TECHNICAL REPORT
R-116

Identifying Independence in Bayesian Networks*

Dan Geiger, Thomas Verma, and Judea Pearl
Cognitive Systems Laboratory, Computer Science Department, University of California Los Angeles, California 90024

An important feature of Bayesian networks is that they facilitate explicit encoding of information about independencies in the domain, information that is indispensable for efficient inferencing. This article characterizes all independence assertions that logically follow from the topology of a network and develops a linear time algorithm that identifies these assertions. The algorithm's correctness is based on the soundness of a graphical criterion, called *d*-separation, and its optimality stems from the completeness of *d*-separation. An enhanced version of *d*-separation, called *D*-separation, is defined, extending the algorithm to networks that encode functional dependencies. Finally, the algorithm is shown to work for a broad class of nonprobabilistic independencies.

1. INTRODUCTION

The practical significance of conditional independence is reflected in three processes that are supported by expert systems: encoding the experience of an expert (elicitation), drawing conclusions (inference), and communicating the system's recommendations to the user (explanation). In eliciting probabilistic models from human experts, qualitative dependencies among variables can often be asserted with confidence, whereas numerical assessments are subject to a great deal of hesitancy. For example, an expert may willingly state that cancer is related to both smoking habits and asbestos exposure; however, he/she would not provide a numeric quantification of these relationships unless he/she has rich experience with cancer patients or is aware of a reliable statistical survey that estimates the strength of these relationships. Developing a direct representation scheme for judgments about dependencies, which is the major contribution of this article, facilitates a qualitative organization of knowledge in a manner that is amenable to a human expert and guards the model builder from assigning numerical values that lead to conceptually unrealistic dependencies.

Knowledge about independence saves space when storing probability distribu-

*This work was partially supported by the National Science Foundation Grant #IRI-8821444.

tion functions and saves time when computing and updating the probability of an event; if we ignore independencies, then representing a discrete distribution function would require exponential size tables and calculating P (x is *true*) would require a lengthy summation over the other variables in the table. Recognizing the independencies among the variables enables us to encode the table with fewer parameters and to considerably reduce the computations. Furthermore, if we choose to represent and process random variables by Bayesian networks, then the topology of such networks as well as the set of transformations that we are permitted to apply to them are determined by the rules that govern conditional independence.

Finally, a qualitative characterization of conditional independence in terms of logical axioms that do not refer to numerical quantities highlights plausible lines of reasoning that would otherwise be hidden in numerical calculations. Such axioms could serve as building blocks of systems that provide qualitative explanations as to why certian facts were or were not taken into account in a given computation. For example, the axiom (4c) below can be phrased to read: "If two items together are judged to be unnecessary for a computation, then learning one of them leaves the other still unnecessary." By contrast, a numeric representation of this argument would involve complicated equations that hide the intuition behind it. Thus, a logical characterization is preferable and is pursued in this paper.

This work develops a scheme for representing and manipulating dependencies in the framework of *Bayesian networks*. This graph-based system records a state of probabilitic knowledge P, provides means for updating the knowledge as new information accumulates and facilitates query answering mechanisms for knowledge retrieval [15,18]. Formally, a Bayesian network (also known as probabilistic influence diagram) is a parameterized directed acyclic graph (dag) D constructed from a probability distribution P called the *underlying* distribution. Each node α in D corresponds to a variable α in P, a set of nodes X correspond to a set of variables X, and the symbols a and x denote values drawn from the domain of α and X, respectively. Each node α in the network is regarded as a storage cell for the distribution $P(\alpha | \pi(a))$, where $\pi(\alpha)$ is a set of variables that correspond to the parent nodes $\pi(\alpha)$ of α. The underlying distribution from which a Bayesian network is drawn decomposes into

$$P(a_1, \ldots, a_n) = \prod_{i=1}^{n} P(a_i | \pi(a_i)); \qquad (1)$$

when α_i has no parents, then $P(a_i | \pi(a_i)) = P(a_i)$. A standard query for a Bayesian network is to find the current belief distribution of a hypothesis α, given an evidence set $Y = y$, i.e., to compute $P(a|y)$ for each value of α and for a given combination of values of Y. The answer to such queries can, in principle, be computed directly from Eq. (1) because this equation defines a full probability distribution. This, however, might be very inefficient both in time and space, unless we exploit the independence relationships embodied in the product of Eq. (1), as portrayed by the network. The following two problems must be examined to take full advantage of such independencies: Given a variable γ, a Bayesian network D, and the task of computing $P(a|y)$, determine, without resorting to numeric calculations, (1) whether the answer to the query in sensitive to the value of a variable γ, and (2) whether the answer to the query is sensitive to the *parameters* $p_\gamma = P(c|\pi(c))$ stored at node γ.

The answer to these questions is given in terms of conditional independence: The value of γ would not affect the query $P(a|y)$ if $P(a|y) = P(a|y,c)$ for all instances a, y, and c, or, equivalently, if α and γ are *conditionally independent* given Y, denoted by $I(\alpha,Y,\gamma)_P$. Similarly, the parameters p_γ stored at node γ would not affect this query if α is conditionally independent of p_γ given Y. The main claim made here is that many of these independencies can be detected directly from the topology of the network, merely by examining the trails along which α, Y, and γ are connected.

The rest of the article is organized as follows. In Section 2, we define dependency models and graphoids, concepts that abstract the most essential properties of conditional independence from probability theory and that provide a qualitative formalism to reason about dependencies. In Section 3, we provide a graphical criterion, *d-separation*, that identifies the maximum number of independencies from the network without resorting to numerical calculations. Sections 4 and 5 are the main contribution of this paper. Section 4 extends the analysis of Section 3 to networks in which deterministic variables are present, namely, variables α's whose value is deterministic function of its parents $\pi(\alpha)$ [22]. A new graphical criterion is provided, called D-separation, which is shown to be maximal for these networks. Finally, Section 5 employs the declarative definition of D-separation as the basis for an efficient linear-time algorithm that solves the two problems raised in the Introduction.

2. DEPENDENCY MODELS

The notion of dependency models presented below was originated by Pearl and Paz [20]. The definitions are taken from [21].

Definition. A *dependency model* M over a finite set of elements U is any subset of triplets (X,Z,Y) where X, Y, and Z are three disjoint subsets of U. The triplets in M represent independencies, that is, $(X,Z,Y) \in M$ asserts that X and Y interact only via Z, or "X is independent of Y given Z." This statement is called an *independence statement* and is also written as $I(X,Z,Y)$ with an optional subscript to clarify the type of dependency when necessary. An independence statement is often called an *independency*; the negation of an independency is called a *dependency*.

We will use both set notations and logic notations when speaking about dependency models. For example, we say that a triplet T belongs to a dependency model M or that an independence statement T holds for M. Similarly, we say that M contains T or that M satisfies the statement T.

Definition. A *Probabilistic Dependency Model* M_P is defined in terms of a discrete probability distribution over a finite set of variables U. If X, Y, and Z are three disjoint subsets of U, and x, y, and z, any instances of the variables in these subsets, then by definition $I(X,Z,Y)_P$ iff

$$P(X=x|Z=z,Y=y)=P(X=x|Z=z) \text{ whenever } P(Y=y,Z=z)>0. \tag{2}$$

This definition conveys the idea that, once Z is fixed, knowing Y can no longer influence the probability of X [2].

The class of all probabilistic dependency models is denoted by $\mathcal{M}_{\mathcal{P}}$. Similarly, in all the definitions that follow when M_X is a specialized type of a dependency model (e.g., probabilistic), then $\mathcal{M}_{\mathcal{X}}$ denotes the class of all such models.

Definition. A *Relational Dependency Model* M_R is defined in terms of a data base R over a set of attributes U, i.e., a set of tuples of values for the attributes. The notation $\langle a_1a_2 \cdots a_n\rangle$ is conventionally used to denote that the tuple is in the relation R. If X, Y, and Z are three disjoint subsets of U, and x_1, x_2, y_1, y_2, z, any instances of the corresponding attributes in X, Y, and Z, then by definition $I(X,Z,Y)_R$ iff

$$\langle x_1y_1z\rangle \text{ and } \langle x_2y_2z\rangle \Rightarrow \langle x_1y_2z\rangle. \tag{3}$$

In other words, the existence of the tuples $\langle x_1y_1z\rangle$ and $\langle x_2y_2z\rangle$ guarantees the existence of $\langle x_1y_2z\rangle$. When the implication above holds, we say that $I(X,Z,Y)_R$ holds in R. The same definition is used in relational data base theory, and it conveys the idea that, once Z is fixed, knowing Y cannot further restrict the range of values permitted for X [5]. A similar definition in another framework was called qualitative independence [24]. For a review on data base dependencies, consult Fagin and Vardi [6] or Vardi [27].

Definition [20]. A *Correlational Dependency Model* M_C is defined in terms of a finite collection of random variables U with finite means and variances and with nonzero variances. If X, Y, and Z are three disjoint subsets of U, then by definiton $I(X,Z,Y)_C$ iff $\rho_{\alpha\beta.Z}=0$ for every $\alpha \in X$ and $\beta \in Y$ where $\rho_{\alpha\beta.Z}$ is the partial correlation of α and β [1]. Intuitively, this definition captures the idea that the linear estimation error of the variables in X using measurements on Z would not be improved if measurements were to be taken also on variables of Y, thus rendering Y irrelevant to the estimation of X.

