth [1996]. Thus, when there exists a polynomial size basis B, the model-based representation is not worse than the DNF representation, and sometimes much better. As we show below, the model-based representation has an additional advantage over the DNF representation in terms of learnability.

6.4. LEARNING TO REASON WITHOUT REASONING. In this section, we give a Learning to Reason algorithm for a class of propositional languages for which there are no efficient reasoning algorithms, when represented as a traditional (formula-based) knowledge base. The result is simply obtained by observing that a L2C algorithm given by Bshouty [1995] can be used for this purpose since it yields as a byproduct the required knowledge representation.

THEOREM 6.4.1. *There is an Exact-Learn to Reason algorithm, that uses an Equivalence Query oracle and a Membership Query Oracle for the reasoning problem $(CNF \cap DNF, \mathcal{Q})$, where $\mathcal{Q}$ is any class of relevant and common queries.*

PROOF. The Learning to Reason algorithm learns a model-based representation for the target function f and then uses model-based reasoning to answer queries with respect to it. In Bshouty [1995], an algorithm is developed that uses an Equivalence Query oracle and a Membership Query Oracle to learn an exact representation of any function $f \in CNF \cap DNF$. As a byproduct of this algorithm, the set of all minimal models with respect to a basis B of f is produced. Using this set of models Γ, model-based reasoning, via Theorem 6.3.2, gives the result for relevant queries. Also, given basis B, the algorithm developed in Bshouty [1995] produces as a byproduct the set of minimal models of f with

Algorithm Ex-L2R-DNF

1. $\forall b \in B$, initialize $\Gamma_b \leftarrow \emptyset$.
2. $\alpha \leftarrow RQ(f, \mathcal{Q})$
3. Answer $f \models \alpha$ by performing model-based test on $\Gamma = \cup_{b \in B} \Gamma_b$.
4. If "wrong" then
5. let x be the counterexample received from $RQ(f, Q)$
6. $\forall b \in B$ such that $x \notin \mathcal{M}_b(\Gamma_b)$
7. $\Gamma_b \leftarrow \Gamma_b \cup$ Find-min-model(x,b)
8. GoTo 2

FIG. 1. The Algorithm *Ex-L2R-DNF*.

respect to B, provided that f has small DNF. Combining this with Theorem 6.3.5, we get the result for common queries. □

The above theorem is an example for a reasoning problem that is provably hard in the "traditional" sense and has an efficient solution in the new model. Given a CNF knowledge base, even with the added information that it has a short DNF, the reasoning problem is still hard. This is so since it is hard to find a satisfying assignment for a CNF expression even if one knows that it has exactly one satisfying assignment [Valiant and Vazirani 1986]. In this case, the additional reasoning power of the agent is gained through the interaction with the world by using $EQ(f)$.

6.5. LEARNING TO REASON WITHOUT LEARNING TO CLASSIFY. In this section, we present two results on Learning to Reason any Boolean function f with a polynomial size DNF. These results are significant since there is no known algorithm that Learns to Classify DNF.[10] We present two algorithms. The first algorithm (Theorem 6.5.1.1) makes use of a Reasoning Query Oracle $RQ(f, \mathcal{Q})$ and a Membership Query Oracle $MQ(f)$ to *exactly* Learn to Reason for the problem $(f, \mathcal{Q})$. It is worth noticing that this algorithm exhibits an Exact-Learning to Reason algorithm even though its knowledge base consists of an approximate description of the world. The second, more complicated algorithm, uses a "weaker" interface, $EX_D(f)$ instead of $RQ(f, \mathcal{Q})$, and yields PAC-Learn to Reason algorithm for f. The main idea in both algorithms is that it is sufficient to learn the least upper bound of a function in order to reason with common queries.

It is interesting to note that in some sense, a PAC-Learn to Reason algorithm for any Boolean function is implied from the one-time sampling approach developed in Section 3. However, the version presented here, while guaranteeing only PAC behavior, has the property that with a bounded number of reasoning

[10] Recently, an algorithm for Learning to Classify DNF expressions using a membership oracle has been developed [Jackson 1994]. However, this result is limited to the uniform distribution, and cannot be applied here. Furthermore, the representation used is a weighted sum of XOR functions which is not useful for reasoning.

