Learning Regular Sets from Queries and Counterexamples*

DANA ANGLUIN

Department of Computer Science, Yale University,
P.O. Box 2158, Yale Station, New Haven, Connecticut 06520

The problem of identifying an unknown regular set from examples of its members and nonmembers is addressed. It is assumed that the regular set is presented by a *minimally adequate Teacher*, which can answer membership queries about the set and can also test a conjecture and indicate whether it is equal to the unknown set and provide a counterexample if not. (A counterexample is a string in the symmetric difference of the correct set and the conjectured set.) A learning algorithm L^* is described that correctly learns any regular set from any minimally adequate Teacher in time polynomial in the number of states of the minimum dfa for the set and the maximum length of any counterexample provided by the Teacher. It is shown that in a stochastic setting the ability of the Teacher to test conjectures may be replaced by a random sampling oracle, $EX(\ )$. A polynomial-time learning algorithm is shown for a particular problem of context-free language identification. © 1987 Academic Press, Inc.

1. INTRODUCTION

Imagine a human expert in some domain, for example, cancer diagnosis, attempting to communicate the method he or she uses in that domain to an expert system. Specific positive and negative examples will form an important component of the communication, in addition to advice about general rules, explanations of significant and irrelevant features, justifications of lines of reasoning, clarifications of exceptions, and so on. Moreover, the examples given are likely to be chosen so that they are "central" or "crucial" rather than random or arbitrary, in an attempt to speed convergence of the system to a correct hypothesis.

Studies of learning general rules from examples have generally assumed a source of examples that is arbitrary or random [2]. The scenario above suggests that it is reasonable to investigate learning methods that assume that the source of examples is "helpful." To emphasize this aspect, the source of examples will be called the *Teacher* and the learning algorithm the *Learner*.

* Research supported by the National Science Foundation Grant IRI-8404226.

0890-5401/87 $3.00

How can we tell whether the Learner is assuming "too much help" from the Teacher? The answer to this question is ultimately empirical; it depends on whether the assumptions about the Teacher are satisfied by the actual source of examples in an intended application. It therefore seems impossible to give a single theoretical definition of "too much help." We demonstrate in this paper a Learner that efficiently learns an initially unknown regular set from any of a fairly wide class of Teachers, called "minimally adequate Teachers."

1.1. *Minimally Adequate Teachers*

A *minimally adequate Teacher* is assumed to answer correctly two types of questions from the Learner about the unknown regular set. The first type is a *membership query*, consisting of a string t; the answer is *yes* or *no* depending on whether t is a member of the unknown set or not. The second type of question is a *conjecture*, consisting of a description of a (equivalence regular set S; the answer is *yes* if S is equal to the unknown language and is a string t in the symmetric difference of S and the unknown language otherwise. In the second case, the string t is called a *counterexample* because it serves to show that the conjectured set S is incorrect.

Note that there is only one correct answer to a membership query, so all minimally adequate Teachers must answer these identically for a particular regular set. However, in the case of an incorrect conjecture, any of the possible counterexamples may be given by a minimally adequate Teacher.

1.2. *Discussion*

How "reasonable" is the assumption of minimal adequacy of the Teacher? Answering membership queries, that is, classifying instances proposed by the Learner, seems like an unobjectionable ability to require of a Teacher, although practically it will be important to be able to deal with errors in the Teacher's answers. Answering the second type of question, which involves a test of whether the conjecture is equivalent to the correct hypothesis, and finding a counterexample if not, is more problematic. This seems to require the Teacher to have a very precise and explicit representation of the correct hypothesis, a common language with the Learner for the description of hypotheses (rather than just examples), and appears to limit the domain to one in which equivalence of hypotheses is at least a decidable question. However, we show later in the paper that using the criterion of approximate identification proposed by Valiant [8], we may substitute a random sampling oracle for the ability to answer questions of the second type, removing some of the limitations of the assumption of minimal adequacy of the Teacher.

How feasible are the computations required of a minimally adequate Teacher in this setting? The answer depends on the form of description of the unknown regular set and the conjectures presented by the Learner. The particular Learner we shall present describes regular sets by means of deterministic finite-state acceptors (dfa's) with some straightforward encoding. If we assume in addition that the unknown regular set to be taught is presented to the Teacher as a dfa of n states, then there exists a minimally adequate Teacher T^* that answers each question in time polynomial in n and the length of the question. To answer membership queries, T^* merely traces the given string through the dfa representing the unknown language. To answer conjectures, T^* performs a standard polynomial-time algorithm for testing the equivalence of two dfa's, which yields a counterexample in case they are not equivalent.

Note that we have avoided giving formal definitions of Teachers, Learners, questions, dialogs and so on, preferring not to obscure the intuitive simplicity of the present paper. An alternative characterization of a minimally adequate Teacher is as a pair of oracles, MEMBER(x) and EQUIV(M), with the obvious interpretations. This characterization tends to direct attention away from the possibility of a "pedagogical strategy" on the part of the Teacher, and I will avoid it except in the exposition of the stochastic setting.

1.3. *Related Results*

The identification of regular sets from examples has been studied quite extensively [2]. Gold [4] has shown that the problem of finding a dfa of a minimum number of states compatible with a given finite set of positive and negative examples is NP-hard. This result is generally interpreted as indicating that even a very simple case of inductive inference, inferring dfas from positive and negative examples, is computationally intractable.

Below we show that with the additional information available in membership queries, there is an efficient algorithm to find the (unique) minimum dfa compatible with the given examples and the answers to the queries. Thus, the information in membership queries can be used to avoid a lengthy computation.

The results to be described are related to and based upon a method of identifying a regular set from a "representative sample" and membership queries in polynomial time, described in [1]. It is also related to the method used by Gold [4,5]. However, the present paper is self-contained.

