Making Maximum Entropy Computations Easier By Adding Extra Constraints

(Extended Abstract)

Sally A. Goldman
Ronald L. Rivest

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

March 12, 1997

Abstract

This paper presents a new way to compute the probability distribution with maximum entropy satisfying a set of constraints. Unlike previous approaches, our method is integrated with the planning of data collection and tabulation. We show how *adding constraints* and performing the associated additional tabulations can substantially speed up computation by replacing the usual iterative techniques with a straight-forward computation. These extra constraints are shown to correspond to the intermediate tables used in Cheeseman's method. We also show that the class of constraint graphs that our method handles is a proper generalization of Pearl's singly-connected networks. An open problem is to determine a minimal set of constraints necessary to make a hypergraph acyclic. We conjecture that this problem is NP-complete, and discuss heuristics to approximate the optimal solution.

1 Introduction

Many applications require reasoning with incomplete information. For example, in artificial intelligence one may wish to develop expert systems that can answer

[0]This research was supported in part by NSF Grant DCR–8006938. Sally Goldman received support from an Office of Naval Research Fellowship.

questions when given an incomplete model of the world. Having an incomplete model means that a question can have more than one answer consistent with the model. How can a system choose an answer to such a question? This paper discusses a technique based on the method of maximum entropy. After formally defining this problem, we discuss previously known methods for calculating the maximum entropy distribution. Then we present a new technique which makes maximum entropy computations easier by adding extra constraints. Finally, we compare our technique to previous methods.

2 Formal Problem Definition

In this section we formally define our problem. We begin by defining some notation. Let $V = \{A, B, C, \ldots\}$ be a finite set of binary-valued variables, or attributes. (The generalization to finite-valued variables is straightforward.) Consider the event space Ω_V defined to be the set of all mappings from V to $\{0, 1\}$. (We call such mappings *assignments* since they assign a value to each variable in V.) It is easy to see that $|\Omega_V| = 2^{|V|}$. If $E \subseteq V$, we have Ω_V is isomorphic to $\Omega_E \times \Omega_{V-E}$; we identify assignments in Ω_E with subsets of Ω_V in the natural manner.

We are interested in probability distributions defined on Ω_V. We use the following convention throughout this paper. If $E \subseteq V$, we write $P(E)$ to denote the probability of an element of $\Omega_E \subseteq \Omega_V$. In other words, we specify only the variables involved in the assignments and not their values. For example

$$P(V) = P(A)P(B)P(C)\cdots \tag{1}$$

represents $2^{|V|}$ equations, stating that the variables are independent. (We do not assume equation (1) in this paper.) By convention, all assignments in an equation must be consistent. We also write $P(A)$ instead of $P(\{A\})$, $P(AB)$ instead of $P(\{AB\})$, and so on.

We use a similar convention for summations: $\sum_E$ stands for a summation over all assignments in Ω_E, when $E \subseteq V$. Using our conventions, we see that $Y \subseteq E \subseteq V$ implies that

$$P(Y) = \sum_{E-Y} P(E) \tag{2}$$

We are interested in probability distributions on Ω_V satisfying a given set of constraints. Let $E_1, \ldots E_m$ be distinct subsets of V. Let us suppose that for each i we are given the $2^{|E_i|}$ constraint values $\{P(E_i)\}$, and that these values are consistent (i.e. there exists at least one probability distribution on Ω_V which satisfies the given constraints). A common way of ensuring that the constraints are consistent is to derive the constraints by computing the observed marginal probabilities from a common set of data. (Using "experts" to provide subjective probability estimates is a well-known way of deriving a set of *inconsistent* constraints.) Note that equation (2) states that constraints on the $P(Y)$'s are implied by the constraints on the $P(E_i)$'s when $Y \subseteq E_i$. In general, there may be many probability distributions P satisfying the given constraints; in this case we are interested in that unique distribution P^* which maximizes the entropy

$$H(P) = -\sum_V P(V)\log(P(V)) \tag{3}$$

Motivation for this choice can be found in [Ja79, Ja82, JS83, Le59, SJ80, TTL84].

The maximum entropy distribution P^* is known to have a simple representation. For each ω in Ω_{E_i}, there are $2^{|E_i|}$ non-negative real variables $\alpha_{E_i}(\omega)$ (i.e., one variable per constraint), that determine P^* as follows. Let us write $\alpha_i(\omega)$ instead of $\alpha_{E_i}(\omega)$ for brevity, and omit the argument ω when it can be deduced from context. Now we may write simply

$$P^*(V) = \alpha_1\alpha_2\ldots\alpha_m. \tag{4}$$

Each element of Ω_V is assigned a probability which is the product of the appropriate α's where each α determines its argument from the assignment to V. This is known as a *log-linear* representation; it is an interesting fact that the maximum entropy distribution is the *unique* log-linear distribution of form (4) which satisfies the given constraints. (That is, to find the maximum entropy distribution satisfying the constraints, it suffices to find the log-linear distribution satisfying the constraints.)