These three types of independence—probabilistic, relational, and correlational—provide different formalisms for the notion of irrelevance, each capturing different aspects of the word "irrelevant." The similarity between these models is summarized axiomatically by the following definition of graphoids.

Definition. A *graphoid*[1] is any dependency model M that is closed under the following *axioms*:

$$\textit{Symmetry} \qquad I(X,Z,Y) \Rightarrow I(Y,Z,X) \tag{4a}$$

$$\textit{Decomposition} \qquad I(X,Z,Y \cup W) \Rightarrow I(X,Z,Y) \tag{4b}$$

$$\textit{Weak union} \qquad I(X,Z,Y \cup W) \Rightarrow I(X,Z \cup W,Y) \tag{4c}$$

$$\textit{Contraction} \qquad I(X,Z,Y) \text{ and } I(X,Z \cup Y,W) \Rightarrow I(X,Z,Y \cup W). \tag{4d}$$

Intuitively, the essence of these axioms lies in Eqs. (4c) and (4d), which assert that when we learn an irrelevant fact, all relevance relationships among other variables in the system should remain unaltered; any information that was relevant remains relevant and that which was irrelevant remains irrelevant. These axioms, common to all formalization of dependencies presented so far, are very similar to those assembled by Dawid [2] for probabilistic conditional independence and those proposed by Smith [25] for *Generalized Conditional Independence*. The difference is only that Dawid and Smith lumped Eqs. (4b) through (4c) into one and added an axiom to handle some cases of overlapping sets X, Y, Z. We shall henceforth call axioms (4a) through (4d) *graphoid* axioms. In view of this generality, these four axioms are selected to represent the general notion of mediated dependence between items of information [19]. A proof that the graphoid axioms hold for conditional independence for all distributions can be found in [26].

Definition. An *intersectional graphoid* is any graphoid M that is also closed under the following property:

$$\textit{Intersection} \qquad I(X,Z \cup Y,W) \text{ and } I(X,Z \cup W,Y) \Rightarrow I(X,Z,Y \cup W). \tag{5}$$

Interestingly, both undirected graphs and directed acyclic graphs (dags) conform to the graphoid axioms (hence, the name) if we associate the statement $I(X,Z,Y)_G$ with the graphical condition "every path between X and Y is blocked by the set of nodes corresponding to Z." In undirected graphs, blocking corresponds to ordinary interception, whereas in dags it is defined by a criteria called d-separation, discussed below. Axiom (5), which reflects the positiveness of a distribution is not satisfied by D-separation (defined in Section 4) because this criterion incorporates functional dependencies and thus inherently represents distributions that assign zero probability to some configurations.

Definition. An *Undirected Graph Dependency Model* M_G is defined in terms

[1]In [19,20], dependency models satisfying axioms (4) are called *semi-graphoids* and those satisfying (4) and (5) are called *graphoids*. We have changed terminology to account for the greater generality of the former.

of an undirected graph G. If X, Y, and Z are three disjoint subsets of nodes in G, then by definition $I(X,Z,Y)_G$ iff every path between nodes in X and Y contains at least one node in Z. In other words, Z is a cutset separating X from Y.

Definition. An *I-map* of a dependency model M is any model M' such that $M' \subseteq M$. For example, if a relation R contains all tuples having nonzero probability in P, then M_P is an I-map of M_R, because every independency embodied by P [defined in Eq. (2)] is also an independency in R [as defined in Eq. (3)]. In this case, we also say that P is an I-map of R.

Definition. A *perfect map* of a dependency model M is any model M' such that $M' = M$.

An important task for representing an arbitrary probabilistic dependency model is the construction of an appropriate graph representation whether directed or undirected. Ideally, to graphically represent all independencies of some distribution P by a graph G, we would like to require that every independency of P would belong to M_G and vice versa and every triplet in M_G would represent an independency that holds in P. In other words, M_G would be a perfect map of M_P. This would provide a clear graphical representation of all variables that are conditionally independent. Unfortunately, this requirement is often too strong because there are many distributions that have no perfect map in graphs. The spectrum of probabilistic dependencies is, in fact, so rich that it cannot be cast into any representation scheme that uses polynomial amount of storage.[2] Thus, the topology of a graph alone cannot always represent all the independencies and dependencies of a given distribution, and we must compromise this requirement and allow some independencies to escape representation. Naturally, we seek a graph that displays only genuine independencies of P and that maximizes the number of such displayed independencies. In other words, we require that M_G be an I-map of M_P and that no edge can be deleted without destroying the I-mapness of G.

The most important properties of intersectional graphoids are that they possess unique edge-minimal I-maps in $\mathcal{M}_{\mathcal{G}}$ and permit the construction of graphical I-maps from local dependencies. We obtain a graph that is an edge-minimal I-map of M by connecting each variable α to the (unique) minimal subset that renders α conditionally independent of all other variables in U [20]. Such local construction of an undirected graph model is not guaranteed for nonintersectional graphoids, but is possible when using the language of dags combined with an appropriate separation criterion, such as d-separation.

[2]This claim is established by showing that the number of distinct probabilistic dependency models over U is at least O $(\exp\{2^{|U|}\})$, thus requiring, on the average, exponential amount of storage to represent a single dependency model [28].

3. SOUNDNESS AND COMPLETENESS OF *d*-SEPARATION

The definition of *d*-separation is best motivated by regarding dags as a representation of causal relationships. Designating a node for every variable and assigning a link between every cause to each of its direct consequences defines a graphical representation of a causal hierarchy. For example, the propositions "It is raining" (α), "the pavement is wet" (β), and "John slipped on the pavement" (γ) are well represented by a three-node chain, from α through β to γ, it indicates that either rain or wet pavement could cause slipping, yet wet pavement is designated as the *direct cause*; rain could cause someone to slip if it wets the pavement, but not if the pavement is covered. Moreover, knowing the condition of the pavement renders "slipping" and "raining" independent, and this is represented graphically by a *d*-separation condition, $I(\alpha,\gamma,\beta)_D$, that shows node α and β separated from each other by node γ. Furthermore, if we assume that "broken pipe" (δ) is another direct cause for wet pavement, as in Figure 1, then an *induced* dependency exists between the two events that may cause the pavement to get wet: "rain" and "broken pipe." Although they appear connected in Figure 1, these propositions are marginally independent and become dependent once we learn that the pavement is wet or that someone broke his leg. An increase in our belief in either cause would decrease our belief in the other as it would "explain away" the observation. The following definition of *d*-separation permits us to graphically identify such induced dependencies from the network (*d* connoted "directional").

Definition. A *trail* in a dag is a sequence of links that form a path in the underlying undirected graph. A node β is called a head-to-head node with respect to a trail t if there are two consecutive links $\alpha \rightarrow \beta$ and $\beta \leftarrow \gamma$ on t. A node β is called a tail-to-tail node with respect to a trail t if there are two consecutive links $\alpha \leftarrow \beta$ and $\beta \rightarrow \gamma$ on t. A node that starts or ends a trail t is a tail-to-tail node if it delivers an arrow but is not a head-to-head node if it receives an arrow.

Definition [19]. A trail t is *active by Z* if (1) every head-to-head node (wrt t) either is or has a descendant in Z and (2) every other node along t is outside Z. Otherwise, the trail is said to be *blocked by Z*.

Definition [19]. If X, Y, and Z are three disjoint subsets of nodes in a dag D,

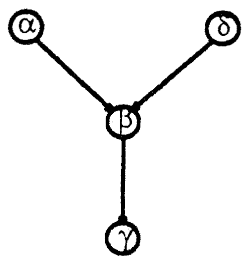

FIG. 1.

then Z is said to d-separate X from Y, denoted $I(X,Z,Y)_D$, iff there exists no active trail by Z between a node in X and a node in Y.

In Figure 2, for example, $X = \{\alpha_4\}$ and $Y = \{\alpha_3\}$ are d-separated by $Z = \{\alpha_2\}$; the trail $\alpha_3 \leftarrow \alpha_2 \rightarrow \alpha_4$ is blocked by $\alpha_2 \in Z$ while the trail $\alpha_3 \rightarrow \alpha_5 \leftarrow \alpha_4$ is blocked because α_5 and all its descendants are outside Z. Thus, $I(\alpha_4,\alpha_2,\alpha_3)_D$ holds in D. However, X and Y are not d-separated by $Z' = \{\alpha_2,\alpha_6\}$ because the trail $\alpha_3 \rightarrow \alpha_5 \leftarrow \alpha_4$ is rendered active: Learning the value of the consequence α_6, renders its causes α_3 and α_4 dependent, like opening a pathway along the converging arrows at α_5. Consequently, $I(\alpha_4,\{\alpha_2,\alpha_6\},\alpha_3)_D$ does not hold in D.

The task of finding a dag that is a minimal I-map of a given distribution P was solved in [21,29]. The algorithm consists of the following steps: First, assign a total ordering d to the variables of P. For each variable α_i of P, identify a minimal set of predecessors $\pi(\alpha_i)$ that renders α_i independent of all its other predecessors (in the ordering of the first step). Then, assign a direct link from every variable in $\pi(\alpha_i)$ to α_i. The resulting dag is an I-map of P and is minimal in the sense that no edge can be deleted without destroying its I-mapness. The input for this construction consists of a list L of n conditional independence statements, one for each variable, all of the form $I(\alpha_i,\pi(\alpha_i),U(\alpha_i) - \pi(\alpha_i))$, where $U(\alpha_i)$ is the set of predecessors of α_i and $\pi(\alpha_i)$ is a subset of $U(\alpha_i)$ that renders α_i conditionally independent in P of all its other predecessors. This set of conditional independence statements is said to *generate* a dag and is called a *recursive basis* drawn from P.