The stochastic setting we use was proposed by Valiant [8] and generalized by Blumer *et al.* [3].

2. Main Result

We describe the learning algorithm L^* and show that it efficiently learns an initially unknown regular set from any minimally adequate Teacher. Let the unknown regular set be denoted by U and assume that it is over a fixed known finite alphabet A.

2.1. *Observation Tables*

At any given time, the algorithm L^* has information about a finite collection of strings over A, classifying them as members or nonmembers of the unknown regular set U. This information is organized into an *observation table*, consisting of three things: a nonempty finite prefix-closed set S of strings, a nonempty finite suffix-closed set E of strings, and a finite function T mapping $((S \cup S \cdot A) \cdot E)$ to $\{0, 1\}$. The observation table will be denoted (S, E, T). (A set is *prefix-closed* if and only if every prefix of every member of the set is also a member of the set. Suffix-closed is defined analogously.)

The interpretation of T is that $T(u)$ is 1 if and only if u is a member of the unknown regular set, U. The observation table initially has $S = E = \{\lambda\}$, and is augmented as the algorithm runs.

An observation table can be visualized as a two-dimensional array with rows labelled by elements of $(S \cup S \cdot A)$ and columns labelled by elements of E, with the entry for row s and column e equal to $T(s \cdot e)$, If s is an element of $(S \cup S \cdot A)$, then row(s) denotes the finite function f from E to $\{0, 1\}$ defined by $f(e) = T(s \cdot e)$.

The algorithm L^* eventually uses the observation table to build a deterministic finite-state acceptor. Rows labelled by elements of S are the candidates for states of the acceptor being constructed, and columns labelled by elements of E correspond to distinguishing experiments for these states. Rows labelled by elements of $S \cdot A$ are used to construct the transition function.

Closed, consistent observation tables. An observation table is called *closed* provided that for each t in $S \cdot A$ there exists an s in S such that row(t) = row(s). An observation table is called *consistent* provided that whenever s_1 and s_2 are elements of S such that row(s_1) = row(s_2), for all a in A, row$(s_1 \cdot a)$ = row$(s_2 \cdot a)$.

If (S, E, T) is a closed, consistent observation table, we define a corresponding acceptor $M(S, E, T)$ over the alphabet A, with state set Q, initial state q_0, accepting states F, and transition function δ as follows:

$$\begin{aligned} Q &= \{\text{row}(s) : s \in S\}, \\ q_0 &= \text{row}(\lambda), \\ F &= \{\text{row}(s) : s \in S \text{ and } T(s) = 1\}, \\ \delta(\text{row}(s), a) &= \text{row}(s \cdot a). \end{aligned}$$

To see that this is a well defined acceptor, note that since S is a nonempty prefix-closed set, it must contain λ, so q_0 is defined. Also, since E is nonempty and suffix-closed, it also must contain λ. Thus, if s_1 and s_2 are elements of S such that $\text{row}(s_1) = \text{row}(s_2)$, then $T(s_1) = T(s_1 \cdot \lambda)$ and $T(s_2) = T(s_2 \cdot \lambda)$ are defined and equal to each other, so F is well defined. To see that δ is well defined, suppose s_1 and s_2 are elements of S such that $\text{row}(s_1) = \text{row}(s_2)$. Then since the observation table (S, E, T) is consistent, for each a in A, $\text{row}(s_1 \cdot a) = \text{row}(s_2 \cdot a)$, and since it is closed, this common value is equal to $\text{row}(s)$ for some s in S.

The important fact about this acceptor is the following.

THEOREM 1. *If (S, E, T) is a closed, consistent observation table, then the acceptor $M(S, E, T)$ is consistent with the finite function T. Any other acceptor consistent with T but inequivalent to $M(S, E, T)$ must have more states.*

This theorem is proved by a sequence of straightforward lemmas.

LEMMA 2. *Assume that (S, E, T) is a closed, consistent observation table. For the acceptor $M(S, E, T)$ and for every s in $(S \cup S \cdot A)$, $\delta(q_0, s) = \text{row}(s)$.*

This lemma is proved by induction on the length of s. It is clearly true when the length of s is 0, i.e., $s = \lambda$, since $q_0 = \text{row}(\lambda)$.

Assume that for every s in $(S \cup S \cdot A)$ of length at most k, $\delta(q_0, s) = \text{row}(s)$. Let t be an element of $(S \cup S \cdot A)$ of length $k + 1$, that is, $t = s \cdot a$ for some string s of length k and some a in A. In fact, s must be in S, for either t is in $S \cdot A$, and therefore s in S, or t is in S, and since S is prefix-closed, s is in S. Then

$$\begin{aligned} \delta(q_0, t) &= \delta(\delta(q_0, s), a) \\ &= \delta(\text{row}(s), a), && \text{by the induction hypothesis,} \\ &= \text{row}(s \cdot a), && \text{by the definition of } \delta, \\ &= \text{row}(t), && \text{since } t = s \cdot a. \end{aligned}$$

This completes the induction and the proof of Lemma 2.

LEMMA 3. *Assume that (S, E, T) is a closed, consistent observation table. Then the acceptor $M(S, E, T)$ is consistent with the finite function T. That is, for every every s in $(S \cup S \cdot A)$ and e in E, $\delta(q_0, s \cdot e)$ is in F if and only if $T(s \cdot e) = 1$.*

This lemma is proved by induction on the length of e. When e is λ and s is any element of $(S \cup S \cdot A)$, by the preceding lemma, $\delta(q_0, s \cdot e)$ is just $\text{row}(s)$. If s is in S, then by definition of F, $\text{row}(s)$ is in F if and only if

$T(s)=1$. If s is in $S \cdot A$, then since the observation table is closed, $\text{row}(s)=\text{row}(s_1)$ for some s_1 in S, and $\text{row}(s_1)$ is in F if and only if $T(s_1)=1$, which is true if and only if $T(s)=1$.