Procedure Find-min-model(x,b)

1. If $\exists y \in \lfloor x \rfloor_b$ such that $f(y) = 1$ /* use MQ(f) */
2. $x \leftarrow y$; GoTo 1
3. Else
4. Return(x)

FIG. 2. The Procedure *Find-min-model(x, b)*.

mistakes it converges to yield exact reasoning performance. We believe that this property is useful for a Learning to Reason algorithm.

We now present the algorithms. Both are based on a modified version of Bshouty's algorithm to learn Boolean functions via the monotone theory [Bshouty 1995]. In the following, fix $\mathcal{Q}$ to be any class of common queries, and let B be a basis for $\mathcal{Q}$; the results are proved relative to $\mathcal{Q}$.

6.5.1. *Exact Learning to Reason*. Figure 1 describes the algorithm *Ex-L2R-DNF*. The algorithm interacts with RQ, and collects a set of models $\Gamma = \cup_{b \in B} \Gamma_b$, the set of *locally* minimal assignments of f with respect to B, as its model-based representation. In order to answer the reasoning questions presented to it, it simply uses model-based reasoning with Γ. Each mistake the algorithm makes provides a counterexample, and using membership queries and the counterexample the algorithm finds an additional minimal model. Since the set Γ contains eventually the set of minimal models of f_{lub} with respect to the basis B it guarantees, by Theorem 6.3.5, exact reasoning with respect to queries in $\mathcal{Q}$.

THEOREM 6.5.1.1. *Algorithm Ex-L2R-DNF is a MB-Learn to Reason algorithm for the problem* (*DNF*, $\mathcal{Q}$), *where* $\mathcal{Q}$ *is the class of all common queries*.

PROOF. Denote $h = \wedge_{b \in B}(\vee_{z \in \Gamma_b} \mathcal{M}_b(z))$. Clearly, at every step in the algorithm, $\Gamma \subseteq f$, and therefore the algorithm *Ex-L2R-DNF* never makes a mistake when it says "no" (and is therefore well defined). Whenever the algorithm errs on an $RQ(f, \mathcal{Q})$ query α, it receives from the oracle a positive counterexample, $x \in f \setminus \alpha$. We first argue that $x \in f \setminus h$. In order to show that, we prove that $h \subseteq \alpha$. Indeed, let y be such that $h(y) = 1$. Then, $\forall b \in B$, $\exists z_b \in \Gamma_b$ such that $z_b \leq_b y$. Since the response on the reasoning query was "yes", we have that in particular $\alpha(z_b) = 1$, for all those points z_b. Now, since $\alpha \in \mathcal{Q}$, using Corollary 6.1.2, we get that $\alpha(y) = 1$.

Now, since $x \notin h$, it is negative for at least one of the b's in the conjunction defining h. Therefore, there exists a model $z \in \min_b(f) \setminus \Gamma_b$ for each such b, and the algorithm can use the procedure Find-min-model(x, b) to find a new model of f, an element of $min_b^*(f)$. Find-min-model(x, b) is a standard procedure [Angluin 1988; Bshouty 1995] that uses a sequence of calls to $MQ(f)$ to find an element of $min_b^*(f)$. Since for all $b \in B$, $|min_b^*(f)| \leq |DNF(f)|$, the size of this representation is polynomial. Moreover, $\Gamma_f^B \subseteq \cup_{b \in B} min_b^*(f) \subseteq f$. (See Figure 2.)

The algorithm might make reasoning mistakes as long as there is an element of Γ_f^B missing from its model-based representation, but with every such mistake it makes progress toward collecting the elements in the set Γ_f^B. Therefore, after at

Algorithm PAC-L2R-DNF

Learning Phase:

1. $\forall b \in B$, initialize $\Gamma_b \leftarrow \emptyset$.
2. Let $h = \wedge_{b \in B} (\vee_{z \in \Gamma_b} \mathcal{M}_b(z))$.
3. Call $EX_D(f)$ $m = \frac{1}{\epsilon} \log \frac{1}{\delta}$ times. Let S be the set of all m samples.
4. If $\neg \exists x \in S$ such that $f(x) = 1$ and $h(x) = 0$ GoTo 9 /* no positive counterexample */
5. $\forall x \in S$ such that $f(x) = 1$ and $h(x) = 0$ do: /* positive counterexample */
6. $\forall b \in B$ such that $x \notin \mathcal{M}_b(\Gamma_b)$ do:
7. $\Gamma_b \leftarrow \Gamma_b \cup$ Find-min-model(x,b)
8. GoTo 2
9. Return $\Gamma = \cup_{b \in B} \Gamma_b$

