Computer Science Department Technical Report
Cognitive Systems Laboratory
University of California
Los Angeles, CA 90024-1596

MEMBERSHIP ALGORITHM FOR MARGINAL INDEPENDENCIES

Azaria Paz

November 1988
CSD-880095

TECHNICAL REPORT
R-117
September 1988

MEMBERSHIP ALGORITHM FOR MARGINAL INDEPENDENCIES *

Azaria Paz

Cognitive Systems Laboratory
Computer Science Department
Los Angeles, CA 90024-1596

and

Technion IIT, Haifa, Israel

ABSTRACT

A quadratic algorithm is given for the membership problems of Independency models induced by marginal probabilistic distributions. The algorithm is very simple and easy to implement.

* This work was supported in part by the National Science Foundation Grant # IRI-8610155.

1. INTRODUCTION

It has been shown by Geiger and Pearl [GP] that a set of 3 axioms exists which is sound and complete for Independency models derived from marginal probabilistic distributions (including the case where the numerical entries in the distribution tables are strictly positive and including the case where the random variables involved in the distributions have binary domains). These axioms are defined in Section 2.

It has also been shown in that paper that the family of marginal-probabilistic-distribution-induced independency models is superexponential (i.e., the number of different such models is $0(2^{2^n})$) implying that the number of independencies included in any particular model may be exponential.

The above mentioned fact raised the question whether a polynomial algorithm exists for the membership problem for such models, i.e. given a polynomial set of independencies induced by a marginal probabilistic distribution and some statement T, the alagorithm should find, in polynomial time, whether T is in the closure of the given set under th 3 above mentioned axioms.

A positive answer for this question is provided in this paper.

2. THE PROBLEM

Find a polynomial algorithm whose input is: A set Σ, of polynomial size, of marginal statements

$$\Sigma = \{(x_i, y_i) : 0 \le i \le poly(n)\}$$

and a marginal (target) statement $T = (u, v)$, and whose output is:

Yes if $T \in \mathbf{cl}(\Sigma)$

No if $T \notin \mathbf{cl}(\Sigma)$

where **cl** (Σ) is the closure of Σ under the three axioms below.

1. $(x, y) \rightarrow (y, x)$ Symmetry

2. $(x, yz) \rightarrow (x, y)$ Decomposition

3. (x, y) & $(xy, z) \rightarrow (x, yz)$ Mixing

In the following section a polynomial algorithm with complexity $O(n\,\mathrm{poly}(n))$ where $|\Sigma|$ = (the size of the set Σ) = poly (n) is given. The complexity is defined in terms of "basic" operation where a basic operation is the operation of comparing two statements, or the operation of deleting some variables from a statement. We need first

Some Notations and Definitions

— $E(d)$ is the set of elements represented in the pair d.

— $E(\Sigma)$ is the set of elements represented in all the pairs in Σ, similarly for any set of pairs D we can define $E(D)$.

— If $d = (x, y)$ is a statement and s is a set of elements then $d'(s)$ (the *reduction* of d by s) is the statement derived from d by removing all the elements in s from d.
Thus if d = (123,45) then $d'(45) = (123, \phi)$ and $d'(25) = (13, 4)$

— All pairs of the form (x, ϕ) are termed *trivial* and trivially belong to **cl** (Σ). Unless otherwise stated all pairs considered will be nontrivial.

— $\Sigma'(s) = \{d'(s) : d \in \Sigma\}$

— $|D|$ = "the size of D" = the number of elements in D.

3. THE ALGORITHM (INPUT Σ, T)

Find (Σ, T):

1. Let s be the set of elements

$$s = E(\Sigma) - E(T).$$

Construct $\Sigma'(s)$

2. If T is "trivial," or for some statement S in $\Sigma'(s)$ we have that $S = T$, up to symmetry, then return.

Find (Σ, T) := True

3. Else if $E(d) \neq E(T)$ for all nontrivial $d \in \Sigma'(s)$ then return.

Find (Σ, T) := False

4. Else there exists a statement S in $\Sigma'(s)$ such that $E(S) = E(T)$, and up to symmetry, $S = (AP, BQ)$ $T = (AQ, BP)$ where one of the sets A, B, P, Q may be empty.

If several such S exist then choose any one such S.

Set $T_1 := (A, P)$, $T_2 = (B, Q)$, and return.