Suppose the result holds for all e from E of length at most k, and let e be an element of E of length $k+1$. Then, since E is suffix-closed, $e=a \cdot e_1$ for some a in A and some e_1 in E of length k. Let s be any element of $(S \cup S \cdot A)$. Because the observation table is closed, there exists a string s_1 in S such that $\text{row}(s)=\text{row}(s_1)$. Then

$$\begin{aligned}
\delta(q_0, s \cdot e) &= \delta(\delta(q_0, s), a \cdot e_1) \\
&= \delta(\text{row}(s), a \cdot e_1), && \text{by the preceding lemma,} \\
&= \delta(\text{row}(s_1), a \cdot e_1), && \text{since } \text{row}(s)=\text{row}(s_1), \\
&= \delta(\delta(\text{row}(s_1), a), e_1) \\
&= \delta(\text{row}(s_1 \cdot a), e_1), && \text{by the definition of } \delta, \\
&= \delta(\delta(q_0, s_1 \cdot a), e_1), && \text{by the preceding lemma,} \\
&= \delta(q_0, s_1 \cdot a \cdot e_1).
\end{aligned}$$

By the induction hypothesis on e_1, $\delta(q_0, s_1 \cdot a \cdot e_1)$ is in F if and only if $T(s_1 \cdot a \cdot e_1)=1$. Since $\text{row}(s)=\text{row}(s_1)$ and $a \cdot e_1 = e$ is in E, $T(s_1 \cdot a \cdot e_1)=T(s \cdot a \cdot e_1)=T(s \cdot e)$. Hence $\delta(q_0, s \cdot e)$ is in F and only if $T(s \cdot e)=1$, as claimed.

LEMMA 4. *Assume that (S, E, T) is a closed, consistent observation table. Suppose the acceptor $M(S, E, T)$ has n states. If $M'=(Q', q_0', F', \delta')$ is any acceptor consistent with T that has n or fewer states, then M' is isomorphic to $M(S, E, T)$.*

We prove this lemma by exhibiting an isomorphism. Define, for each q' in Q', $\text{row}(q')$ to be the finite function f from E to $\{0, 1\}$ such that $f(e)=1$ if and only if $\delta'(q', e)$ is in F'.

Note that since M' is consistent with T, for each s in $(S \cup S \cdot A)$ and each e in E, $\delta'(q_0', s \cdot e)$ is in F' if and only if $T(s \cdot e)=1$, which means that $\delta'(\delta'(q_0, s), e)$ is in F' if and only if $T(s \cdot e)=1$, so $\text{row}(\delta'(q_0', s))$ is equal to $\text{row}(s)$ in $M(S, E, T)$. Hence as s ranges over all of S, $\text{row}(\delta'(q_0, s))$ ranges over all the elements of Q, so M' must have at least n states, i.e., it must have exactly n states.

Thus, for each s in S there is a unique q' in Q' such that $\text{row}(s)=\text{row}(q')$, namely, $\delta'(q_0', s)$. Define for each s in S, $\phi(\text{row}(s))$ to be $\delta'(q_0', s)$. This mapping is one-to-one and onto. We must verify that it carries q_0 to q_0', that it preserves the transition function, and that it carries F to F'.

To see that $\phi(q_0)=q'_0$, note the following:

$$\begin{aligned}\phi(q_0)&=\phi(\text{row}(\lambda))\\&=\delta'(q'_0,\lambda)\\&=q'_0.\end{aligned}$$

For each s in S and a in A, let s_1 be an element of S such that $\text{row}(s\cdot a)=\text{row}(s_1)$. Then

$$\begin{aligned}\phi(\delta(\text{row}(s),a))&=\phi(\text{row}(s\cdot a))\\&=\phi(\text{row}(s_1))\\&=\delta'(q'_0,s_1).\end{aligned}$$

Also,

$$\begin{aligned}\delta'(\phi(\text{row}(s)),a)&=\delta'(\delta'(q'_0,s),a)\\&=\delta'(q'_0,s\cdot a).\end{aligned}$$

Since $\delta'(q'_0,s_1)$ and $\delta'(q'_0,s\cdot a)$ have identical row values, namely $\text{row}(s_1)$ and $\text{row}(s\cdot a)$, they must be the same state of M', so we conclude that $\phi(\delta(\text{row}(s),a))=\delta'(\phi(\text{row}(s)),a)$ for all s in S and a in A.

To conclude the proof of isomorphism of $M(S,E,T)$ and M', we must see that ϕ maps F to F', but this is clear since if s in S has $\text{row}(s)$ in F, then $T(s)=1$, so since $\phi(\text{row}(s))$ is mapped to a state q' with $\text{row}(q')=\text{row}(s)$, it must be that q' is in F'. Conversely, if $\text{row}(s)$ is mapped to a state q' in F', then since $\text{row}(q')=\text{row}(s)$, $T(s)=1$, so $\text{row}(s)$ is in F.

This concludes the proof of Lemma 4 and also the proof of Theorem 1, since Lemma 3 shows that $M(S,E,T)$ is consistent with T, and Lemma 4 shows that any other acceptor consistent with T is either isomorphic to $M(S,E,T)$ or contains at least one more state. Thus, $M(S,E,T)$ is the uniquely smallest acceptor consistent with T.

2.2. *The Learner L^**

The Learner L^* maintains an observation table (S,E,T). A summary of L^* is given in Fig. 1.