Reasoning Phase:

Answer queries by performing model-based reasoning on $\Gamma = \cup_{b \in B} \Gamma_b$

FIG. 3. The Algorithm *PAC-L2R-DNF*.

most $|\cup_{b \in B} min_b^*(f)| \leq |B| \cdot |DNF(f)|$ mistakes algorithm *Ex-L2R-DNF* makes no more mistakes on $RQ(f, \mathcal{Q})$ queries and by Theorem 6.3.4, $h = f_{lub}$.

Therefore, Theorem 6.3.5 implies that *Ex-L2R-DNF* guarantees exact reasoning on queries from $\mathcal{Q}$. □

6.5.2. *PAC Learning to Reason*. Similar to the previous algorithm, the PAC learning algorithm collects a set of locally minimal models of f with respect to a fixed basis B, a basis for $\mathcal{Q}$. By Theorem 6.3.4 this yields a representation of the least upper bound of f in $\mathcal{Q}$, and by Theorem 6.3.5, this is enough to solve the ($\mathcal{F}$, $\mathcal{Q}$) reasoning problem. Unlike the exact L2R algorithm, here we simulate the $RQ(f, \mathcal{Q})$ oracle by calling the Example Oracle $EX_D(f)$ sufficiently many times. This simulation builds on the standard simulation of Equivalence Query Oracle by calls to Example Oracle [Angluin 1988] and uses some nice properties of the reasoning with models framework. On the other hand, this algorithm is not a mistake-bound algorithm, and the time it takes to achieve its final state does not depend on the behavior of the oracles and can be bounded.

Figure 3 describes the algorithm *PAC-L2R-DNF*. For all $b \in B$ we initialize $\Gamma_b = \emptyset$ and maintain a hypothesis $h = \wedge_{b \in B}(\vee_{z \in \Gamma_b} \mathcal{M}_b(z))$. To get counterexamples, we simulate $EQ(f)$ by $m = (1/\epsilon)\log(1/\delta)$ calls to $EX_D(f)$. On a positive counterexample the algorithm looks for a *locally* minimal assignment of f to be added to Γ_b. Such an assignment always exists since this example is negative for at least one of the b's in the intersection of h. Finding the locally minimal assignment is done by calling the procedure *Find-min-model*(x, b) that uses $MQ(f)$ greedily to find a locally minimal element. That is, the algorithm increases the size of at least one of the Γ_b's. Whenever the algorithm receives a negative counterexample, it simply ignores it. The algorithm stops if m consecutive calls to $EX_D(f)$ do not return any positive counterexample.

We note that simulating $EQ(f)$ by $EX_D(f)$ is essential to our algorithm. Otherwise, an adversarial equivalence oracle can adapt its strategy to the hypothesis that the algorithm holds. In particular, it could prevent the algorithm from making progress by presenting the same negative counterexample forever. The following lemma shows that with an example oracle this is not the case.

LEMMA 6.5.2.1. *Let f be a Boolean function with a polynomial size DNF representation and let f_{lub} be its least upper bound in $\mathcal{Q}$. Then algorithm PAC-L2R-DNF runs in time polynomial in n and, with probability $> 1 - \delta$ reaches the reasoning phase with a hypothesis h, which has the following properties*: (1) $h \subseteq f_{lub}$ (2) $Pr_D[f \setminus h] < \epsilon$.

PROOF. Property (1) easily follows from Theorem 6.3.4 and from the fact that at any point the algorithm holds in Γ_b only (locally) minimal assignment of f.

For (2), we first note that the size of Γ_b for any basis assignment is bounded by the DNF size of f, and is therefore polynomial. This implies that there is a polynomial bound on the number of positive counterexamples that the algorithm might receive. Now consider the algorithm's hypothesis when it stopped. If the algorithm already used up its mistake bound then by Theorem 6.3.4, $h = f_{lub}$ and we are done. Otherwise, it stopped because it did not receive a positive counterexample from the simulation of the equivalence query. By the selection of m, this implies that with probability $1 - \delta$, $\Pr_D[f \setminus h] < \epsilon$. □

The following theorem shows that the properties of Lemma 6.5.2.1 guarantee the correctness of the algorithm.