Find (Σ, T) := Find $(\Sigma, T_1) \wedge$ Find (Σ, T_2).

We will show first that the algorithm is correct and then prove its complexity.

Lemma 1: If $d \in \mathbf{cl}(\Sigma)$ and $s \subset E(\Sigma)$ then $d'(s) \in \mathbf{cl}(\Sigma)$.

Proof: Follows from the decomposition and symmetry axioms.

Lemma 2: If $d_1 \ldots d_k$ is a proper derivation chain (i.e. for $1 \leq i \leq k$, d_i is either in Σ or follows from previous d_j's by one of the axioms) and $s \subset E(\Sigma)$ then $d_i'(s), \ldots d_k'(s)$ is also a proper derivation chain.

Proof: Follows from Lemma 1 and from the fact that the axioms are preserved under the reduction operation.

Lemmas 1 and 2 justify Steps 2 and 3 of the algorithm. To justify Step 4 we will first prove several lemmas.

Lemma 3: Let $S = (AP, BQ), T_1 = (A, P), T_2 = (B, Q)$ be statements. If S, T_1, T_2 are in $\mathbf{cl}(\Sigma)$. Then the statement $T = (AQ, BP)$ is in $\mathbf{cl}(\Sigma)$ and all the statements (A_1, P_1), (B_1, Q_1) such that $A_1Q_1 = AQ$ and $B_1P_1 = BP$ are in $\mathbf{cl}(\Sigma)$.

Proof: The derivation chain below is a proper derivation chain for T:

1. $(AP, BQ) = S$; $(A, P) = T_1$; $(B, Q) = T_2$
2. (A, PBQ) : Mix S with T_1
3. (APB, Q) : Mix S with T_2
4. (PB, Q) : Decomposition from 3
5. $(AQ, BP) = T$: Mix 2 with 4

Now $(AQ, BP) = (A_1Q_1, B_1P_1)$. Delete all the elements of Q_1 and B_1 from the above deriva-

tion chain. The result will be a proper derivation chain for (A_1, P_1) (not passing through T). Delete all the elements of A_1 and P_1 from the above derivation chain. The result will be a proper derivation chain for (B_1, Q_1) (not passing through T). □

Lemma 4: If $S \in \mathbf{cl}(\Sigma)$ then $T \in \mathbf{cl}(\Sigma)$ iff $T_1 \in \mathbf{cl}(\Sigma)$ and $T_2 \in \mathbf{cl}(\Sigma)$ where S, T, T_1, T_2 are defined as in Step 4 of the algorithm.

Proof: If $S, T_1, T_2 \in \mathbf{cl}(\Sigma)$ then T can be derived as in the proof of the previous Lemma 3.

If $T \in \mathbf{cl}(\Sigma)$ then let

$d_1, d_2 \ldots, d_k = T = (AQ, BP)$ be a proper derivation chain for T. Let $s = E((Q, B))$. Then $d_1'(s), d_2'(s), \ldots d_k'(s) = T_1 = (A, P)$ is a proper derivation chain for T_1. Similarly a proper derivation chain for T_2 can be constructed. □

Lemma 5: $T \in \mathbf{cl}(\Sigma)$ iff Find (Σ, T) = true.

Proof: By induction on the size of T. If $|T| = 1$ then T is trivial, $T \in \mathbf{cl}(\Sigma)$ and Find $(\Sigma, T) =$ true. If $|T| = 2$ then $T \in \mathbf{cl}(\Sigma)$ Iff Find $(\Sigma, T)) =$ true as follows from Steps 2 and 3 of the algorithm.

Assume that the Lemma holds for all $|T'| < k$ and let T be a statement $|T| = k$,$T = (AQ, BP)$. Then Find (Σ, T) = true iff (by the definition of Step 4) Find (Σ, T_1) = true and Find (Σ, T_2) = true iff (by induction) $T_1, T_2 \in \mathbf{cl}(\Sigma)$ iff (by Lemma 4) $T \in \mathbf{cl}(\Sigma)$. □

The next Lemma shows that the algorithm is independent on the choice of S in its Step 4.

Lemma 6: The value of Find (Σ, T) computed in Step 4 of the algorithm does not depend on

the choice of the statement S.