Initially $S=E=\{\lambda\}$. To determine T, L^* asks membership queries for λ and each a in A. This initial observation table may or may not be closed and consistent.

The main loop of L^* tests the current observation table (S,E,T) to see if it is closed and consistent. If (S,E,T) is not consistent, then L^* finds s_1 and s_2 in S, e in E, and a in A such that $\text{row}(s_1)=\text{row}(s_2)$ but $T(s_1\cdot a\cdot e)$ is not equal to $T(s_2\cdot a\cdot e)$. L^* adds the string $a\cdot e$ to E and extends T to

```
Initialize $S$ and $E$ to $\{\lambda\}$.
Ask membership queries for $\lambda$ and each $a \in A$.
Construct the initial observation table $(S, E, T)$.

Repeat:
    While $(S, E, T)$ is not closed or not consistent:
        If $(S, E, T)$ is not consistent,
          then find $s_1$ and $s_2$ in $S$, $a \in A$, and $e \in E$ such that
          $row(s_1) = row(s_2)$ and $T(s_1 \cdot a \cdot e) \neq T(s_2 \cdot a \cdot e)$,
          add $a \cdot e$ to $E$,
          and extend $T$ to $(S \cup S \cdot A) \cdot E$ using membership queries.
        If $(S, E, T)$ is not closed,
          then find $s_1 \in S$ and $a \in A$ such that
          $row(s_1 \cdot a)$ is different from $row(s)$ for all $s \in S$,
          add $s_1 \cdot a$ to $S$,
          and extend $T$ to $(S \cup S \cdot A) \cdot E$ using membership queries.

    Once $(S, E, T)$ is closed and consistent, let $M = M(S, E, T)$.
    Make the conjecture $M$.
    If the Teacher replies with a counter-example $t$, then
        add $t$ and all its prefixes to $S$
        and extend $T$ to $(S \cup S \cdot A) \cdot E$ using membership queries.
Until the Teacher replies *yes* to the conjecture $M$.
Halt and output $M$.
```

FIG. 1. The Learner L^*.

$(S \cup S \cdot A) \cdot (a \cdot e)$ by asking membership queries for missing elements. (Note that in this case, s_1, s_2, e, and a must exist, and since e is in E, E remains suffix-closed when $a \cdot e$ is added.)

If (S, E, T) is not closed, then L^* finds s_1 in S and a in A such that row$(s_1 \cdot a)$ is different from row(s) for all s in S. L^* adds the string $s_1 \cdot a$ to S and extends T to $(S \cup S \cdot A) \cdot E$ by asking membership queries for missing elements. (Note that in this case, s_1 and a must exist, and since s_1 is in S, S remains prefix-closed when $s_1 \cdot a$ is added.)

These two operations are repeated as long as the table (S, E, T) is not closed and consistent. When (S, E, T) is found to be closed and consistent, L^* makes a conjecture of $M(S, E, T)$. The Teacher replies either with *yes*, signifying that the conjecture is correct, or with a counterexample t. If the Teacher replies with *yes*, L^* terminates with output $M(S, E, T)$. If the Teacher replies with a counterexample t, then t and all its prefixes are added to the set S and the function T is extended to $(S \cup S \cdot A) \cdot E$ by means of membership queries for the missing entries. The main loop of the algorithm is then repeated for this new observation table (S, E, T).

Correctness of L^*. To see that L^* is correct, note that if the Teacher is minimally adequate then if L^* ever terminates its output is clearly an acceptor for the unknown regular set U being presented by the Teacher.

Termination of L^*. To see that L^* terminates, we need a simple lemma.

LEMMA 5. *Let (S, E, T) be an observation table. Let n denote the number of different values of row(s) for s in S. Any acceptor consistent with T must have at least n states.*

Let $M = (Q, \delta, q_0, F)$ be any acceptor consistent with T. Define $f(s) = \delta(q_0, s)$ for every s in S. Suppose s_1 and s_2 are elements of S such that row(s_1) and row(s_2) are distinct. Then there exists e in E such that $T(s_1 \cdot e) \neq T(s_2 \cdot e)$. Since M is consistent with T, exactly one of $\delta(q_0, s_1 \cdot e)$ and $\delta(q_0, s_2 \cdot e)$ is in F. Thus, $\delta(q_0, s_1)$ and $\delta(q_0, s_2)$ must be distinct states. Thus, $f(s)$ takes on at least n different values as s ranges over S, so M must have at least n states, proving Lemma 5.

Now suppose that n is the number of states in the minimum acceptor M_U for the unknown regular set U. We show that the number of distinct values of row(s) for s in S increases monotonically up to a limit of n as L^* runs. (Note that at every iteration of the main loop, (S, E, T) is indeed an observation table.)

Suppose a string is added to E because the table is not consistent. Then the number of distinct values row(s) for s in S must increase by at least one, because two previously equal values, row(s_1) and row(s_2), are no longer equal after E is augmented. (Note that two values unequal before E is augmented remain unequal after E is augmented.)

Suppose a string $s_1 \cdot a$ is added to S because the table is not closed. Then by definition, row$(s_1 \cdot a)$ is different from row(s) for all s in S before S is augmented, so the number of distinct values row(s) is increased by at least one when $s_1 \cdot a$ is added to S.

Thus, the total number of operations of either type over the whole run of the algorithm L^* must be at most $n-1$, since there is initially at least one value of row(s) and there cannot be more than n. Hence L^* always eventually finds a closed, consistent observation table (S, E, T) and makes a conjecture $M(S, E, T)$.