THEOREM 6.5.2.2. *Algorithm PAC-L2R-DNF is a PAC-Learn to Reason algorithm for the problem* (*DNF*, $\mathcal{Q}$), *where $\mathcal{Q}$ is the class of all common queries*.

PROOF. Let $\Gamma = \cup_{b \in B} \Gamma_b$ be the output of the learning phase of the algorithm. Γ is a subset of the set of all locally minimal assignments of f with respect to B.

To prove the correctness of the algorithm consider first the case in which $f \models \alpha$. Since $\Gamma \subseteq f$, $\forall x \in \Gamma$, $\alpha(x) = 1$ and the algorithm answers "yes".

Assume now that $f \not\models \alpha$, and suppose, by way of contradiction, that algorithm answers "yes" on α. Since α is (f, ϵ)-fair, we know that $\Pr_D[f \setminus \alpha] > \epsilon$. Since by Lemma 6.5.2.1, $\Pr_D[f \setminus h] < \epsilon$, this implies that $h \setminus \alpha \neq \emptyset$.

Let $y \in h \setminus \alpha$. Since, by the assumption, the algorithm answers "yes", we know that for all $u \in \Gamma$, $\alpha(u) = 1$. Since B is a basis for α and for all $u \in \Gamma$, $\alpha(u) = 1$, Corollary 6.1.2 implies that

$$\forall u \in \Gamma,\ \forall b \in B,\ \exists v_{u,b} \in \min_b(\alpha), \qquad \text{such that} \quad u \geq_b v_{u,b}. \tag{7}$$

Since $y \in h = \wedge_{b \in B}(\vee_{z \in \Gamma_b} \mathcal{M}_b(z))$, we have that

$$\forall b \in B,\ \exists z \in \Gamma_b, \qquad \text{such that} \quad y \geq_b z. \tag{8}$$

By the assumption, all the elements z identified in Eq. (8) satisfy α and therefore, as in Eq. (7) we have that

$$\forall z \in \Gamma,\ \forall b \in B,\ \exists v_{z,b} \in \min_b(\alpha) \qquad \text{such that} \quad z \geq_b v_{z,b}. \tag{9}$$

Substituting Eq. (9) into Eq. (8) gives

$$\forall b \in B,\ \exists v_{(z),b} \in \min_b(\alpha) \qquad \text{such that} \quad y \geq_b v_{(z),b}$$

which implies, by Corollary 6.1.2, that $\alpha(y) = 1$, a contradiction. □

We have shown that there exist Learning to Reason algorithms for the all Boolean functions with a polynomial size DNF, provided that the queries are common. This should be contrasted with the inability to *Learn to Classify* DNF. The problem of learning DNF expression with Membership queries and either Equivalence or Superset[11] queries, is still an open problem. (The result by Jackson [1994] mentioned above holds only for a restricted set of distributions.) Our results show that one can learn f_{lub} and use it for reasoning with respect to common queries, but f_{lub} is not sufficient as a substitute for f when classifying examples, since the size of $\Pr_D[h \setminus f]$ cannot be bounded.

6.6. Structure Identification. We briefly describe another application of model-based reasoning.

Dechter and Pearl [1992] investigate the problem of identifying tractable classes of CNF formulas. In particular they consider the following problem: Given a set ρ of models, can one:

(1) Find a Horn theory f such that $f = \rho$, if one exists.
(2) Find a Horn theory f such that $\rho \subseteq f$ and there is no Horn function g such that $\rho \subseteq g \subset f$.

In Dechter and Pearl [1992], it is shown ("Horn theories are identifiable", Corollary 4.11) that when ρ is a set of models of a Horn theory, a theory f can be found in time polynomial in $|\rho|$. The second problem ("strong-identifiability of Horn-theories"), which is just the problem of finding ρ^H_{lub} (i.e., the least upper bound of ρ with respect to Horn) is left open.

The interest in the question of the identifiability and strong identifiability of Horn theories is partly motivated by the fact that Horn theories are tractable, and in particular, given a CNF formula in a Horn form, reasoning with it can be performed efficiently. Therefore, using the strong-identifiability one can perform efficient reasoning with a theory that is the best Horn approximation of ρ. We show that some insight on this question can be gained from the theory developed here.