Definition. Given a probability distribution P on a set of variables U, a dag $D = (U,E)$ is called a *Bayesian network* of P iff D is a minimal-edge I-map of P.

The three theorems below summarize the discussion above and further characterize the independencies that are displayed in a dag D. A preliminary definition is needed.

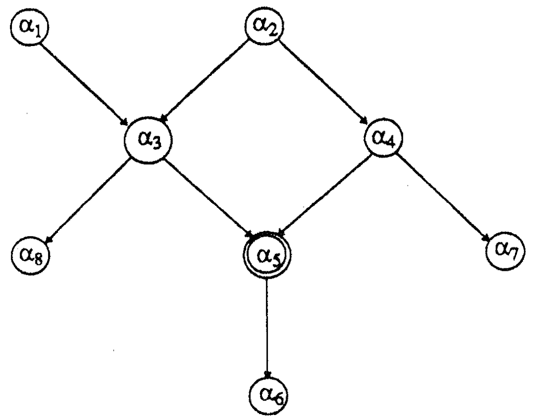

FIG. 2.

Definition. An independence statement T *logically follows* (wrt a class of dependency models $\mathcal{M}$) from a set L of such statements if T holds in every dependency model $M \in \mathcal{M}$ that satisfies L. In this case, we also say that T is a *semantic consequence* of L (wrt $\mathcal{M}$).

Theorem 1 (soundness) [29]. If M is a graphoid and L is any recursive basis drawn from M, then the dag generated by L is an *I-map of M*.

Theorem 2 (closure) [29]. Let D be a dag generated by a recursive basis L. Then M_D, the dependency model defined by D, is exactly the closure of L under the graphoid axioms.

Theorem 3 (completeness) [9]. Let D be a dag generated by a recursive basis L drawn from a probabilistic model M_P. Then, every semantic consequence of L wrt $\mathcal{M}_{\mathcal{P}}$ holds in D.

The first theorem guarantees that d-separation identifies only independencies that hold in the original graphoid. The second theorem further characterizes the statements identified by d-separation as being exactly the statements derivable from a recursive basis L via the graphoid axioms. The third theorem assures that a dag displays all statements that logically follow from L (wrt $\mathcal{M}_{\mathcal{P}}$), that is, the graphoid axioms are *complete*, capable of deriving all semantic consequences of a recursive basis. Thus, Theorem 3 implies that for any independence relationship not displayed by d-separation there exists an underlying distribution for which this indepenency is violated; hence, we cannot hope to improve the d-separation criterion to display more independencies. Moreover, since a statement in a dag can be verified in linear time (Section 5), Theorem 3 provides a complete polynomial inference mechanism for deriving all independence statements that logically follow from a recursive basis. A generalized version of these three theorems is proven in Section 4. Analogous results are proven in [8] for Markov fields; a representation scheme based on undirected graphs [13,14].

We conclude this section by showing how these thoerems can be employed as an inference mechanism. Assume an expert has identified the following conditional independencies among five variables, denoted α_1 through α_5:

$$L = \{I(\alpha_2,\emptyset,\alpha_1),\ I(\alpha_3,\{\alpha_1,\alpha_2\},\emptyset),\ I(\alpha_4,\alpha_2,\{\alpha_1,\alpha_3\}),\ I(\alpha_5,\{\alpha_3,\alpha_4\},\{\alpha_1,\alpha_2\})\},$$

in which the second statement in L is trivial. We raise two questions: First, what is the set of all semantic consequences of L? Second, in particular, is $I(\alpha_1,\{\alpha_2,\alpha_3,\alpha_5\},\alpha_4)$ a semantic consequence of L? For general lists of independencies, the answer for such questions may be undecidable [6,27], but, since L is a recursive basis, it defines a dag D that graphically verifies each and every semantic consequence of L. This dag is shown in Figure 2, ignoring nodes α_6, α_7, α_8 and their adjacent links. To answer the second question, we simply observe that $I(\alpha_1,\{\alpha_2,\alpha_3,\alpha_5,\}\alpha_4)$ holds in D.

4. NETWORKS WITH DETERMINISTIC NODES

The analysis of Section 3 assumes that the input information L is a recursive basis, containing only statements of the form $I(\alpha,\pi(\alpha),U(\alpha) - \pi(\alpha))$. Occasionally, however, we are in possession of strong forms of independence relationships, in which case additonal statements should be read off the dag. A common example is the case of a variable that is functionally dependent on its corresponding parents in the dag (*deterministic variable*, [22]). The existence of each such variable α could be encoded in L by a statement of *global* independence $I(\alpha,\pi(\alpha),U - \pi(\alpha))$ asserting that, conditioned on its parents $\pi(\alpha)$, α is independent of all other variables, not merely of its predecessors. The independencies implied by these statements can be read from the dag using an enhanced version of d-separation, named, D-separation.

Definition. A node α is (*functionally*) *determined* by Z iff $\alpha \in Z$ or α is a deterministic node and all its parents are functionally determined by Z. If α is a deterministic node with no parents, then it is functionally determined by Z. A set of nodes is determined by Z if each of its members is determined by Z.

Definition. If X, Y, and Z are three disjoint subsets of nodes in a dag D, then Z is said to *D-separate* X from Y, iff there exists no trail t between a node in X and a node in Y along which (1) every node with converging arrows either is or has a descendant in Z, (2) every other node is outside Z, and (3) no tail-to-tail node on t is functionally determined by Z. A trail satisfying the three conditions above is said to be *active*; otherwise, it is said to be *blocked*[3] (by Z).

The new criterion certifies all independencies that are revealed by d-separation plus additional ones due to condition 3 of the definition. In the dag of Figure 2, for example, if node α_5 is functionally determined by its parents, then the independence $I(\alpha_5,\{\alpha_3,\alpha_4\},\alpha_6)_D$ holds in D (by definition of D-separation), conveying the idea that once a node (α_5) is functionally determined, its value becomes independent of the rest of the network, independent even of its immediate successors. It should be noted that the definition of D-separation can be condensed without altering its meaning. This is shown by the following Lemma.

Lemma 4. The following assertions are equivalent:

(a) A trail t is activated by Z, namely, t is a trail along which (a1) every node with converging arrows either is in Z or has a descendant in Z (a2), every other node is outside Z, and (a3) no tail-to-tail node (wrt t) is functionally determined by Z.

[3]For the rest of the paper, the terms *active* and *blocked* will refer to D-separation (not to d-separation as defined in Section 3).

(b) t is a trail along which (b1) every node with converging arrows either is in Z or has a descendant in Z and (b2) no other node is functionally determined by Z.

Proof. Let t be a trail connecting α and β that satisfies the three conditions in (a). Assume, by contradiction, that condition (b2) is violated, namely, that there exists a node α_1 on t that is not a head-to-head node, yet is determined by Z. By (a3) α_1 cannot be a tail-to-tail node. Examine a link in t that points toward α_1, say the link $\alpha_2 \rightarrow \alpha_1$. Since α_1 is determined by Z, either α_2 is in Z, in which case t violates condition (a2) or α_2 is determined by Z. We repeat the same argument for α_2 and obtain the chain $\alpha_3 \rightarrow \alpha_2 \rightarrow \alpha_1$ where either α_3 is in Z or determined by Z. Eventually (since the number of nodes is finite), we either reach a node that is in Z, thus violating condition (a2) or we reach a tail-to-tail node that is determined by Z, in which case the trail violates condition (a3). Thus, both cases contradict our assumption that t satisfies the three conditions stated in (a). The other direction is imediate. Condition (a1) follows from (b1), and (a3) follows from (b2). Condition (a2) follows from (b2) because a node that is not determined by Z must be outside Z. ■

Note that, in principle, to check whether Z D-separates X and Y, the definition requires us to examine all trails connecting a node in X and a node in Y, including trails that form loops. For example, in Figure 2, to check whether $X = \{\alpha_1\}$ and $Y = \{\alpha_8\}$ are D-separated by $Z = \{\alpha_6\}$ would require checking trails such as α_1, α_3, α_5, α_4, α_2, α_3, α_8, and many others. The next lemma states that such trails need not be examined because whenever there is an active trail with a loop there is an active *simple* trail as well, i.e., a trail that forms no cycles in the underling undirected graph. In the previous example, the trail α_1, α_3, and α_8 is the simple trail (given $\{\alpha_6\}$), guaranteed by Lemma 5. The proof can be found in [8].

Lemma 5. Let Z be a set of nodes in a dag D, and let $\alpha, \beta \notin Z$ be two nodes of D. Then α and β are connected via an active trail (given Z) only if α and β are connected via a simple active trail (given Z).

Parallel to the discussion of Section 3, we define a new basis and prove soundness and completeness of D-separation with respect to this basis.