How many distinct conjectures can L^* make? If a conjecture $M(S, E, T)$ is found to be incorrect by the counterexample t, then since the correct minimum acceptor M_U is consistent with T and inequivalent to $M(S, E, T)$ (since they disagree on t), by Theorem 1, M_U must have at least one more state. That is, $M(S, E, T)$ has at most $n-1$ states. Furthermore, L^* must eventually make a next conjecture, $M(S', E', T')$, which is consistent with T (since T' extends T) and also classifies t the same as M_U (since t is in S' and λ is in E), and so is inequivalent to $M(S, E, T)$. Thus, $M(S', E', T')$ must have at least one more state than $M(S, E, T)$.

This shows that L^* can make a sequence of at most $n-1$ incorrect conjectures, since the number of their states must be monotonically increasing, is initially at least one, and may not exceed $n-1$. Since L^* must, as long as it is running, eventually make another conjecture, it must terminate by making a correct conjecture.

Thus, L^* correctly terminates after making at most n conjectures and executing its main loop a total of at most $n-1$ times.

Time analysis of L^*. How much time does L^* consume? That depends partly on the length of the counterexamples t presented by the Teacher. If long counterexamples are given, it will take correspondingly long to process them. Hence we analyze the running time of L^* as a function of n, the number of states in the minimum acceptor M_U of the unknown regular set, and m, the maximum length of any counterexample string presented by the Teacher during the running of L^*.

We describe a straightforward implementation of L^* to show that its running time is bounded by a polynomial in m and n. Let k denote the cardinality of the alphabet A.

Initially S and E each contain one element. Each time (S, E, T) is discovered to be not closed, one element is added to S. Each time (S, E, T) is discovered to be not consistent, one element is added to E. For each counterexample t of length at most m supplied by the Teacher, at most m strings are added to S. (Note that λ is already in S.)

Thus, the total number of strings in E cannot exceed n, since the observation table is discovered to be not consistent at most $n-1$ times. The maximum length of any string in E is initially zero and is increased by at most one with every string added to E, so the maximum length of any string in E is $n-1$.

The total number of strings in S cannot exceed $n+m(n-1)$. This is because the observation table is discovered to be not closed at most $n-1$ times, and there can be at most $n-1$ counterexamples, each of which may cause at most m strings to be added to S. The maximum length of any string in S is increased by at most one with every string added because the observation table is not closed. Thus, the maximum length of any string in S is at most $m+n-1$.

Putting these together, the maximum cardinality of $(S \cup S \cdot A) \cdot E$ is at most

$$(k+1)(n+m(n-1))\,n = O(mn^2).$$

The maximum length of any string in $(S \cup S \cdot A) \cdot E$ is at most

$$m+2n-1 = O(m+n).$$

Thus, the observation table can be explicitly represented by a finite table of size polynomial in m and n ($O(m^2n^2+mn^3)$ suffices.)

Now consider the operations performed by L^*. Checking the observation table to be closed and consistent can be done in time polynomial in the size of the observation table and must be done at most $n-1$ times. Adding a string to S or E requires at most $O(mn)$ membership queries of strings of length at most $O(m+n)$ to find the missing entries in the table. When the observation table is closed and consistent, $M(S, E, T)$ may be constructed in time polynomial in the size of the observation table, and this must be done at most n times. A counterexample requires the addition of at most m strings of length at most m to S, and this can happen at most $n-1$ times. Thus, the total running time of L^* can be bounded by a polynomial function of m and n.

We summarize the results concerning L^* in the following theorem.

THEOREM 6. *Given any minimally adequate Teacher presenting an unknown regular set U, the Learner L^* eventually terminates and outputs an acceptor isomorphic to the minimum dfa accepting U. Moreover, if n is the number of states of the minimum dfa accepting U and m is an upper bound on the length of any counterexample provided by the Teacher, then the total running time of L^* is bounded by a polynomial in m and n.*

One corollary of this is that if the Teacher always presents counterexamples of minimal length, then they wil be at most n in length and L^* will run in time polynomial in n. The polynomial time Teacher T^* of Section 1.2 may be constructed to present counterexamples of minimal length, in which case the total running time of L^* and T^* together will be polynomial in n.

3. AN EXAMPLE RUN OF L^*

Suppose the unknown regular set U is the set of all strings over $\{0, 1\}$ with an even number of 0's and an even number of 1's.

Initially, L^* asks membership queries for the strings λ, 0, and 1. The initial observation table T_1 is shown in Fig. 2. This observation table is consistent, but not closed, since row(0) is distinct from row(λ).

L^* chooses to move the string 0 to the set S and then queries the strings 00 and 01 to construct the observation table T_2 shown in Fig. 3. This observation table is closed and consistent, so L^* makes a conjecture of the acceptor M_1, shown in Fig. 4. The initial state of M_1 is q_0 and the final state is also q_0. M_1 is not a correct acceptor for U, so the Teacher selects a

T_1	λ
λ	1
0	0
1	0

FIG. 2. Initial observation table, $S = E = \{\lambda\}$.

T_2	λ
λ	1
0	0
1	0
00	1
01	0

FIG. 3. Augmented observation table, $S = \{\lambda, 0\}$, $E = \{\lambda\}$.

counterexample. In this case we assume that the counterexample 11 is selected; it is in U but rejected by M_1.

To process the counterexample 11, L^* adds the strings 1 and 11 to S (the string λ is already in S), and queries the strings 10, 110, and 111 to construct the observation table T_3 shown in Fig. 5. This observation table is closed but not consistent since row(0) = row(1) but row(00) $\neq$ row(10).

Thus L^* adds the string 0 to E, and queries the strings 000, 010, 100, 1100, and 1110 to construct the observation table T_4 shown in Fig. 6. This observation table is closed and consistent, so L^* conjectures the acceptor M_2 shown in Fig. 7. The initial state of M_2 is q_0 and the final state is also q_0. M_2 is not a correct acceptor for the set U, so the Teacher answers the conjecture with a counterexample. We suppose that the counterexample supplied is 011, which is not in U but is accepted by M_2.