Of course, in order for equation (4) to hold, the α's must be correctly computed. The problem of computing the maximum entropy distribution becomes the problem of computing the α_i's from the $P(E_i)$'s.

3 Previous ME Methods

Most existing methods for calculating the maximum entropy distribution are iterative. They typically begin with a representation of the uniform distribution and converge towards a representation of the maximum entropy distribution. Each step adjusts the representation so that a given constraint is satisfied. To enforce a given constraint $P(E_i)$, all of the elementary probabilities $P(V)$ relevant to that constraint are multiplied by a common factor. Because constraints are dependent, adjusting the representation to satisfy one constraint may cause a previously satisfied constraint to no longer hold. Thus one must iterate repeatedly through the constraints until the desired accuracy is reached. (We note that the implicit constraint — that the probabilities sum to one — must usually be explicitly considered here.) Examples of this type of algorithm are given in [Br59, Ch83, Cs75, Fi70, IK68, KK69]. Representing the probability distribution explicitly as a table of $2^{|V|}$ values is usually impractical. For this problem, it is most convenient to store only $\alpha_1, \alpha_2, \ldots \alpha_m$; this is a representation as compact as the input data, which represents the current probability distribution implicitly via equation (4). To represent the uniform distribution, every α is set to 1, except for the α corresponding to the requirement that entries of the probability distribution must sum to 1 — which is set to $2^{-|V|}$. To determine if a constraint is satisfied, one must sum the appropriate elements of the probability distribution Any particular element can be computed using equation (4). If the constraint is not satisfied, the relevant α is multiplied by the ratio of the desired sum to the computed sum. Thus in originally calculating the α's and later in evaluating queries it is necessary to evaluate a sum of terms, where each term is a product of α's. This sum is difficult to compute since it may involve an exponential number of terms.

Cheeseman [Ch83] proposes a clever technique for rewriting such sums in order to evaluate them more efficiently. For example

$$\alpha \sum_{A \ldots F} \alpha_{AB}\ \alpha_{ACD}\ \alpha_{DE}\ \alpha_{AEF}$$

is rewritten as follows. First, $\sum_{A \ldots F}$ is broken into six sums, each over one variable. Arbitrarily choosing the variable ordering $CDFEAB$ we obtain

$$\alpha \sum_B \sum_A \sum_E \sum_F \sum_D \sum_C \alpha_{AB}\alpha_{ACD}\alpha_{DE}\alpha_{AEF}.$$

Now each α is moved left as far as possible (it stops when reaching a sum over a variable on which it depends). The above sum then becomes

$$\alpha \sum_B \sum_A \alpha_{AB} \sum_E \sum_F \alpha_{AEF} \sum_D \alpha_{DE} \sum_C \alpha_{ACD}.$$

The sums are evaluated from right to left. The result of each sum is an intermediate *table* containing the value of the sum evaluated so far as a function of variables further to the left which have been referenced. For example after evaluating $\sum_C$ a table is kept containing α_{ACD} for Ω_{AD}. (As we shall see, in some cases the intermediate table is most efficiently represented as two or more smaller tables.) The variable ordering must be chosen carefully in order to take full advantage of this technique. A poor choice of variable ordering can yield a sum which is not much better than explicitly considering all $2^{|V|}$ terms; a good choice can dramatically reduce the work required. The choice of variable ordering which minimizes the cost of evaluating a sum can be viewed as a vertex ordering problem in a graph. This problem is very similar to the minimum fill-in problem encountered when performing Gaussian elimination on sparse symmetric matrices [RT78, RTL76]. Since the minimum fill-in problem has been proven to be NP-Complete [Ya81], we conjecture that this problem is as well.

Some alternative approaches to the iterative scheme have been proposed. One of the more interesting proposals is due to Geman [Ge86, Li86]. This method uses stochastic relaxation to simultaneously adjust the α's to meet all of the constraints, instead of satisfying one constraint at a time.