Definition. An *enhanced basis* L drawn from a dependency model M in an ordering $\alpha_1, \ldots, \alpha_n$ of M's elements is a set of n independence statements (i.e., triplets) $(\alpha_i, \pi(\alpha_i), W(\alpha_i)) \in M$, $i = 1 \ldots n$, where $W(\alpha_i)$ is either $U(\alpha_i) - \pi(\alpha_i)$ or $U - \pi(\alpha_i) - \{\alpha_i\}$, $U(\alpha_i) = \{\alpha_1, \ldots, \alpha_{i-1}\}$ and $\pi(\alpha_i) \subseteq U(\alpha_i)$. An enhanced basis is said to *generate* a dag over n nodes where each node α_i corresponds to an element α_i and its parents are those nodes corresponding to the elements in $\pi(\alpha_i)$. When the i-th statement is a global independence $(W(a_i) = U - \pi(\alpha_i) - \{\alpha_i\})$, then node α_i is depicted in D with a double circle to denote that it is a deterministic node. Other nodes, which are called *chance* nodes, are depicted using a single circle.

The next lemma explicates those independencies that are implied from the deterministic nodes alone; it is needed for establishing the soundness of D-separation. For clarity, we will omit the union symbol from complicated expressions and write, as a shorthand notation, XY for $X \cup Y$ and $X\alpha$ for $X \cup \{\alpha\}$.

Lemma 6. Let M be a graphoid and L be any enhanced basis drawn from M. If Z functionally determines X in the dag generated by L, then $(X,Z,U-XZ) \in M$.

Proof. We prove the lemma by induction on the highest index l of an element in X as determined by the ordering of L. If the highest index is 1, then X is a singleton that has no parents in the dag. It is therefore determined by Z only if it is a deterministic node, if which case, $(X,\emptyset,U-X)$ is a member of L (and thus a member in M). By weak union, $(X,Z,U-XZ) \in M$ follows. Otherwise, let $X = X'\alpha$, where α has the highest index in X. Since Z determines X', by the induction hypothesis, the triplet $(X',Z,U-X'Z) \in M$.

We will show that the triplet $(\alpha,Z,U-\alpha Z)$ is also in M and that the last two triplets together imply that $(X,Z,U-XZ) \in M$. The set Z determines α and $\alpha \notin Z$; therefore, Z determines the parents of α, denoted by V, and α is a deterministic node. Since all elements of V have a smaller index than α, by the induction hypothesis, $(V,Z,U-VZ) \in M$ $(\overset{\Delta}{=} J_1)$. The triplet $(\alpha,V,U-V\alpha) \in L(\overset{\Delta}{=} J_2)$ because α is a deterministic node and therefore this triplet is also a member of M. Letting $W = U - VZ\alpha$, J_1 and J_2 are written as

$$(V,Z,W\alpha) \in M \text{ and } (\alpha,V,WZ) \in M,$$

respectively. The triplets $(\alpha,Z,V) \in M$ and $(\alpha,ZV,W) \in M$ are derived from the previous ones, respectively, by summetry, decomposition, and weak union. By using contraction on the latter triplets, it follows that $(\alpha,Z,VW) \in M$. Substituting $U - VZ\alpha$ for W, we obtain that $(\alpha,Z,U-\alpha Z) \in M$.

It remains to be shown that $(X',Z,U-X'Z) \in M$ and $(\alpha,Z,U-\alpha Z) \in M$ imply that $(X,Z,U-XZ) \in M$. Letting $W = U - ZX'\alpha$, we show that

$$(X',Z,W\alpha) \in M \text{ and } (\alpha,Z,WX') \in M \Rightarrow (X'\alpha,Z,W) \in M$$

follows from the graphoid axioms. The two triplets $(W,Z,X') \in M$ and $(W,ZX',\alpha) \in M$ are derived from the antecedents by summetry, decomposition, and weak union. Using contraction on the resulting two triplets and then symmetry yields that $(X'\alpha,Z,W) \in M$. Substituting $U - ZX'\alpha$ for W and X for $X'\alpha$, yields the desired conclusion $(X,Z,U-ZX) \in M$. ■

Theorem 7 (soundness). If M is a graphoid and L is any enhanced basis drawn from M, then the dag D generated by L is an I-map of M.

Proof. Induct on the number of elements in the graphoid. For graphoids of one variable, the single node dag generated is trivially an I-map. Suppose for graphoids with fewer than k elements that the dag generated is an I-map. Let M have k elements, let u be the last element in the ordering of L, let $M[u]$ be the graphoid formed by removing u and all triplets involving u from M, and let $D[u]$ be the dag formed by removing u and all its incident links from D. Additionally, let $L[u]$ be the set of triplets formed from L by removing the last triplet and deleting the element u from the remaining triplets, namely, $L[u]$ is equal to $\{(j,B,R-u) \mid j \neq u,(j,B,R) \in L\}$. The set $L[u]$ is an enhanced basis of $M[u]$, and it generates the dag $D[u]$. Thus, since $M[u]$ has $k-1$ elements, by the induction hypothesis, $D[u]$ is an I-map of it. Let M_D be the dependency model corresponding to the dag D and $M_{D[u]}$ correspond to $D[u]$, (i.e., M_D contains all triplets (X,Z,Y) for which X and Y are D-separated by Z in D). Each triplet T of M_D falls into one of three categories: (1) u does not appear in T, (2) u appears on the first or third entry of T, or (3) u appears in the second entry of T. These will be treated separately as cases 1, 2, and 3, respectively. For each case, we will show that $T \in M_D$ implies that $T \in M$, thus proving that D is an I-map of M.

Case 1. If u does not appear in T, then T must be (X,Z,Y) with X, Y, and Z three disjoint subsets of elements, none of which contain u. Since T is in M_D, it must also be in $M_{D[u]}$, for if it were not, then there would be an active trail (given Z) in $D[u]$ between a node in X and a node in Y. But if this trail were active in $D[u]$, then it would also be active in any dag containing $D[u]$ as a subgraph; specifically, this trail would have remained active in D. By the induction hypothesis, $M_{D[u]}$ is a subset of $M[u]$; thus, T must be an element of $M[u]$. Also, $M[u]$ is a subset of M, so T is in M.

Case 2. If u appears in the first entry of the triplet, then $T = (Xu,Z,Y)$ with X, Y, and Z three disjoint subsets of elements, none of which contain u. Let (u,B,R) be the last triplet in L, B_X, B_Y, B_Z, and B_0 be a partitioning of B and R_X, R_Y, R_Z and R_0 be a partitioning of R such that $X = B_X \cup R_X$, $Y = B_Y \cup R_Y$, and $Z = B_Z \cup R_Z$ as in Figure 3. We first show that $(Y,ZXB_0,u) \in M$. Then, we will show that $(Y,Z,XB_0) \in M$. Since M is a graphoid containing these two statements, it will follow by contraction that $(Y,Z,XB_0u) \in M$ and by

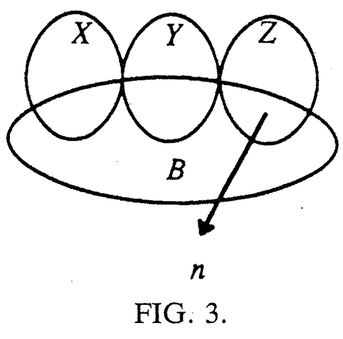

FIG. 3.

decomposition and symmetry that $T = (Xu,Z,Y) \in M$. This will complete the proof of case 2 because if u appears in the third entry of T, namely, $T = (Y,Z,Xu)$, then (Xu,Z,Y) is a member of M_D, which would imply that (Xu,Z,Y) is a member of M, and since M is closed under symmetry, T would be a member of M as well.

We now show that $(Y,ZXB_0,u) \in M$. Consider the set B_Y; any node in this set is directly linked to u. Thus, in order for Y to be D-separated from u given Z, B_Y must be determined by Z in D (or be empty, in which case, by our definitions, it is determined by Z). By Lemma 6, $(B_Y,Z,U - B_YZ) \in M$. Using weak union and decomposition, it follows that (B_Y,ZXB_0,u) is in M. The triplet (u,B,R) in L implies (by decomposition and weak union) that $(u,XZB_0B_Y, R_Y) \in M$. The last triplet together with (B_Y,ZXB_0,u) imply (by symmetry and contraction) that $(Y,ZXB_0,u) \in M$.

It remains to be shown that $(Y,Z,XB_0) \in M$. The triplet (Y,Z,B_0) must belong to M_D since otherwise there would have been an active trail between a node in Y and a node in B_0 that could have been augmented to form an active trail (by Z) between Y to u, by using the link that connects any element in B_0 to u (pointing to u). This would contradict the assumption that $(u,Z,Y) \in M_D$, as implied by decomposition from $(Xu,Z,Y) \in M_D$. Thus, $(Y,Z,B_0) \in M_D$. $(Y,Z,X) \in M_D$ because it is implied by decomposition from the fact that (Xu,Z,Y) is in M_D. By the definition of D-separation, two sets are D-separated iff each of their individual elements is D-separated. Therefore, $(Y,Z,X) \in M$ and $(Y,Z,B_0) \in M$ imply that (Y,Z,XB_0) must also be in M_D. The last triplet does not contain u; thus, by the argument of case 1, $(Y,Z,XB_0) \in M$.