Claim 6.6.1. *Let ρ be a set of models and g the least Horn upper bound of ρ. Then*,

(*i*) *A closed form formula for the required approximation can be written as a conjunction of DNF formulas:*

$$g = \bigwedge_{B_H} \bigvee_{z \in \rho} \mathcal{M}_b(z). \tag{10}$$

(*ii*) *For every query α, the reasoning problem $g \models \alpha$ can be answered correctly using the set ρ.*

Proof. (i) is immediate from Theorem 6.2.2, since $g = \rho^H_{lub}$. For (ii) notice that if the query α is a Horn query, then model-based reasoning with ρ is correct, by Theorem 6.3.5. For correct reasoning with general queries, it is possible to use the reasoning algorithm from Kautz et al. [1993]. □

[11] While the oracle *RQ* has not been studied elsewhere, it is similar to superset queries that have been considered in the literature [Angluin 1988].

7. *Learning to Reason with Horn Queries*

Model-based representations are not unique in supporting Learning to Reason. In this section, we exhibit another instance of Learning to Reason in which (formula-based) reasoning is computationally hard. (The corresponding learning problem has not been considered before.) This time we use formulas in Horn form as the knowledge representation although the world function f may not be Horn.

In previous sections (e.g., Theorem 6.4.1, Theorem 6.5.1.1), we have shown that for various worlds we can learn to reason with respect to all common queries. We give here an orthogonal result that shows that we can Learn to Reason from every Boolean function (with polynomial size Horn least upper bound), with respect to Horn queries. The result is based of an algorithm developed by Frazier and Pitt [1993] for the purpose of Learning to Classify Horn functions.

Another difference from the previous section is that the oracles used here, following Frazier and Pitt [1993], are entailment oracles. The agent "learns" the world from examples that are statements that the world entails and does not entail, rather than, seeing positive and negative instances or the world, as in $EX_D(f)$ and $MQ(f)$.

Recall the definitions for the entailment oracles $EnMQ(f)$, and $EnEQ(f)$ given above (Definitions 2.2.4 and 2.2.5). The algorithm LRN presented in Frazier and Pitt [1993] interacts with $EnMQ(f)$, and $EnEQ(f)$, and is shown there to exactly learn Horn expressions. It can be easily shown (by a similar proof) that the algorithm LRN can be used to find the least upper bound of any Boolean function f with respect to Horn and thus, by Theorems 6.3.4 and 6.3.5, this can be used further to reason with Horn queries.

In particular, this provides another example in which there exists an efficient L2R algorithm but reasoning is hard, similar to Section 6.4. In this case, the problem $f \models \alpha$, even when α is a Horn query, is in general hard if f is given as a CNF formula. We summarize in the next theorem:

THEOREM 7.1. *There is an Exact-Learn to Reason algorithm, that uses an Entailment Equivalence Query oracle and an Entailment Membership Query Oracle, for* $(\mathcal{F}, H)$, *where* $\mathcal{F}$ *is any class of Boolean functions with polynomial size Horn least upper bound.*

8. *Conclusions*

We have introduced a new framework for the study of reasoning in intelligent systems. The approach suggested, *Learning to Reason*, is intended to overcome some of the basic problems in the traditional approach to reasoning. This framework differs from existing ones in that it sees learning as an integral part of the process, it avoids enforcing rigid syntactic restrictions on the intermediate knowledge representation, and it makes explicit the dependence of the reasoning performance on the input from the environment.

We have shown that under some restrictions on the queries asked (which depend on the "world"), Learning to Reason is possible for all propositional languages. This should be contrasted with the hardness of traditional reasoning even with very restricted languages. We have discussed the relation of the new

framework to previous computational approaches to learning and to reasoning, and have shown how results in learning and reasoning can be used in the new framework.

Our main results exhibit cases in which the new framework allows for successful Learning to Reason algorithms, but stated separately, either the reasoning problem or the learning problem are not (or not known to be) tractable.

Overall, the approach developed here suggests an "operational" approach to the study of reasoning, that is nevertheless rigorous and amenable to analysis. We believe that this framework is a step toward constructing an adequate computational theory of reasoning. In particular, it draws attention to many issues that should be addressed in developing such a theory. We mention just two such issues that do not get much attention in the traditional reasoning framework. The first is quantifying the reasoning performance relative to the "world" and to a restricted set of queries. The second is the consideration of a plausible interface between the reasoner and the world (instead of, or in addition to, the traditional formula-based representation of the world).