Some authors restrict their attention to probability distributions that can be easily worked on without resorting to the iterative methods needed to compute the maximum entropy distribution. These approaches usually restrict the kinds of constraints that might be supplied, and assume conditional independence explicitly as needed to force a unique result. This approach is taken by Chow and Liu [CL68] and Pearl [Pe85]. These methods construct a dependency tree where nodes represent variables and links represent direct dependencies; all direct influences on a node come from its parent. Here the set of all conditional probabilities of the form, $P(\text{child}|\text{parent})$, together with the probability distribution of the variable at the root, suffice to define a unique probability distribution. Pearl [Pe85] generalizes the tree condition to a network which has at most one undirected path between any pair of nodes (a *singly-connected network*).

One can view the contribution of the current paper as providing a synthesis of these two approaches, by showing how the difficulties of computing a maximum entropy distribution can be substantially alleviated by *enlarging* the set of constraints to be considered *before* the data is gathered and tabulated. With the enlarged set of constraints, the computation of the maximum entropy distribution has a simple form which generalizes the equation suggested by Pearl.

4 Using Acyclic Hypergraphs

Our approach is based on the work of Malvestuto [Ma85], who derived sufficient conditions for writing marginals of the maximum entropy formula as a product of easily calculated probabilities. We begin by describing how to model a set of attributes and associated constraints as a hypergraph. (A similar model was given by [EK83].) It is interesting to note that the work on the desirability of acyclic schemas first appeared in the database literature [BFMMUY81, BFMY83, TY82]. The attributes of the database replace the attributes in our problem, and the relations replace the constraint sets. Given that the database scheme is acyclic many problems are simplified

A hypergraph is like an ordinary undirected graph, except that each edge may be an arbitrary subset of the vertices, instead of just a subset of size two. We define the hypergraph $G = (\mathcal{V}, \mathcal{E})$ to contain a vertex for each variable, and a hyperedge for each constraint. For example the hyperedge $\{ABC\}$ corresponds to the constraint set $\{A, B, C\}$. We say that hyperedge X *subsumes* hyperedge Y if $Y \subseteq X$. It is important to observe that the constraints on a sub-hypergraph induced by restricting attention to a subset of the vertices can be inferred from the original hypergraph constraints.

A hypergraph is *acyclic* if repeatedly applying the following reduction steps gives the empty hypergraph (containing no edges and no vertices):

1. Delete any vertices which belong to only one hyperedge.

2. Delete any hyperedges which are subsumed by another hyperedge.

Graham's algorithm is the procedure of applying reduction steps 1 and 2 until either the empty set is reached, or neither can be applied [Gr79].

Before proceeding, we define some necessary notation regarding the above reduction procedure. Let $\mathcal{E}^{(0)} = \{E_1^{(0)}, \ldots, E_m^{(0)}\}$, where $E_i^{(0)}$ is the i^{th} hyperedge of G. Let $E_i^{(k)} = Z_i^{(k)} \cup Y_i^{(k)}$, where $Z_i^{(k)}$ is the set of variables which appear only in $E_i^{(k)}$ and $Y_i^{(k)}$ is the set of variables which appear in at least one hyperedge other than $E_i^{(k)}$. Finally let $\mathcal{E}^{(i+1)}$ be the result of applying reduction step (1) and then (2) to $\mathcal{E}^{(i)}$. If G is acyclic then there exists an l such that $\mathcal{E}^{(l+1)} = \emptyset$.

When the hypergraph is acyclic, the maximum entropy distribution, $P^*(V)$, is given by: [Ma85]

$$P^*(V) = \left(\prod_{k=0}^{l-1} \frac{\prod_i P(E_i^{(k)})}{\prod_i P(Y_i^{(k)})} \right) \left(\prod_i P(E_i^{(l)}) \right) \tag{5}$$

Note that no α's are needed; the formula depends only on probabilities in the original input data (constraints). This formula is an immediate extension of the following theorem.

Theorem 1 [Ma85] *Given a decomposition, $\mathcal{E} = \{E_1, \ldots, E_m\}$, the maximum entropy distribution is given by the following.*

$$P^*(V) = \frac{P(E_1) \cdots P(E_m)}{P(Y_1) \cdots P(Y_m)} P^*(Y)$$

where $P^*(Y)$ is the maximum entropy distribution for the constraints $P(Y_1), \ldots,$ (Y_m).