Case 3. If u appears in the second entry, then $T \in M_D$ has the form (X,Zu,Y). The triplet (X,Z,Y) must be a member of M_D as well, for if there were an active trail (given Z) between a node in X and a node in Y, this trail would have remained activated by Zu because u is a sink on that trail. This would contradict our assumption that $(X,Zu,Y) \in M_D$. The triplets $(X,Z,Y) \in M_D$ and $(X,Zu,Y) \in M_D$ imply that either $(X,Z,u) \in M_D$ or $(u,Z,Y) \in M_D$.[4] By definition of D-separation, two sets are D-separated iff each of their individual elements is D-separated. Therefore, $(X,Z,Y) \in M_D$ and the disjunction above imply that either $(X,Z,Yu) \in M_D$ or $(Xu,Z,Y) \in M_D$. By the argument of case 2, it follows that either $(X,Z,Yu) \in M$ or $(Xu,Z,Y) \in M$. In both cases, it follows by weak union and symmetry that $(X,Zu,Y) \in M$. ■

Theorem 8 (closure). If L is an enhanced basis drawn from an arbitrary dependency model M, the dag dependency model M_D generated from L is a perfect map of the closure $cl(L)$ of L under the graphoid axioms. In other words, a triplet belongs to M_D if and only if it can be derived from the triplets of L using the four graphoid axioms.

[4]Pearl [19 (p. 129)] proves this property, called weak-transitivity, for d-separation, but exactly the same proof applies also to D-separation.

Proof. By the previous theorem, $M_D \subseteq cl(L)$. It remains to be shown that $cl(L) \subseteq M_D$. We will show, instead, that $L \subseteq M_D$. This will imply that $cl(L) \subseteq cl(M_D)$, and since every dag dependency model M_D is a graphoid, it must be the case that $cl(M_D) = M_D$, which will complete the proof. Let (u,B,R) be a triplet in L. There are two cases: R does not contain any successor of u in which case u is a chance node, or R contains all of u's successors, in which case u is a deterministic node. If u is a deterministic node, then its parents B D-separate it from any other node; thus $(u,B,R) \in M_D$. If u is a chance node, then u is D-separated from R given B in the dag [i.e., $(u,B,R) \in M_D$], for if not, there would be a trail from a node in R to u that is active given B. But since every link into u emitted from B, the trail must lead out of u into some node that was placed after u. Since, in the case of a chance node, every node in R was placed before u, this trail must contain a head-to-head node that was placed after u. But this trail cannot be activated by B since B contains no nodes placed after u, and thus, B would D-separate u from R in the dag. ■

The completeness proof for D-separation requires the following lemma.

Lemma 9. For every dag D and a triplet $T = (\alpha,Z,\beta) \notin M_D$, there exists a dag D' with the following properties:

1. $D' = (E',V)$ is a subgraph of $D = (E,V)$, i.e., $E' \subseteq E$.
2. $(\alpha,Z,\beta) \notin M_{D'}$
3. The links of D' consist exclusively of the following three sets:
   a. A trail q between α and β.
   b. A single directed path p_i from every head-to-head node h_i on q to a distinct member z_i of Z. The paths p_j's do not share any node with each other and each p_i intersects q only at h_i.
   c. For each functional tail-to-tail node t_i on q, D' contains a directed path r_i from some chance node l_i to t_i such that l_i is the ony chance node on r_i and the entire path includes no nodes of Z. The paths r_j's do not share any node with each other or with any p_i and each path r_i intersects q only at node t_i.

A dag satisfying the three conditions above is called an *$\alpha\beta$-trail dag.*

Proof. We first construct the dag D' and then prove it satisfies the requirements. Let q be an active trail (given Z) between α and β with a minimum number of head-to-head nodes denoted, from α to β, $h_1, h_2, \ldots, h_k$. Such a trail exists because $T \notin M_D$. Let z_i be the closest (wrt path length) descendent of h_i in Z, and let p_i be a directed path from h_i to z_i (if $h_i \in Z$ then $z = h_i$). Let $t_1 \ldots t_l$ be all deterministic tail-to-tail nodes on q. Let l_i be the closest chance node, not in Z, that is an ancestor of t_i such that q_i, a shortest directed path connecting l_i and t_i, includes no nodes of Z. The paths p_i's exist because the trail q is active only if every h–h node on it is or has a descendent in Z. The paths q_i's exist in D because otherwise t_i would have been functionally

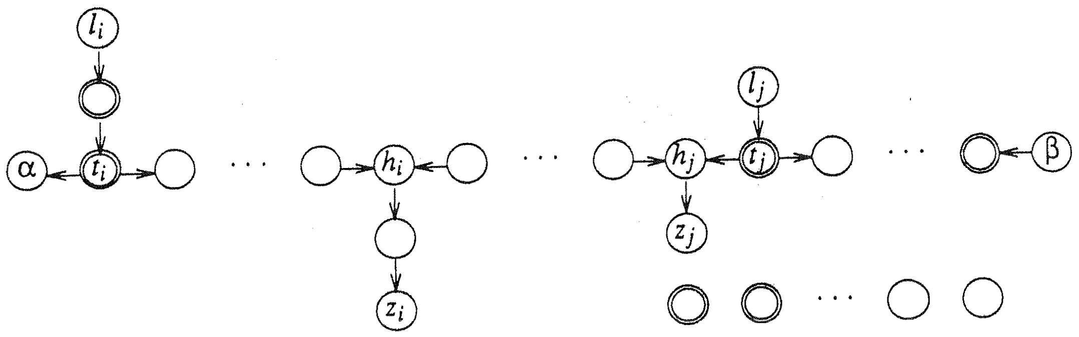

FIG. 4.

determined by Z and the trail q would not have been active. Let $D' = (E',V)$ where E' consists exclusively of the links contained in q, p_i's and q_i's (e.g., Fig. 4).

By our construction, D' satisfies conditions 1, 2, and 3a. Next we prove that it satisfies requirement 3b. First we claim that the path p_i's are distinct. Assume, by contradiction, that there are two paths p_i and p_j $(i < j)$ with a common node γ (Fig. 5). Under this assumption, we find an active trail between α and β that has fewer head-to-head nodes than q, contradicting the minimality of the latter. If γ, the common node, is neither h_i nor h_j, then the trail $(\alpha,h_i,\gamma,h_j,\beta)$ is an active trail (given Z); each of its head-to-head nodes is or has a descendant in Z because it is either γ or a head-to-head node of q and every node that has been added is not determind by z. Every other node δ lies either on the active trail q and therefore is not determined by Z (Lemma 4) or it lies on either p_i or p_j. In either of the last two cases, since z_i or z_j are closest descendants of h_i or h_j, respectively, δ must be outside Z. The resulting active trail contradicts the minimimality of q since both h_i and h_j are no longer head-to-head nodes whereas γ is the only newly introduced head-to-head node. If $\gamma = h_j$, then the trail $(\alpha,h_i,\gamma,h_j,\beta)$ shrinks to be $(\alpha,h_i,\gamma,\beta)$, which, using similar arguments, has fewer head-to-head nodes than q and is activated by Z (the case $\gamma = h_i$ is symmetric).

We complete the proof of 3b by showing that each path p_i intersects q only at node h_i. Assume, by contradiction, that p_i and q have in common a node γ other than h_i and assume that it lies between h_i and β (the case where γ lies between h_i and α is symmetric) (Fig. 6). It has been shown that p_i is distinct from all other p_j's; therefore, in particular, node γ is not a head-to-head node

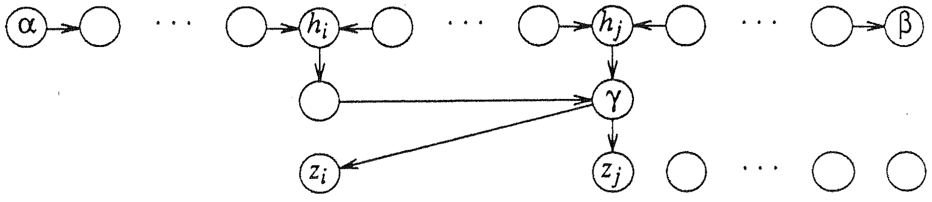

FIG. 5.

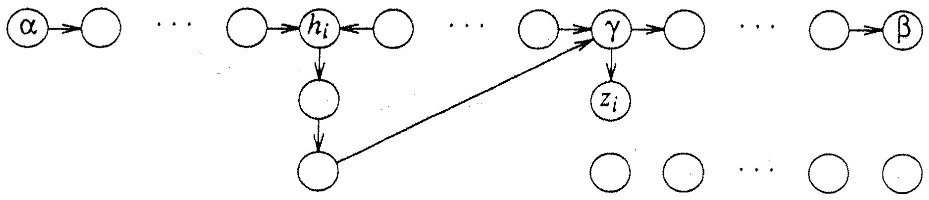

FIG. 6.

on q. Thus, γ cannot belong to Z because otherwise q were blocked by γ and thus would not have been active. Hence, the trail $q' = (\alpha,h_i,\gamma,\beta)$ is activated by Z. The trail q' contradicts the minimality of q because h_i is no longer a head-to-head node on q' while no new head-to-head nodes are introduced.

To prove 3c, we use similar arguments. Assume paths r_i and r_j have a common node γ (e.g., Fig. 7). Then the trail $(\alpha,t_i,\gamma,t_j,\beta)$, depicted in Figure 7, is an active trail that contains fewer head-to-head nodes than does q because the fragment of q between any two tail-to-tail nodes t_i's must contain a head-to-head node while the new bypass does not contain any. The new trail is active because no node on the bypass is determined by Z. Thus, the new trail is active and therefore contradicts the minimality of q.