Although in this work we have presented the Learning to Reason framework in the context of deductive reasoning, it should be viewed in a more general setting, where a variety of reasoning tasks can be considered. For example, the results on learning model-based representations in this paper together with the results in Khardon and Roth [1996] yield an algorithm for Learning to Reason abductively. Together with results in Khardon and Roth [1997] it yields an algorithm for Learning to Reason with defaults. Other reasoning tasks, which are currently under investigation, include Learning to Reason in a probabilistic framework and Learning to Act [Greiner et al. 1996; Roth 1995; Khardon 1996]. Other extensions of the approach, where we consider cases in which the interface provides the agent only partial information about the world or the queries are studied in Khardon and Roth [1995] and Roth [1995].

ACKNOWLEDGMENTS. We are grateful to Les Valiant for many enjoyable discussions that helped us develop the ideas presented here, and to Joe Halpern for useful comments on an earlier draft.

REFERENCES

AMSTERDAM, J. 1988. Extending the Valiant learning model. In *Proceeding of the 5th International Workshop on Machine Learning*. pp. 364–375.

ANGLUIN, D. 1988. Queries and concept learning. *Mach. Learn. 2*, 4 (Apr.), 319–342.

ANGLUIN, D. 1992. Computational learning theory: Survey and selected bibliography. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing* (Victoria, B.C., Canada, May 4–6). ACM, New York, pp. 351–369.

ANGLUIN, D., AND LAIRD, P. 1988. Learning from noisy examples. *Mach. Learn. 2*, 4, 343–370.

ANGLUIN, D., AND SLONIM, D. K. 1994. Randomly fallible teachers: Learning monotone DNF with an incomplete membership oracle. *Mach. Learn. 19*, 7–126.

BACCHUS, F., GROVE, A., HALPERN, J. Y., AND KOLLER, D. 1993. Statistical foundations for default reasoning. In *Proceedings of the International Joint Conference of Artificial Intelligence*. 563–569.

BACCHUS, F., GROVE, A., HALPERN, J. Y., AND KOLLER, D. 1996. From statistical knowledge bases to degrees of beliefs. *Artif. Int. 87*, 1–2, 75–143.

BROOKS, R. A. 1991. Intelligence without representation. *Artif. Int. 47*, 139–159.

BSHOUTY, N. H. 1995. Exact learning via the monotone theory. *Inf. Comput. 123*, 1, 146–153.

CADOLI, M. 1995. *Tractable Reasoning in Artificial Intelligence*. In Lecture Notes in Artificial Intelligence, vol. 941. Springer-Verlag, New York.

DECHTER, R., AND PEARL, J. 1992. Structure identification in relational data. *Artif. Int. 58*, 237–270.

DOYLE, J., AND PATIL, R. 1991. Two theses of knowledge representation: Language restrictions, taxonomic classification, and the utility of representation services. *Artif. Int. 48*, 261–297.

FAGIN, R., HALPERN, J. Y., MOSES, Y., AND VARDI, M. Y. 1995. *Reasoning about Knowledge*. MIT Press, Cambridge, Mass.

FRAZIER, M., AND PITT. L. 1993. Learning from entailment: An application to propositional Horn sentences. In *Proceedings of the International Conference on Machine Learning*. Morgan-Kaufmann, San Mateo, Calif.

GREINER, R., GROVE, A., AND ROTH, D. 1996. Learning active classifiers. In *Proceedings of the International Conference on Machine Learning.*

HAUSSLER, D. 1987. Bias, version spaces and Valiant's learning framework. In *Proceedings of the 4th International Workshop on Machine Learning*. Univ. California, Irvine, Irvine, Calif.

HENRION, M. 1988. Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In *Uncertainty in Artificial Intelligence.*

JACKSON, J. 1994. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. In *Proceedings of the IEEE Symposium on Foundation of Computer Science*. IEEE, New York, pp. 42–53.

KAUTZ, H., KEARNS, M., AND SELMAN, B. 1993. Reasoning with characteristic models. In *Proceedings of the National Conference on Artificial Intelligence*. pp. 34–39.