Proof: From the marginal constraints we have the following

$$\begin{aligned} P(E_i) &= \sum_{V-E_i} \alpha_1 \cdots \alpha_m \\ &= \alpha_i \sum_{V-E_i} \prod_{j \neq i} \alpha_j \end{aligned} \tag{6}$$

Similarly we have,

$$\begin{aligned} P(Y_i) &= \sum_{Z_i} \sum_{V-E_i} \alpha_1 \cdots \alpha_m \\ &= \left(\sum_{Z_i} \alpha_i \right) \left(\sum_{V-E_i} \prod_{j \neq i} \alpha_j \right) \end{aligned} \tag{7}$$

Let $\beta_i = \sum_{Z_i} \alpha_i$. Combining equations (6) and (7) from above gives:

$$\alpha_i = \frac{P(E_i)}{P(Y_i)} \beta_i \tag{8}$$

Now writing $P^*(V)$ in its product form we get

$$\begin{aligned} P^*(V) &= \alpha_1 \cdots \alpha_m \\ &= \frac{P(E_1) \cdots P(E_m)}{P(Y_1) \cdots P(Y_m)} \beta_1 \cdots \beta_m \end{aligned} \tag{9}$$

We want to show that $\psi(Y) = \beta_1 \cdots \beta_m$ is $P^*(Y)$, the maximum entropy distribution for the constraints $P(Y_i)$. To do this, it is suffices to prove that the marginal constraints hold.

$$\begin{aligned} P^*(E_i) &= \sum_{V-E_i} \left(\prod_j \frac{P(E_j)}{P(Y_j)} \right) \psi(Y) \\ &= \sum_{V-E_i} \psi(Y) \frac{P(E_i)}{P(Y_i)} \prod_{j \neq i} \frac{P(E_j)}{P(Y_j)} \\ &= \frac{P(E_i)}{P(Y_i)} \sum_{Y-Y_i} \left(\psi(Y) \prod_{j \neq i} \frac{1}{P(Y_j)} \sum_{Z-Z_i} \left(\prod_{j \neq i} P(E_j) \right) \right) \end{aligned} \tag{10}$$

where $Z = Z_1 \cup \cdots \cup Z_m$, so that $V = Y \cup Z$. Now, since the Z_j's are disjoint,

$$\begin{aligned}
\sum_{Z-Z_i} \prod_{j \neq i} P(E_j) &= \prod_{j \neq i} \sum_{Z-Z_i} P(E_j) \\
&= \prod_{j \neq i} \sum_{Z_j} P(E_j) \\
&= \prod_{j \neq i} P(Y_j) \qquad (11)
\end{aligned}$$

Substituting equation (11) into equation (10) gives:

$$P^*(E_i) = \frac{P(E_i)}{P(Y_i)} \sum_{Y-Y_i} \psi(Y)$$

However since $P^*(E_i) = P(E_i)$ we get $P(Y_i) = \sum_{Y-Y_i} \psi(Y)$, so $\psi(Y)$ satisfies the constraints $P(Y_i)$. ■

5 A New ME Method

In this section we present a new procedure for calculating the maximum entropy distribution. The main advantage of our procedure is that it avoids the iteration previously required by providing a direct formula for the desired answer. The major disadvantage is that the method cannot ordinarily be applied if the data is already tabulated and the constraints already derived; the method requires that one "plan ahead" and tabulate additional constraints when processing the data.

Equation (5) allows one to avoid iteration when calculating the maximum entropy distribution for schemas having acyclic hypergraphs. What should one do for *cyclic* hypergraphs? Our method is based on the observation that *a hypergraph can always be made acyclic by adding hyperedges.* (This is trivial to prove, since at worst a hypergraph can be made acyclic by adding the hyperedge containing all vertices.) For example, the hypergraph:

$$(\mathcal{V}, \mathcal{E}) = (\{ABCDEF\}, \{\{AB\}, \{ACD\}, \{DE\}, \{AEF\}\})$$

becomes acyclic when the hyperedge $\{ADE\}$ is added. Thus by adding additional constraints (edges) the maximum entropy calculation can be simplified so that no iteration is required. Here is a summary of how our method works:

1. We begin with a set of variables (attributes) and a set of constraint groups deemed to be of interest. (Cheeseman [Ch84] discusses a learning program which uses the raw data to find a set of significant constraints. Edwards and Kreiner [EK83] also discuss how to choose a good set of constraints.) Here a "constraint group" is a set of variables; the intent is that during data-gathering there will be one table created for each constraint group, and the observed events will be tabulated once in each table according to the values of the attributes in the constraint group. For example, if $\{A, B, C\}$ is a constraint group of three binary valued attributes, then there will be a table of size 8 used to categorize the data with respect to these three attributes. This will give rise to 8 constraints on the maximum-entropy distribution desired, one for each of the eight observed probabilities $P(ABC)$.

2. Construct the corresponding hypergraph $G = (\mathcal{V}, \mathcal{E})$, where there is one vertex for each variable and one hyperedge corresponding to each constraint group.

3. Perform Graham's algorithm on G, and let G' denote the resulting hypergraph. If G' is the empty hypergraph, then G is acyclic, and the following step is skipped.