If a node γ is shared by r_i and p_j (Fig. 8), then the trail $(\alpha,t_i,\gamma,h_j,\beta)$ (e.g., Fig. 7) contradicts the minimality of q; it contains fewer head-to-head nodes than does q because h_j is no longer a head-to-head node and no new head-to-head nodes are added. It is active since neither of the nodes on the bypass is in Z nor is determined by Z.

If γ is shared by r_i and q, then the new trail $(\alpha,t_i,\gamma,\beta)$ (e.g., Fig. 9) contradicts the minimality of q. It contains fewer head-to-head nodes than does q because no new head-to-head nodes are added while the fragment of q between t_i and γ must contain a head-to-head node; for otherwise, D would have contained a circle. The new trail is active since no node on the bypass is in Z or is determined by Z. Thus, D' satisfies all the requirements of the lemma. ■

Theorem 7 states that every independency in M_D must be a semantic consequence (of L) wrt $\mathcal{M}$, where $\mathcal{M}$ is any dependency model that satisfies the

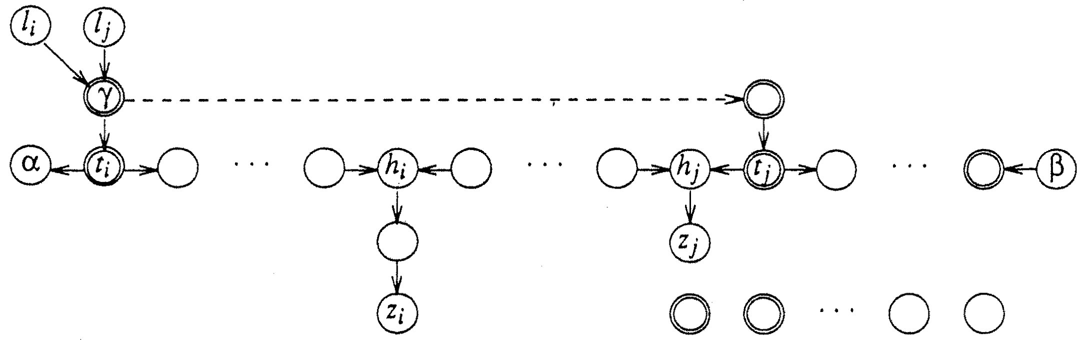

FIG. 7.

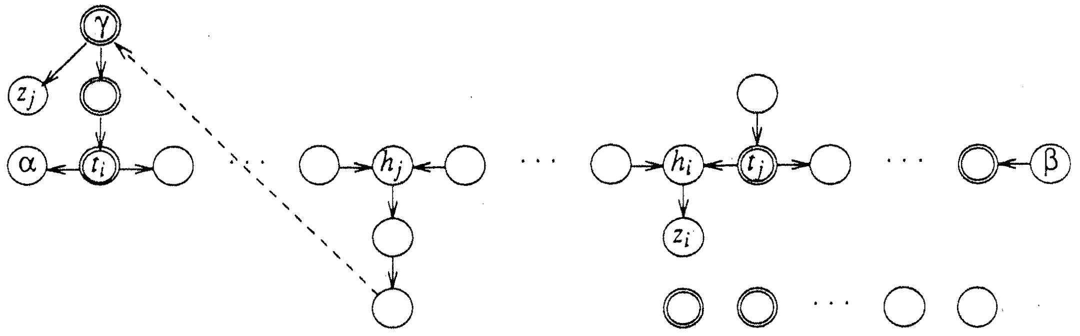

FIG. 8.

graphoid axioms, e.g., probabilistic, relational, and correlational dependency models. Theorem 10 below states the converse for probabilistic dependency models, namely, every semantic consequence wrt $\mathcal{M}_{\mathcal{P}}$ of an enhanced basis L is an independency in M_D. Similar claims for correlational and relational dependency models are stated as corollaries and follow from the proof of Theorem 10.

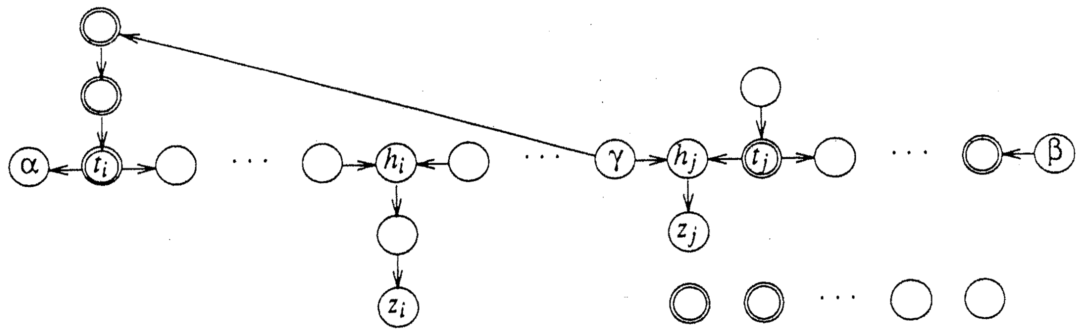

FIG. 9.

Theorem 10 (completeness). Let $D = (E,U)$ be a dag generated by an enhanced basis drawn from a probabilistic dependency model M_P. Then, every semantic consequence of L wrt $\mathcal{M}_{\mathcal{P}}$ holds in D.

Proof. Let $T = (X,Z,Y)$ be an arbitrary triplet not in M_D (we assume that $XZY \subseteq U$ and that U is finite). We construct a distribution P_T whose dependency model P_T[5] contains all triplets of L and does not contain T. This distribution precludes T from being a semantic consequence of L, and, therefore, as the theorem claims, every semantic consequence of L must be a member in M_D.

The triplet $(X,Z,Y) \notin M_D$. Hence, the definition of D-separation guarantees

[5]The symbol P_T is overloaded—sometimes it denotes a distribution and sometimes it denotes the dependency model defined by that distribution. The meaning will be clear from the context.

the existence of an active trail between a node α in X and a node β in Y that is not D-separated by Z. Constructing a distribution P_T that does not contain the triplet (α,Z,β), denoted T', guarantees also that $(X,Z,Y) \notin P_T$ because any distribution that renders X and Y conditionally independent must render each of their individual variables independent as well [decomposition (4b)]. The triplet $(\alpha,Z,\beta) \notin M_D$. Hence, by Lemma 9, there exists an $\alpha\beta$-trail dag D' (Fig. 4). We will construct a distribution P_T whose dependency model contains all triplets of $M_{D'}$ (i.e., $M_{D'} \subseteq P_T$) and that does not contain T. This will complete the proof. By property (i) of Lemma 9, $M_D \subseteq M_{D'}$. By Theorem 7, $L \subseteq M_D$. Thus, $L \subseteq P_T$, as required by the theorem.

P_T is defined as follows: Each chance node with no parents corresponds to an independent fair binary coin. Every other node corresponds to a variable that is the sum modulo 2 of the variables corresponding to its parents. A deterministic node with no parents (a degenerate configuration) corresponds to a binary variable whose value is known with certainty. It remains to be shown that P_T satisfies the requirements. Variables α and β are conditionally dependent given Z in P_T because constraining α and Z to some specific values determines a value for β via the single trail q that connects them in D'. It remains to be shown that $M_{D'} \subseteq P_T$. Let L' be an enhanced basis that generates D' in the following ordering of the nodes of D': All nodes that have no parents appear first in the ordering, followed by the rest of the nodes in any order compatible with the partial order defined by D' (e.g., α must precede β if $\alpha \rightarrow \beta$ in a link in D'). The basis L' is contained in P_T because all chance nodes with no parents correspond to mutually independent variables and every other variable in P_T is a function of the variables corresponding to its parents and, therefore, it must be independent from all its other predecessors and successors in the ordering of L'. Thus, $L' \subseteq P_T$. Taking the closure under the graphoid axioms on both sides yields $cl(L') \subseteq cl(P_T)$. However, $P_T = cl(P_T)$ because P_T is a graphoid and $cl(L') = M_{D'}$ (by Theorem 8). Thus, $M_{D'} \subseteq P_T$. ■

Note that in the proof of Theorem 10 we used only bivalued variables. The same construction can be used when the variables are preassigned arbitrary (finite) domains; we need only replace the sum modulo 2 by sums modulo the size of the domain of each child node. This renders the completeness of Theorem 10 stronger, since the class of probabilities considered can be narrowed to those defined over variables having prespecified domains.