KAUTZ, H., KEARNS, M., AND SELMAN, B. 1995. Horn approximations of empirical data. *Artif. Int. 74*, 129–145.

KEARNS, M. 1992. Oblivious pac learning of concepts hierarchies. In *Proceedings of the National Conference on Artificial Intelligence*. pp. 215–222.

KEARNS, M., AND LI, M. 1988. Learning in the presence of malicious errors. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing* (Chicago, Ill., May 2–4). ACM, New York, pp. 267–280.

KHARDON, R. 1996. Learning to take actions. In *Proceedings of the National Conference on Artificial Intelligence*. pp. 787–792.

KHARDON, R., AND ROTH, D. 1995. Learning to reason with a restricted view. In *Workshop on Computational Learning Theory*. pp. 301–310.

KHARDON, R., AND ROTH, D. 1996. Reasoning with models. *Artif. Int. 87*, 1–2, 187–213.

KHARDON, R., AND ROTH, D. 1997. Default-reasoning with models. *Artif. Int. J*., to appear.

KIRSH, D. 1991. Foundations of AI: The big issues. *Artif. Int. 47*, 3–30.

LEVESQUE, H. J. 1984. Foundations of a functional approach to knowledge representation. *Artif. Int. 23*, 155–212.

LEVESQUE, H. 1986. Making believers out of computers. *Artif. Int. 30*, 81–108.

LEVESQUE, H. 1992. Is reasoning too hard? In *Proceeding of the 3rd NEC Research Symposium.*

LEVESQUE, H., AND BRACHMAN, R. 1985. A fundamental tradeoff in knowledge representation and reasoning. In *Readings in Knowledge Representation*, R. Brachman and H. Levesque, eds. Morgan-Kaufman, San Mateo, Calif.

LITTLESTONE, N. 1988. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Mach. Learn. 2*, 285–318.

MCCARTHY, J. 1985. Programs with common sense. In *Readings in Knowledge Representation, 1985*, R. Brachman and H. Levesque, eds. Morgan-Kaufmann, San Mateo, Calif.

MOSES, Y., AND TENNENHOLTZ, M. 1996. Off-line reasoning for on-line efficiency: Knowledge bases. *Artif. Int. 83*, 2, 229–239.

NILSSON, N. J. 1986. Probabilistic logic. *Artif. Int. 28*, 71–87.

NILSSON, N. J. 1991. Logic and artificial intelligence. *Artif. Int. 47*, 31–56.

PAPADIMITRIOU, C. H. 1991. On selecting a satisfying truth assignment. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science*. IEEE, New York.

ROTH, D. 1995. Learning to reason: The non-monotonic case. In *Proceedings of the International Joint Conference of Artificial Intelligence*. pp. 1178–1184.

ROTH, D. 1996. On the hardness of approximate reasoning. *Artif. Int. 82*, 1–2 (Apr.), 273–302.

SELMAN, B. 1990. Tractable default reasoning. Ph.D. dissertation. Department of Computer Science, University of Toronto, Toronto, Ont., Canada.

SELMAN, B., AND KAUTZ, H. 1996. Knowledge compilation and theory approximation. *J. ACM 43*, 2 (Mar.), 193–224.

SHASTRI, L. 1993. A computational model of tractable reasoning—Taking inspiration from cognition. In *Proceeding of the International Joint Conference of Artificial Intelligence* (Aug.). pp. 202–207.
VALIANT, L. G. 1984. A theory of the learnable. *Commun. ACM 27*, 11 (Nov.), 1134–1142.
VALIANT, L. G. 1985. Learning disjunctions of conjunctions. In *Proceedings of the International Joint Conference of Artificial Intelligence*. Morgan-Kaufmann, San Mateo, Calif., pp. 560–566.
VALIANT, L. G. 1994. *Circuits of the Mind* (Nov.). Oxford University Press, Oxford, England.
VALIANT, L. G. 1994. Rationality. In *Workshop on Computational Learning Theory* (July). pp. 3–14.
VALIANT, L. G., AND VAZIRANI, V. V. 1986. NP is as easy as detecting unique solutions. *Theoret. Comput. Sci. 47*, 85–93.

RECEIVED AUGUST 1994; REVISED FEBRUARY 1996; ACCEPTED MARCH 1997