4. Find a minimal set $\mathcal{X}$ of additional hyperedges (constraint groups) which can be added to G' to make it acyclic. Note that any original edges subsumed by edges in $\mathcal{X}$ are eliminated.

5. Collect data for the expanded set $\mathcal{E} \cup \mathcal{X}$ of constraints [1].

6. Apply equation (5) to calculate individual elements of the maximum entropy distribution. If sums of elements are desired, use Cheeseman's summation technique, choosing a good variable ordering. Note that instead of having a summation over a product of α's, here the summation is over a product of probabilities which are equivalent in form to the α's.

6 Possible Problems With Our Method

In this section we consider possible inefficiencies of our method. First, it may be necessary to add "large" hyperedges containing many vertices in order to make

[1]Our method is unusual in that it extends the set of tables (constraints) used to tabulate the data. To fill in the entries of a new table, the raw data must still be available in step (5). Thus steps 1-4 may be considered to be "planning" steps.

the hypergraph acyclic. For example, to make the complete undirected graph (containing all hyperedges of size two) acyclic, one must add the "maximum" hyperedge containing all vertices. Since the size of the table corresponding to a hyperedge is an exponential function of the size of the hyperedge, adding large hyperedges creates a problem. Furthermore, the table corresponding to the maximum hyperedge is itself the probability distribution that we are estimating, so the above situation is clearly undesirable. This kind of behavior depends on the structure of the hypergraph; hypergraphs which are "highly connected" will tend to require the addition of large hyperedges. However, when the graph is highly connected other computational techniques seem to "blow up" as well.

Second, because of our method's unique approach, we have a unique concern. Recall that since the data is tabulated *after* adding the additional constraints; steps 1-4 of our algorithm must be performed while the source of the constraints (i.e., the raw data) is still available. If the added hyperedges are too large, there may not be enough data to calculate meaningful statistics. Tabulating 100,000 data points in a table of size $\approx 1,000$ will give reasonable estimates, while tabulating them in a table of size $\approx 1,000,000$ will not.

7 Comparison with Cheeseman's Method

We now compare the tables (constraints) added by our technique, with the intermediate tables used in Cheeseman's method. We demonstrate, by means of an example, that these are the same, except that Cheeseman's tables are half the size, since they are are already summed over the variable being eliminated.

When evaluating a sum one can visualize an imaginary "scan line" moving from left to right across the hypergraph, where all variables to the left of the scan line have already been summed over.

- According to the position of a scan line the vertices of the graph may be divided into three parts ("eliminated", "boundary", and "unseen") as follows:

    1. $V_E = \{v \in V \mid v \text{ is left of the scan line } \}$
    2. $V_B = \{v' \in V - V_E \mid \exists v \in V_E : (v, v') \in E \}$
    3. $V_U = V - (V_E \cup V_B)$

- Let $G_1, G_2, \ldots, G_l$ be the connected components of the subgraph induced by $V_E \cup V_B$.

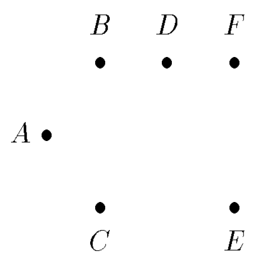
- Let $\Gamma(G_i)$ denote the set of vertices of V_B in G_i.

- Let $\Gamma'(G_i)$ be that subset of $\Gamma(G_i)$ consisting of the vertices adjacent to some vertex in V_U or V_B.

Theorem 2 *Let $X_1, X_2, \ldots, X_l$ be a collection of subsets of V_B. Then $\{X_j\}$ is sufficient and necessary for Cheeseman's algorithm if*

$$(\forall i)(\exists j) \mid \Gamma'(G_i) \subseteq X_j. \tag{12}$$

Now we will look at the following example:



To explain this example we introduce new notation, where the variable sets for the intermediate tables are put in parenthesis above the summation sign. Consider the vertex ordering $DEFCAB$; given such a vertex ordering, theorem 2 specifies the temporary tables needed.

$$P^*(\emptyset) = \sum_{AB} \alpha_{AB} \overset{(AB)}{\sum_{C}} \alpha_{AC}\alpha_{BC} \overset{(BC)}{\sum_{F}} \alpha_{CF} \overset{(BF,CF)}{\sum_{E}} \alpha_{CE}\alpha_{EF} \overset{(BF)}{\sum_{D}} \alpha_{BD}\alpha_{DF}$$

(This summation is only being used for explanatory purposes. Since the elements of the probability distribution sum to 1, we know that $P^*(\emptyset) = 1$.) When evaluating $\sum_D$ it will be necessary to keep a table with the value of $\alpha_{BD}\alpha_{DF}$ for Ω_{BF}. When evaluating $\sum_E$ as well as keeping the table for B and F, another table with all combinations of C and F must be created. Below are the temporary tables used by Cheeseman's method.

$$BF, CF, BC, AB$$

Now we look at the hyperedges which must be added to make a hypergraph acyclic.