L^* responds to this counterexample by adding the strings 01 and 011 to S. (The other prefixes of 011, λ and 0, are already members of S.) L^* then queries the strings 0110, 0100, 01100, 0111, and 01110 to construct the observation table T_5 shown in Fig. 8. This table is found to be closed but not consistent, since row(1) = row(01) but row(11) $\neq$ row(011).

Thus L^* adds the string 1 to E and queries the strings 001, 101, 1101, 1111, 0101, 01101, and 01111 to construct the observation table T_6 shown in Fig. 9. This table is closed and consistent, so L^* conjectures the acceptor M_3 shown in Fig. 10. The initial state of M_3 is q_0 and the final state is also q_0. M_3 is a correct acceptor for the language U, so the Teacher replies to this conjecture with *yes* and L^* terminates with M_3 as its output.

The total number of membership queries during this run of L^* is 25. L^* makes two incorrect conjectures and one correct conjecture. The number of membership queries seems rather large for a practical system. One open problem is to find a method to reduce substantially the number of membership queries required.

δ	0	1
q_0	q_1	q_1
q_1	q_0	q_1

FIG. 4. M_1, the first conjecture of L^*.

T_3	λ
λ	1
0	0
1	0
11	1
00	1
01	0
10	0
110	0
111	0

FIG. 5. Observation table, $S=\{\lambda, 0, 1, 11\}$, $E=\{\lambda\}$.

4. Finding Counterexamples Using Sampling

As defined, a minimally adequate Teacher must be able to test an arbitrary conjecture, reply whether the conjecture is correct, and supply a counterexample if not. As previously mentioned, this seems too restrictive to require more general domains. If we instead postulate a stochastic setting analogous to that proposed by Valiant [8] and require only that the Learner get an approximately correct hypothesis with high probability, then this aspect of the Teacher can be replaced by a random sampling oracle.

4.1. *The Setting*

For the stochastic setting, we assume that there is some probability distribution *Pr* on the set of all strings over the alphabet A. We do not assume that this distribution is known to the Learner.

There is an unknown regular set U over the alphabet A. The Learner has access to information about U by means of two oracles. The first is just the membership oracle, MEMBER(x), that returns *yes* if x is an element of U

T_4	λ	0
λ	1	0
0	0	1
1	0	0
11	1	0
00	1	0
01	0	0
10	0	0
110	0	1
111	0	0

FIG. 6. $S=\{\lambda, 0, 1, 11\}$, $E=\{\lambda, 0\}$.

δ	0	1
q_0	q_1	q_2
q_1	q_0	q_2
q_2	q_2	q_0

FIG. 7. M_2, the second conjecture of L^*.

and *no* otherwise. The second is a random sampling oracle, $EX(\ )$, that selects a string x from A^* according to the distribution Pr and returns the pair (x, d), where d is *yes* if x is in U and *no* otherwise. Separate calls to $EX(\ )$ are statistically independent events.

In addition, the Learner is given, at the start of the computation, two positive numbers between 0 and 1, the *accuracy* ε and the *confidence* δ.

4.2. *Approximation*

Let ε denote a positive number between 0 and 1, and let V and W denote sets of strings over the alphabet A. V is an *ε-approximation* of W provided that

$$\sum_{x \in V \oplus W} Pr(x) \leqslant \varepsilon.$$

($V \oplus W$ denotes the symmetric difference of V and W, that is, the elements that are in exactly one of the two sets). If M is a dfa, it is said to be an *ε-approximation* of the set W provided the set of strings V accepted by M is an ε-approximation of W.

T_5	λ	0
λ	1	0
0	0	1
1	0	0
11	1	0
01	0	0
011	0	1
00	1	0
10	0	0
110	0	1
111	0	0
010	0	0
0110	1	0
0111	0	0

FIG. 8. $S = \{\lambda, 0, 1, 11, 01, 011\}$, $E = \{\lambda, 0\}$.

T_6	λ	0	1
λ	1	0	0
0	0	1	0
1	0	0	1
11	1	0	0
01	0	0	0
011	0	1	0
00	1	0	0
10	0	0	0
110	0	1	0
111	0	0	1
010	0	0	1
0110	1	0	0
0111	0	0	0

FIG. 9. $S=\{\lambda, 0, 1, 11, 01, 011\}$, $E=\{\lambda, 0, 1\}$.

If M is an ε-approximation of the unknown regular set U, then the probability of finding a discrepancy between the set accepted by M and U with one call to the random sampling oracle *EX*( ) is at most ε.

4.3. *The Approximate Learner* L_a^*

The approximate Learner L_a^* is obtained by modifying L^*. A membership query of the string x is satisfied by a call to $MEMBER(x)$. Each conjecture is tested by a number of calls to *EX*( ). If any of the calls to *EX*( ) returns a pair (t, d) such that $d=yes$ but $M(S, E, T)$ rejects t or vice versa, then t is said to be a counterexample, and L_a^* proceeds to modify its hypothesis as L^* does for the counterexample t. If none of the calls to *EX*( ) returns a counterexample, then L_a^* halts with output $M(S, E, T)$.

How many calls to *EX*( ) does L_a^* make to test a given conjecture? That depends on the accuracy and confidence parameters, ε and δ, as well as how many previous conjectures have been tested. Let

$$r_i=\frac{1}{\varepsilon}\left(\log\frac{1}{\delta}+(\log 2)(i+1)\right).$$

If i previous conjectures have been tested, then L_a^* makes $\lceil r_i \rceil$ calls to *EX*( ). (Note: log denotes the natural logarithm.)