Dags have been used also as a representation scheme for structural equations [3]. Each node represents a variable that is the linear combination of the variables corresponding to its parents and a term representing noise. The noise sources are assumed to be independent and normally distributed and have zero means and nonzero variances. Thus, the variable corresponding to node x is given by

$$x = a_1 z_1 + \cdots + a_k z_k + z, \tag{6}$$

where $z_1, \ldots, z_k$ are the variables corresponding to the parents of x and z is the noise term. In this interpretation, two variables are "independent" given a set of variables Z, denoted by $I(\alpha,Z,\beta)_c$ iff $\rho_{\alpha\beta.Z} = 0$, where $\rho_{\alpha\beta.Z}$ is defined recursively by

$$\rho_{\alpha\beta.Z\gamma} = \frac{\rho_{\alpha\beta.Z} - \rho_{\alpha\gamma.Z}\rho_{\beta\gamma.Z}}{(1-\rho^2_{\alpha\gamma.Z})^{1/2}(1-\rho^2_{\beta\gamma.Z})^{1/2}}$$

and

$$\rho_{\alpha\beta} \triangleq \rho_{\alpha\beta.\emptyset} = \frac{E[\alpha\beta - E[\alpha\beta]]}{(E[\alpha - E[\alpha])^{1/2}(E[\beta - E[\beta]])^{1/2}},$$

where $E[x]$ is the mean of x [1]. Each structural equation *corresponds* to one independence statement. For example, Eq. (6) asserts that $I(x,\{z_1 \ldots z_k\}, U(x) - \{z_1, \ldots, z_k\})_c$, where $U(x)$ are the variables preceding x in a given total ordering. The soundness of D-separation in this representation follows because I_c, being identical to conditional independence for normal distributions, must satisfy the graphoid axioms. Completeness is shown by Corollary 11, following the proof of Theorem 10. A similar, but more restricted, completeness theorem for $I(\alpha,Z,\beta)_c$, where Z is a singleton, is given in [11]. We remark that no functional dependencies exist in this interpretation of dags, because all noise terms have nonzero variances; thus, D-separation coincides with d-separation.

Corollary 11 (completeness). Let D be a dag generated by a set of structural equations, and let L be the corresponding enhanced basis. Then, every semantic consequence of L wrt $\mathcal{M}_C$ holds in D.

Proof. In normal distributions, two variables α and β are conditionally independent given Z iff the partial correlation $\rho_{\alpha\beta.Z} = 0$ [1]. Therefore, it suffices to construct a normal distribution N_T with the same properties P_T had in the proof of Theorem 10. We define N_T as follows: Each chance node with no parents corresponds to the outcome of an independent normal variable. Every other node corresponds to a variable that is a (noisy) sum of the variables corresponding to its parents. Deterministic nodes are not present. N_T is a multivalued normal distribution. The proof that N_T fulfills the requirements is the same as in the proof of Theorem 10. ■

The following corollary shows that D-separation is also complete when dags represent the EMVDs and functional dependencies encoded in a data base.

Corollary 12 (completeness). Let D be a dag generated by an arbitrary basis L. Then, every semantic consequence of L wrt $\mathcal{M}_{\mathcal{R}}$ holds in D.

Proof. Let P_T be the distribution constructed in the proof of Theorem 10, and let R_T be the relation defined to be the set of tuples for which P_T has a positive probability. R_T (viewed as a dependency model) contains P_T. P_T contains $M_{D'}$, where D' is an $\alpha\beta$-trail dag constructed in the proof of Theorem 10. Thus, R_T contains $M_{D'}$. Variables α and β are conditionally dependent given

Z in R_T because constraining α and Z to some specific values determines a value for β via the single trail q that connects them in D'. ■

5. A LINEAR TIME ALGORITHM FOR IDENTIFYING INDEPENDENCE

Conditional independence assertions encoded in Bayesian networks can be used for identifying what information and which parameters may be needed for performing a given computation. The analysis of Section 4 guarantees that the D-separation criterion could identify the maximal set of variables that are independent of a set of variables X, given another set Z, without resorting to numerical calculations. However, it does not provide an efficient algorithm to do so. The algorithm we develop in this section is a variant of the well-known Breath First Search algorithm; it finds all nodes reachable from X through an active trail (given Z); hence, the maximal set of nodes Y satisfying $I(X,Z,Y)_D$. This task can be viewed as an instance of a more general problem of finding a path in a directed graph for which some specified pairs of links are restricted not to appear consecutively. In this context, D-separation serves to specify such restrictions. For example, two links, $u \rightarrow v$, $v \leftarrow w$, cannot appear consecutively in an active trail unless $v \in Z$ or v has a descendent in Z. The following notations are employed: $D = (V,E)$ is a directed graph, not necessarily acyclic, where V is a set of nodes, $E \subseteq V \times V$ is the set of directed links, and $F \subseteq E \times E$ is a list of pairs of adjacent links that cannot appear consecutively (F connotes fail). We say that an ordered pair of links (e_1,e_2) is *legal* iff $(e_1,e_2) \notin F$ and that a path is *legal* if every pair of adjacent links on it is legal. We emphasize that by "path" we mean a directed path, not a trail.

First, we devise a simple algorithm for the following problem: Given a finite directed graph $D = (V,E)$, a subset $F \subseteq E \times E$ of illegal pairs of links, and a set of nodes X, find all nodes reachable from X via a legal path in D. The algorithm and its proof of correctness are a slight modification of those found in [4].

Algorithm 1

Input: A directed graph $D = (V,E)$, a set of illegal pairs of links F, and a set of nodes X.

Output: A labeling of the nodes such that a node is labeled with R (connoting "reachable") iff it is reachable from X via a legal path.

```
(i)   Add a new node $s$ to $V$, and for each $\alpha \in X$, add the link $s \rightarrow \alpha$
      to $E$ and label them with 1. Label $s$ and all $\alpha \in X$ with $R$. Label
      all other nodes with "undefined."
(ii)  $i := 1$.
(iii) Find all unlabeled links $v \rightarrow w$ adjacent to at least one link
      $u \rightarrow v$ labeled $i$, such that $(u \rightarrow v, v \rightarrow w)$ is a legal pair. If no
      such link exists, stop.
(iv)  Label each link $v \rightarrow w$ found in Step (iii) with $i + 1$ and the
      corresponding node $w$ with $R$.
(v)   $i := i + 1$, Go to Step (iii).
```

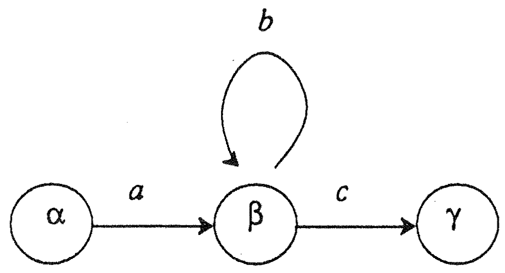

FIG. 10.

The main difference between this algorithm and BFS, a change that has been proposed by Gafni [7], is the traversal of the graph according to a labeling of the links and not according to a labeling of nodes. This change is essential as the example in Figure 10 shows. Let F consist of one pair (a,c). The path from α to γ through links a, b, and c is legal, whereas the path not traversing b is not legal because $(a,c) \in F$. BFS with node labeling would not identify the legal path $(\alpha,\beta,\beta,\gamma)$ connecting nodes α and γ, because it visits every node, in particular β, only once.

Lemma 10. Algorithm 1 labels with R all nodes that are reachable from s (and thus from X) via a legal path, and only those nodes.

Proof. First, we show that if a node α_l is labeled with R, then there exists a legal path from s to α_l. Let $\alpha_{l-1} \rightarrow \alpha_l$ be a link through which α_l has been labeled. We induct on the label l of the link $\alpha_{l-1} \rightarrow \alpha_l$. If $l = 1$ then $\alpha_l \in X$ and is therefore reachable from s. If $l > 1$, then by step (iii), there exists a link $\alpha_{l-2} \rightarrow \alpha_{l-1}$ labeled with $l-1$ such that $(\alpha_{l-2} \rightarrow \alpha_{l-1}, \alpha_{l-1} \rightarrow \alpha_l)$ is a legal pair. Repeatedly applying this argument for $i = l \ldots 2$ yields a legal path $\alpha_0 \rightarrow \alpha_1 \rightarrow \cdots \alpha_l$, where $\alpha_0 \rightarrow \alpha_l$ is labeled with 1. However, the only links labeled 1 emanate from s; hence, the above path is the required legal path from s to α_l.

It remains to be shown that each node that is reachable from s via a legal path is labeled with R. Instead, we show that every link $\alpha \rightarrow \alpha_m$ that is reachable from s via a legal path (i.e., it participates in a legal path emanating from s) will eventually be labeled by the algorithm. The latter claim is stronger than is the former because for every reachable nodes α_m there exists a reachable link $\alpha \rightarrow \alpha_m$ and by step (iv), whenever $\alpha \rightarrow \alpha_m$ is labeled with some integer, α_m is labeled with R. We continue by contradiction. Let $l_m = \alpha_{m-1} \rightarrow \alpha_m$ be the closest link to s via a legal path that remains unlabeled. Let $p = s \rightarrow \alpha_1 \rightarrow \cdots \alpha_{m-1} \rightarrow \alpha_m$ be the legal path emanating from s and terminating with the link l_m. The portion of this path that reaches the link $l_{m-1} = \alpha_{m-2} \rightarrow \alpha_{m-1}$ is shorter than p. Thus, by the induction hypothesis, l_{m-1} is labeled by the algorithm. Hence, the link l_m is labeled as well [by the next application of step (iv)], contradicting our assumption that it remains unlabeled. ■

The complexity of Algorithm 1 for a general F is $O(|E| \cdot |V|)$. In the worst case, each of the $|V|$ nodes might be reached from $|V| - 1$ entry points and, for each entry, the remaining links may need to be examined afresh for reachability. (For example, link c in the example of Fig. 10 is examined twice.) Thus, in the worst case, a link may be examined $|V - 2|$ times before it is labeled, which leads to an $O(|E| \cdot |V|)$ complexity. However, for the special case where F is induced by the D-separation criterion, we shall later see that each link is examined only a constant number of times; therefore, the complexity reduces to $O(|E|)$.