- Let $v_1, \ldots, v_n$ be an ordering of the vertices.
- Let $G_i = \begin{cases} G & \text{if } i = 0 \\ G_{i-1} + \phi(v_i) - \{e \mid v_i \in e\} & \text{otherwise} \end{cases}$

- Let $\phi(v_i) = \{v \mid v \text{ is adjacent to } v_i \text{ in } G_{i-1}\}$.
- Let $\Phi(v_i) = \begin{cases} \emptyset & \text{if } \phi(v_i) = \emptyset \\ \phi(v_i) \cup v_i & \text{otherwise} \end{cases}$

Theorem 3 *The set of all non-empty $\Phi(v_i)$ is necessary and sufficient to make a hypergraph, G, acyclic.*

Now we return to our example. Theorem 3 defines a set of additional hyperedges that make our hypergraph acyclic. First, the hyperedge BFD eliminates vertex D. (Hyperedge BFD subsumes hyperedges BD and DF and thus eliminates them. Now D is only in hyperedge BDF and so is eliminated, leaving hyperedge BF.) Second, hyperedge CFE eliminates vertex E. Now vertex F is only in hyperedges BF and CF so BCF eliminates it. Finally, hyperedge ABC eliminates the remaining vertices. Thus the following additional hyperedges will make our example graph acyclic (in parenthesis is the variable eliminated by adding the edge):

$$BF(D), CF(E), BC(F), AB(C)$$

Ignoring the variables in parentheses, these are identical to Cheeseman's tables. However there are important differences between these methods.

First, in terms of time complexity, Cheeseman's method specifies an iterative approximation of the αs, whereas our method requires no such iteration. So, if Cheeseman's method requires 10 iterations on the average, our method should yield an average speed-up of a factor of 10.

Second, in terms of space complexity, both methods use approximately the same amount of space. However, our method adds what might be called "permanent" edges, since they correspond to tabulations of the raw data. Note, however, that new edges may subsume and thus eliminate original edges, so the space required by our method may not be quite as great as it first appears. In Cheeseman' method the tables exist only temporarily during the course of the computation, and not all such tables may be needed at the same time.

And finally, in terms of the "precomputation" needed, both methods need to compute a vertex ordering to use. We observe that a good summation ordering is a good ordering for eliminating vertices. So the problem of choosing the hyperedges to make a graph acyclic seems comparable to the problem of choosing an optimal summation ordering.

8 Comparison to Pearl's Work

In this section we will compare our work to that of Pearl, and show that the formula we use is a proper generalization of Pearl's. Pearl's technique depends on the Bayesian network being singly-connected. A Bayesian network is a directed acyclic graph; such a network is said to be *singly-connected* if it has no *undirected* cycles (i.e., no cycles if we ignore the directions of the edges).

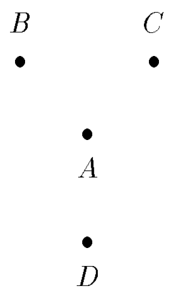
We begin by proving that a singly-connected network is a special case of an acyclic hypergraph. Then we show that both Pearl's and Malvestuto's formulas yield the same estimated probability distribution for a singly-connected network. Given a network N, we define a corresponding hypergraph G as follows. The vertices of G are the nodes of N. For each node x in N we create a hyperedge in G, consisting of the corresponding vertex and the vertices corresponding to all immediate predecessor of x in N.

Theorem 4 *If a Bayesian network N is singly-connected, then the corresponding hypergraph G is acyclic.*

Proof: Since N is singly-connected, it contains no undirected cycles, by the definition of singly-connected. Since an acyclic undirected graph is a tree or forest, N must contain some node s with degree at most one. The corresponding vertex in G is contained in exactly one hyperedge and so can be eliminated by reduction step 1. Finally, we must show that the reduced network N' so obtained corresponds to the reduced hypergraph G' that remains after eliminating the vertex corresponding to s. When s is a source in N (or a sink which is the second to last node) this correspondence is obtained immediately. In the remaining cases, the correspondence holds only after applying reduction step 2 to eliminate the hyperedge remaining after s is eliminated. (This hyperedge contains only the parent of s.) The network N' is singly-connected, and so by induction every node in G can be eliminated. Thus G is acyclic. ■

Theorem 5 *A Bayesian network is not necessarily singly-connected if the corresponding hypergraph is acyclic.*

Proof: We prove this by means of an example. The following Bayesian network is not singly-connected since there is an undirected cycle.