The main result concerning L_a^* is the following.

δ	0	1
q_0	q_1	q_2
q_1	q_0	q_3
q_2	q_3	q_0
q_3	q_2	q_1

FIG. 10. M_3, the third conjecture of L^*.

THEOREM 7. *If n is the number of states in the minimum dfa for the unknown regular set U, then L_a^* terminates after $O(n+(1/\varepsilon)(n\log(1/\delta)+n^2))$ calls to the EX( ) oracle. Moreover, the probability that the acceptor output by L_a^* is an ε-approximation of U is at least $1-\delta$.*

To see that this holds, note that as before there will be a total of at most $n-1$ counterexamples. The total number of calls to *EX*( ) before L_a^* terminates is at most

$$\sum_{i=0}^{n-2}(r_i+1)=(n-1)+\frac{1}{\varepsilon}\left((n-1)\log\left(\frac{1}{\delta}\right)+(\log 2)\sum_{i=0}^{n-2}(i+1)\right)$$
$$=O\left(n+\frac{1}{\varepsilon}\left(n\log\left(\frac{1}{\delta}\right)+n^2\right)\right).$$

What is the probability that L_a^* will terminate with an acceptor that is not an ε-approximation of U after testing i previous conjectures? It is at most $(1-\varepsilon)^{r_i}$, so the probability that L_a^* terminates with an acceptor that is not an ε-approximation of U is at most

$$\begin{aligned}\sum_{i=0}^{n-2}(1-\varepsilon)^{r_i}&\leqslant\sum_{i=0}^{n-2}e^{-\varepsilon r_i}\\&\leqslant\sum_{i=0}^{n-2}\frac{\delta}{2^{i+1}}\\&\leqslant\delta.\end{aligned}$$

This concludes the proof of Theorem 7.

The running time of L_a^*. The running time of L_a^* can be analyzed analogously to that of L^*. Namely, if n is the number of states of the minimum dfa for U and m is an upper bound on the length of any string returned by *EX*( ) during the run, then the running time of L_a^* is bounded by a polynomial in m and n. If the distribution *Pr* is such that it concentrates a lot of probability on very long strings (compared to n), then this is not much of a guarantee.

One thing that can be done to mitigate this problem somewhat (especially if the formal model allows the Learner to determine the length of and discard long sample strings in sublinear time), is to have L_a^* do a preliminary sampling from *EX*( ) to determine a cutoff length l such that with probability at least $(1-\delta/2)$ the weight (with respect to *Pr*) of strings longer than l is at most $\varepsilon/2$. Then L_a^* runs as before, replacing the parameters ε and δ by $\varepsilon/2$ and $\delta/2$, and ignoring any strings returned by *EX*( ) that are longer than l. The output will with probability at least $(1-\delta)$ be an acceptor that is an $\varepsilon/2$-approximation of an $\varepsilon/2$-

approximation of U. The termination claim and argument must be modified somewhat to account for the (unlikely) possibility that $EX(\ )$ will return many strings of length greater than l, and of which are ignored. Standard statistical analysis shows that the number of samples that L_a^* must take in the preliminary phase is bounded by a polynomial in $1/\varepsilon$ and $1/\delta$.

Note that this general approach, of replacing the Teacher's replies to conjectures by statitical sampling, does not depend on the domain being the regular sets. It is applicable whenever the problem of testing a conjecture against randomly drawn samples is sufficiently tractable.

5. Context-free Grammars, a Special Case

The idea of a minimally adequate Teacher is related to Shapiro's assumptions about the kinds of queries a user can answer in Prolog program debugging [7]. In addition to being able to answer membership queries and provide counterexamples, Shapiro assumes the ability to answer "existential queries" and "termination queries."

To give an example of how the ideas of the present paper extend to some of the domains considered by Shapiro, we describe a Learner L^{cf} for a class of context-free languages.

5.1. *The Setting*

There is an unknown context-free grammar G in Chomsky normal form. The Learner knows the set T of terminal symbols, the set V of nonterminal symbols, and the start symbol S of G.

A minimally adequate Teacher in this domain is assumed to be able to answer two types of questions. The first type is a membership query, $MEMBER(x, A)$, where x is a string of terminals and A is a nonterminal symbol. The Teacher determines whether the string x can be derived from A using the rules of G, answering *yes* if so and *no* if not. The other type of question is a conjecture, EQUIV(H), where H is a context-free grammar. The Teacher determines whether H is equivalent to G (i.e., they generate the same sets of terminal strings from the start symbol S.) If so, it replies *yes*, otherwise, it replies with a counterexample t that is generated by G but not H or vice versa. (This second type of query can be replaced by sampling as in the preceding section.)

5.2. *The Learner* L^{cf}

Note that the restriction to grammars in Chomsky normal form and the Learner's knowledge of T and V imply that the L^{cf} can explicitly enumerate all the possible productions of G in time bounded by a polynomial in the cardinalities of T and V.

Initially, L^{cf} places all the possible productions of G in the hypothesized set of productions P. The main loop of L^{cf} asks an EQUIV(H) question for the grammar H with terminals T, nonterminals V, start symbol S, and productions P. If H is equivalent to G, then L^{cf} halts with output H. Otherwise, it "diagnoses" the counterexample t returned, which results in at least one production being removed from P. The main loop is then repeated.

The process of "diagnosis" is special case of Shapiro's general diagnosis algorithm. In particular, the set P will always contain the productions of G as a subset, so the grammar H will always generate a superset of the strings generated by G. Hence the only type of counterexample is one that is generated by H but not by G.