Proof: Assume that for a given choice S we get Find (Σ, T) = true. Then, by Lemma 5, $T \in \mathbf{cl}(\Sigma)$ and also $T_1, T_2 \in \mathbf{cl}(\Sigma)$. If another possible choice in Step 4 exists, say S_1, $S_1 = (A_1P_1, B_1Q_1)$, (notice that $(A_1Q_1, B_1P_1) = T$)), then by Lemma 3 (A_1, P_1) and (B_1, Q_1) are also in $\mathbf{cl}(\Sigma)$.

This implies, by Lemma 5, that Find $(\Sigma, (A_1, P_1))$ = Find $(\Sigma, (B_1Q_1))$ = true, which implies that the value computed for Find (Σ, T) when the algorithm chooses S_1 instead of S is also "true." □

Theorem 7: The algorithm halts and when it halts it produces the correct answer.

Proof: Everytime the algorithm passes through Step 4 the size of the statements involved strictly decrease. If it did not halt beforfe, it will halt when the size of the two statements have reached the value 2 (at Step 2 or 3).

It follows from Lemmas 5 and 6 that when the algorithm halts it provides an exact and correct answer to the question whether $T \in \mathbf{cl}(\Sigma)$, namely, the answer is "yes" iff Find (Σ, T) = true. □

4. THE COMPLEXITY OF THE ALGORITHM

The procedure of the algorithm may be described as the counstruction of a binary tree whose root is labeled T and every 2 nodes emerging from a node T' are labeled T_1' and T_2' as defined in Step 4 of the algorithm. The leaves of the tree represent singleton statement pairs which must be checked against the elements of Σ.

Let n be the number of elements represented in T and let $M(n)$ be an upper bound on the number of nodes of the tree determinded by T. We verify first that $M(n)$ satisfies the recursion below

$$M(n) \leq M(n_1) + M(n_2) + 1$$

where $n_1 + n_2 = n$

This follows from Step 4 of the algorithm: We must count the root, (1), representing T plus the nodes of the two subtrees whose roots are T_1 and T_2. The number of vertices n_1 represented in T_1 is $|AP|$ and the number of vertices n_2 represented in T_2 is $|BQ|$. Thus, $n_1 + n_2 = n$.

It is easy to see that the above recursion is satisfied if $M(n) \leq n - 1$. Clearly $M(1) = 0 = 1 - 0$ (trivial statements need no processing) and $M(2) = 1 = 2 - 1$.

It follows that the number of nodes in the tree determined by a statement T of size n is bounded by $n - 1$. For every such node we will have to perform $0(|\Sigma|)$ operations (constructing $\Sigma'(s)$ and comparing T against the statements in $\Sigma'(s)$).

The above observations can be summarized in the theorem below.

Theorem 8: The complexity of the algorithm is $0(|\Sigma| \ n)$ measured in basic operations.

5. REMARKS:

1. It is reasonable to assume that the bound is pessimistic at least in its $|\Sigma|$ part, since as the algorithm proceeds the number of statements in the $\Sigma'(s)$ decreases.

2. The algorithm can be slightly modified so as to produce a derivation chain for T if $T \in \mathbf{cl}(\Sigma)$, whose length is $0(n)$.

3. The fact that the longest derivation chain for any statement of size n in $\mathbf{cl}(\Sigma)$ is bounded by n bounds the number of elements of size n in $\mathbf{cl}(\Sigma)$ as a function of $|\Sigma|$ and of n.

4. The algorithm can be expanded into a polynomial algorithm (provided that $|\Sigma|$ is polynomial) for the following problems:

   a. Given Σ and Σ', is $\mathbf{cl}(\Sigma) = \mathbf{cl}(\Sigma')$, or is $\mathbf{cl}(\Sigma) \subset \mathbf{cl}(\Sigma')$?

   b. Minimize the size of Σ while preserving $\mathbf{cl}(\Sigma)$:

      Start with 2 "independent" maximal size statements in Σ and probe all other statements in some non-increasing (as to size) order. Any statement which is found "dependent" on the set of previously probed statements is deleted.

References

[GP] Geiger, D., & J. Pearl, "Logical and Algorithmic Properties of Conditional Independence," UCLA Cognitive Systems Laboratory, *Technical Report 870056 (R-97)*, August 1987. To appear in *Annals of Mathematics and Artificial Intelligence* (2nd International Workshop on Artificial Intelligence and Statistics). January 1989.