The hypergraph corresponding to the above network is acyclic as shown by the reduction below:

$$\begin{array}{c} \{B, C, ABC, BCD\} \\ \Downarrow \\ \{BC\} \\ \Downarrow \\ \emptyset \end{array}$$

■

Thus Pearl's condition of the network being singly-connected is a special case of having an acyclic hypergraph. So when given a singly-connected network, our technique will apply without adding any constraints.

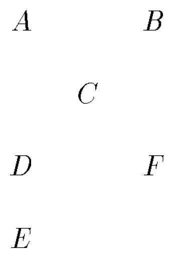
Now we show (by example) that for singly-connected networks, Pearl's and Malvestuto's formulas for the estimated probability distribution are equivalent. We use the following notation for stating Pearl's formula.

- $\{f_x\}$ are the fathers (immediate ancestors) of node x in the network.
- $\mathcal{R}$ is the set of roots (sources).

Pearl's equation is

$$P^*(V) = \left(\prod_{x \in \mathcal{R}} P(x)\right)\left(\prod_{x \notin \mathcal{R}} P(x|\{f_x\})\right) \tag{13}$$

We use the following example to compare Pearl's and Malvestuto's formulas.



Pearl's formula gives:

$$\begin{aligned} P^*(A \ldots F) &= P(A)P(B)P(C|AB)P(D|C)P(F|C)P(E|D) \\ &= P(A)P(B)\frac{P(ABC)}{P(AB)}\frac{P(CD)}{P(C)}\frac{P(CF)}{P(C)}\frac{P(DE)}{P(D)} \end{aligned}$$

By definition A and B must be independent since otherwise there would be a link connecting them. Thus $P(AB) = P(A)P(B)$ and the above becomes

$$P^*(A \ldots F) = P(ABC)\frac{P(CD)}{P(C)}\frac{P(CF)}{P(C)}\frac{P(DE)}{P(D)} \tag{14}$$

To apply Malvestuto's formula, it is necessary to perform Graham's algorithm. (Elements of $Y_i^{(0)}$ are underlined.)

$$\begin{array}{c} \{AB\underline{C}, \underline{CD}, \underline{C}F, \underline{D}E\} \\ \Downarrow \\ \{CD\} \\ \Downarrow \\ \emptyset \end{array}$$

Applying equation (5) gives:

$$P^*(A \ldots F) = \frac{P(ABC)}{P(C)}\frac{P(CF)}{P(C)}\frac{P(DE)}{P(D)}P(CD) \tag{15}$$

As shown by equations (14) and (15) the probability distribution given by Pearl's and Malvestuto's formulas are the same. So Pearl's formula is a special case of Malvestuto's formula. Thus our method is a proper generalization of the method given by Pearl. That is, not only does our technique give the same results as Pearl's when the network is singly-connected, but our formula applies (without adding any hyperedges) to cases where Pearl's does not.

9 Conclusions and Open Problems

We have presented an efficient algorithm for calculating the maximum entropy distribution given a set of attributes and constraints. Using a hypergraph to model the attributes and constraints, we show the benefits of making the corresponding hypergraph *acyclic*. We then show how to make a hypergraph acyclic by adding hyperedges (constraints). We have shown that our technique is at least as efficient as Cheeseman's method, and that our technique generalizes Pearl's method for singly-connected networks.

An open problem is how to choose the best set of hyperedges which will make a hypergraph acyclic; we conjecture that this problem is NP-complete. If so, then step (4) of our method cannot be done efficiently. One can use heuristics (such as a minimum-degree heuristic) to approximate the optimal answer, or maybe there is a pseudo-polynomial time algorithm in the size of the contingency tables corresponding to the edges of the optimal hypergraph.

We intend to try our technique on some realistic examples. Our goal is to determine if the size of the hyperedges will remain within reasonable limits for such realistic examples. We expect that in practice our new method will give substantial improvements in running time.

Finally, we will study the effects on accuracy of keeping tables which may be larger than the original tables.

References

[BFMMUY81] Berri, C., R. Fagin, D. Maier, A. Mendelzon, J.D. Ullman and M. Yannakakis, "Properties of Acyclic Database Schemas," in *Proc. 13th Annual ACM STOC* (1981), 355–362.

[BFMY83] Beeri, C., R. Fagin, D. Maier and M. Yannakakis, "On the Desirability of Acyclic Database Schemas,"*J. ACM*, **30**,3 (1983), 355-362.