To diagnose the counterexample t, L^{cf} finds a parse tree exhibiting a derivation of t from S using the productions in P. The root of the parse tree is labelled by S. Suppose it has two sons, labelled A and B, and the frontier of the subtree rooted at A is t_1 and the frontier of the subtree rooted at B is t_2. Clearly, $t = t_1 \cdot t_2$.

L^{cf} then asks the question MEMBER(t_1, A). If the reply is *no*, then t_1 can be derived from A in H but not in G and L^{cf} recursively continues to diagnose the subtree of the parse tree rooted at A. If the answer is *yes*, then L^{cf} asks the question MEMBER(t_2, B). If the reply is *no*, it recursively continues to diagnose the subtree of the parse tree rooted at B. If the answer is *yes*, then we have the fact that in G, t_1 may be derived from A and t_2 from B, but t may not be derived from S. Hence the production $S \rightarrow AB$ cannot be in the grammar G, and we remove it from P and return.

The recursive calls must eventually find a production of the form $A \rightarrow BC$ to discard from P, or arrive at a node labeled by a nonterminal A with only one son, labelled by a terminal symbol a. In this case, we have just asked MEMBER(a, A) and found that A does not generate a in G, so the production $A \rightarrow a$ may be removed from P.

In either case, the diagnosis procedure eventually finds a production in P that cannot be in G and removes it from P. Thus the condition that P contains the productions of G as a subset is preserved.

The correctness and termination of L^{cf} are immediate.

Time analysis of L^{cf}. Every counterexample causes the elimination from P of another production that is not in G, which means that after at most $|P|$ counterexamples, L^{cf} must find a grammar equivalent to G and halt. The processing of each counterexample may be done in time polynomial in $|P|$ and the length of the counterexample. Since $|P|$ is bounded by a polynomial in $|T|$ and $|V|$, this means that the running time of L^{cf} is bounded by a polynomial in $|T|$, $|V|$, and the maximum length of any counterexample provided during the run.

5.3. *Comments*

The Learner L^{cf} is considerably simpler than the Learner L^*. This is because the information correspond to the nonterminal set V and the "enriched" membership questions MEMBER(x, A) are not available in the setting of L^*.

Note that the MEMBER(x, A) questions may be answered by a computation polynomial in the length of x and the size of G. However, to answer the question EQUIV(H) is not in general decidable, so there is no computable implementation of a minimally adequate Teacher for this domain. (Hence the stochastic setting may be preferable.)

Moreover, in contrast to the case of dfa's, the shortest counterexample will not in general be bounded by a polynomial in $|T|$ and $|V|$. (It is not difficult to construct a grammar of size polynomial in n that generates each string of 1's whose length is not a positive multiple of $n!$. For this case, the smallest counterexample to the initial conjecture (which generates all strings) is a string of length $n!$.)

The choice of Chomsky normal form for G guarantees that initially $|P|$ is bounded by a polynomial in $|T|$ and $|V|$. Any other class of grammars with this property would work as well. In this case, convergence is by means of a decreasing chain of subsets to a correct hypothesis.

6. Open Problems

Can we substantially reduce the number of membership queries used by L^*? Can we do without membership queries altogether for either the deterministic or the stochastic settings for regular sets and still infer them efficiently? ?

An interesting domain proposed by Phil Laird is that of propositional Horn sentences. (That is, propositional sentences in conjunctive normal form with at most one positive literal per clause.) The satisfiability and equivalence problems for these sentences can be solved in polynomial time, making them an attractive class [6].

A minimally adequate Teacher for propositional Horn sentences must answer two types of questions. A membership query consists of an assignment of truth-values to the (known) variables of the unknown sentence ϕ, and the Teacher replies according to whether this assignment satisfies ϕ. A conjecture consists of a Horn sentence ψ, and the Teacher tests the equivalence of ϕ and ψ and replies either with *yes* or an assignment of truth-values that satisfies one of the sentences but not the other. There is a polynomial-time algorithm (in the sizes of ϕ and ψ) to implement a minimally adequate Teacher in this domain.

An interesting open problem is whether there is a Learner L^h that can learn any propositional Horn sentence ϕ from any minimally adequate Teacher of propositional Horn sentences in time polynomial in the size of ϕ. (So far, I have only been able to find such an algorithm with some annoying additional restrictions on the Teacher.)

Acknowledgments

Discussions with Phil Laird, Jingke Li, David Haussler, and Manuel Blum have been quite helpful in formulating these ideas. The support of the National Science Foundation grant IRI-8404226 is gratefully acknowledged.

Received April 30, 1986; accepted November 25, 1986

References

1. Angluin, D. (1982), A note on the number of queries needed to identify regular languages, *Inform. Contr.* **51**, 76–87.
2. Angluin, D., and Smith, C. (1983), Inductive inference: Theory and methods, *Comput. Surveys* **15**, 237–269.
3. Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. (1986), Classifying learnable geometric concepts with the Vapnik-Chervonenkis dimension, *in* "Proceedings, Eighteenth Annual ACM Symposium on Theory of Computing," pp. 273–282, Assoc. Comput. Mach., New York.
4. Gold, E. M. (1978), Complexity of automaton identification from given data, *Inform. Contr.* **37**, 302–320.
5. Gold, E. M. (1972), System identification via state characterization, *Automatica* **8**, 621–636.
6. Jones, N. D., and Laaser, W. T. (1977), Complete problems for deterministic polynomial time, *Theoret. Comput. Sci.* **3**, 107–113.
7. Shapiro, E. (1982), "Algorithmic Program Debugging," Ph.D. thesis, Yale University Computer Science Dept.; (1983), published by MIT Press, Cambridge, MA.
8. Valiant, L. G. (1984), A theory of the learnable, *Comm. ACM* **27**, 1134–1142.