[Br59] Brown, D.T., "A Note on Approximations to Discrete Probability Distributions," *Information and Control*, **2** (1959), 386–392.

[Ch83] Cheeseman, P.C., "A Method For Computing Generalized Bayesian Probability Values For Expert Systems," in *Proc. Eighth International Conference on Artificial Intelligence* (August 1983), 198–202.

[Ch84] Cheeseman, P.C., "Learning of Expert Systems From Data," in *Proc. IEEE Workshop on Principles of Knowledge Based Systems* (1984), 115–122.

[CL68] Chow, C.K. and C.N. Liu, "Approximating Discrete Probability Distributions With Dependence Trees,"*IEEE Trans. on Info. Theory*, **IT-14**,3 (May 1968), 462–467.

[Cs75] Csiszár, I., "I-Divergence geometry of probability distributions and minimization problems," *Annals of Probability*, **3**,1 (1975), 146–158.

[EK83] Edwards, D., and S. Kreiner, "Analysis of contingency tables by graphical models," *Biometrika* **70**,3 (1983), 553–565.

[Fi70] Fienberg, S.E., "An Iterative Procedure For Estimation In Contingency Tables," *The Annals of Mathematical Statistics*, **41**,3 (1970), 907–917.

[Ge86] Geman, S., "Stochastic Relaxation Methods For Image Restoration and Expert Systems," In Cooper, D.B., R.L. Launer, and E. McClure, editors, *Automated Image Analysis: Theory and Experiments*, New York: Academic Press, (to appear).

[Gr79] Graham, M.H., "On the Universal Relation," *University of Toronto Technical Report* (1979).

[IK68] Ireland, C.T., and S. Kullback, "Contingency tables with given marginals," *Biometrika* **55**,1 (1968), 179–188.

[Ja79] Jaynes, E.T., "Where Do We Stand On Maximum Entropy," In Levine and Tribune, editors, *The Maximum Entropy Formalism*, M.I.T. Press, (1979).

[Ja82] Jaynes, E.T., "On the Rationale of Maximum-Entropy Methods," *Proceedings of the IEEE*, **70**,9 (September 1982), 939–952.

[JS83] Johnson, R.W., and J.E. Shore, "Comments and corrections to 'Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy'," *IEEE Trans. Inform. Theory* **IT-29**, 6 (Nov. 1983), 942–943.

[KK69] Ku, H.H. and S. Kullback, "Approximating Discrete Probability Distributions," *IEEE Trans. on Info. Theory*, **IT-15**,4 (July 1969), 444–447.

[Le59] Lewis, P.M., "Approximating Probability Distributions to Reduce Storage Requirements," *Information and Control*, **2** (1959), 214–225.

[Li86] Lippman, A.F., "A Maximum Entropy Method for Expert System Construction," PhD thesis, Brown University, Division of Applied Mathematics, (May 1986).

[Ma85] Malvestuto, F.M., "Approximating Discrete Probability Distributions: Easy and Difficult Cases," Unpublished Manuscript, (December 1985).

[Pe85] Pearl, J., "Fusion, Propagation and Structuring in Bayesian Networks," *University Of California, Los Angeles Dept. of Computer Science Technical Report CSD-850022 R-42*, (April 1985).

[RT78] Rose, D.J. and R.E. Tarjan, "Algorithmic Aspects of Vertex Elimination in Directed Graphs," *SIAM Journal Applied Math*, **24** (1978), 176–197.

[RTL76] Rose, D.J., R.E. Tarjan, and G.S. Lueker, "Algorithmic Aspects of Vertex Elimination on Graphs," *SIAM Journal Comput.*, **5**,2 (June 1976), 266–283.

[SJ80] Shore, J.E., and R.W. Johnson, "Axiomatic Derivation of the Principle of Maximum Entropy and the Principle of Minimum Cross-Entropy," *IEEE Trans. Inform. Theory*, **IT-26**,1 (Jan. 1980), 26–37.

[TTL84] Tikochinsky, Y., N.Z. Tishby, and R.D. Levine, "Consistent inference of probabilities for reproducible experiments," *Physical Rev. Letters* **52**, 16 (16 April 1984), 1357–1360.

[TY82] Tarjan, R.E. and M. Yannakakis, "Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs," *SIAM J. Comp.*, **13**,3 (August 1984), 566–579.

[Ya81] Yannakakis, M., "Computing the Minimum Fill-in is NP-Complete," *SIAM Journal Alg. Disc. Meth.*, **2**,1 (March 1981), 77–79.