Next we solve the problem of identifying the set of nodes that are D-separated from X by Z. For this aim, we will construct a directed graph D' with a set of legal pairs such that a node v is reachable from X via an **active trail** (given Z) in D iff v is reachable from X via a **legal path** in D'. Algorithm 1 is then applied to solve the latter problem. The following observations are the basis of our algorithm. First, any link on a trail can be traversed both ways. Therefore, to ensure that every active trail in D corresponds to a legal (directed) path, D' must consist of all links of D in their forward and reverse direction. Second, constructing a table that for each node indicates whether it is determined by or has a descendant in Z, would facilitate a constant-time test for legal pairs in D'.

Algorithm 2

Input: A Bayesian network $D = (V,E)$ and two disjoint sets of nodes X and Z.
Data structure: A list of incoming links (in-list) for each node $v \in V$.
Output: A set of nodes Y where $Y = \{\beta | I(X,Z,\beta)_D\}$.

(i) Construct the following tables:

$$\text{determined } [v] = \begin{cases} \text{true} & \text{if } v \text{ is determined by } Z \\ \text{false} & \text{otherwise.} \end{cases}$$

$$\text{descendant } [v] = \begin{cases} \text{true} & \text{if } v \text{ is or has a descendant in } Z \\ \text{false} & \text{otherwise.} \end{cases}$$

(ii) Construct a directed graph $D' = (V,E')$ where

$$E' = E \cup \{(u \to v) | (v \to u) \in E\}.$$

(iii) Using Algorithm 1, find the set of all nodes Y' that have a legal path from X in D', where a pair of links $(u \to v, v \to w)$ is legal iff $u \neq w$ and either (1) v is a head-to-head node on the trail $u—v—w$ in D and descendant$[v]$ = true or (2) v is not a head-to-head node on the trail $u—v—w$ in D and determined $[v]$ = false.

(iv) $Y = V - (Y' \cup X \cup Z)$

Return (Y).

The correctness of this algorithm is established by the following theorem.

Theorem 13. The set Y returned by the algorithm is exactly $\{\beta | I(X,Z,\beta)_D\}$.

Proof. The set Y' constructed in step (iii) contains all nodes reachable from X via a legal path in D'. For any two nodes $\alpha_0 \in X$ and $\beta \notin X \cup Z$, if $\alpha_0 - \alpha_1 \ldots \beta$ is an active trail (given Z) in D, then the directed path $\alpha_0 \rightarrow \alpha_1 \rightarrow \cdots \beta$ is a legal path in D', and vice versa. Thus Y' contains all nodes not in $X \cup Z$ that are reachable from X via an active trail (given Z) in D. By definition of D-separation, $I(X,Z,\beta)_D$ holds if $\beta \notin X \cup Z$ and β is not reachable from X (by an active trail by Z). Thus, $Y = V - (Y' \cup X \cup Z)$ is exactly the set $\{\beta | I(X,Z,\beta)_D\}$. ■

Next, we show that the complexity of the algorithm is $O(|E|)$. The construction of *descendant*$[v]$ is implemented as follows: Initially mark all nodes of Z with true. Follow the incoming links of the nodes in Z to their parents and then to their parents and so on. This way, each link is examined at most once; hence, this construction requires $O(|E|)$ operations. The construction of *determined*$[v]$ is similar and requires the same complexity. Step (ii) of Algorithm 2 requires the construction of a list for each node that specifies all the links that emanate from v in D (out-list). The in-list and the out-list completely and explicitly specify the topology of D'. This step also requires $O(|E|)$ steps. Using the two lists the task of finding a legal pair in step (iii) of Algorithm 1 requires only constant time; if $e_i = u \rightarrow v$ is labeled i, then depending on the direction of $u - v$ in D and whether v is determined by or has a descendent in Z, either all links of the out-list of v, or all links of the in-list of v, or both are selected. Thus, a constant number of operations per encountered link is performed. Hence, step (iii) requires no more than the $O(|E|)$ operation, which is therefore the upper bound (assuming $|E| \geq |V|$) for the entire algorithm.

The above algorithm can also be employed to verify whether a specific statement $I(X,Z,Y)_D$ holds in a dag D. Simply find the set $Y_{\max}$ of all nodes that are D-separated from X given Z and observe that $I(X,Z,Y)_D$ holds in D iff $Y \subseteq Y_{\max}$. In fact, for this task, Algorithm 2 can slightly be improved by forcing termination once the condition $Y \subseteq Y_{\max}$ has been detected. Recently, another algorithm for the same task (for networks without deterministic nodes) has been proposed [16]. The algorithm consists of the following steps: First, form a dag D' by removing from D all nodes that are not ancestors of any node in $X \cup Y \cup Z$ (and removing their incident links). Second, form an undirected graph G, called the *moral graph*, by stripping the directionality of the links of D' and connecting any two nodes that have a common child in D' that is or has a descendent in Z. $I(X,Z,Y)_D$ holds by the definition of d-separation iff all undirected paths between X and Y in G are intercepted by Z.

The complexity of the moral graph algorithm is $O(|V|^2)$ because the moral graph G may contain up to $|V|^2$ links, and, so, checking separation in G might require, in the worst case, $O(|V|^2)$ steps. Our algorithm requires $O(|E|)$ steps, which is a significant gain for sparse graphs, namely, graphs having $|E| = O(|V|)$. If the maximal number of parents of each node is bounded by a constant, then the two algorithms achieve the same asymptotic behavior, i.e., linear in $|E|$. However, the moral graph algorithm is conceptually simpler to communicate, and for small graphs, might offer computational advantages as well.[6] On the other hand, when the task is to find *all* nodes d-separated from X by Z, then a brute force application of the moral graph algorithm requires $O(|V|^3)$ steps, because for each node not in $X \cup Z$, the algorithm must construct a new moral graph. For this task, our algorithm offers a considerable improvement.

An area where Bayesian networks have been used extensively is decision analysis; an analyst elicits information from an expert about a decision problem, formulates the appropriate network, and, then, by an automated sequence of graphical and probabilistic manipulations, an optimal decision is obtained [12,17,22]. When such a network is constructed it is important to determine what information is needed to answer a given query $P(x|z)$ (where $X \cup Z$ is an arbitrary set of nodes in the network), because eliciting irrelevant parameters may be a waste of effort [22]. Assuming that each node α_i stores the conditional distribution $P(a_i|\pi(a_i))$, the task is to identify the set Q of chance nodes that must be consulted in the process of computing $P(x|z)$ or, alternatively, the set of chance nodes that can be assigned arbitrary conditional distributions without affecting the quantity $P(x|z)$. The required set can also be identified by the D-separation criterion. We represent the parameters p_i of the distribution $P(a_i|\pi(a_i))$ as value in the domain of a dummy parent π_i of node α_i. This is clearly a legitimate representation complying with the format of Eq. (1), since for every node α_i, $P(a_i|\pi(a_i))$ can also be written as $P(a_i|\pi(a_i),p_i)$, so π_i can be regarded as a parent variable of α_i. From Theorem 1, all dummy nodes that are D-separated from X by Z represent variables that are conditionally independent of X given Z, and, so, the information stored in these nodes can be ignored. Thus, the information required to compute $P(x|z)$ resides in the set of dummy nodes that are not D-separated from X given Z. Moreover, the completeness of D-separation further implies that Q is minimal; no node in Q can be exempted from processing on purely topological grounds (i.e., without considering the numerical values of the probabilities involved). The algorithm below summarizes these considerations.

Algorithm 3

Input: A Bayesian network, two sets of nodes X and Z.

Output: A set of nodes Q that contains sufficient information to compute $P(x|z)$.

```
(i) Construct a dag $D'$ by augmenting $D$ with a dummy node $v'$
    for every chance node $v$ in $D$ and adding a link $v' \rightarrow v$.
```

[6]The average complexity of Algorithm 2 can be reduced by adapting the first step of the moral graph algorithm, but the worst case complexity would not be improved.

(ii) Use Algorithm 2 to compute the set Y' of nodes not D-separated from X by Z.

(iii) Q is the set of all dummy nodes v' that are included in Y'.

Note that the algorithm adds dummy nodes only to chance nodes. Hence, the algorithm should not be used to detect those functional relationships that could be ignored; it identifies, however, the set of probabilistic parameters that are sufficient for a computation of $P(x|z)$. In order to identify the functional relationship that could be ignored, a more elaborated algorithm is required. The subtle point is illustrated in Shachter's example (8d) and his algorithm addresses this task [23].

We conclude with an example. Consider the network D of Figure 11(a) and query $P(a_3)$.

The computation of $P(a_3)$ requires only to multiply the matrices $P(a_3|a_1)$ and $P(a_1)$ and to sum over the values of α_1. These two matrices are stored at the dummy nodes β_1 and β_3 of Figure 11(b), which are the only dummy nodes not D-separated from node α_3 (given $\emptyset$). Thus Algorithm 3 reveals the fact that the parameters represented by node β_2 and β_4 ($P(a_2)$,$P(a_4|a_1,a_2)$) are not needed for the computation of $P(a_3)$. Note that the questions of the value of a variable, or the parameters stored with the variable influencing a given computation, may result in two different answers. For example, the value of α_4 might influence the computation of $P(a_3)$, because α_3 and α_4 could be dependent, whereas the parameters stored at node α_4 never affect this computation. Algorithm 3, by representing parameters as dummy variables, reveals this fact.

Shachter was the first to present an algorithm that identifies irrelevant parameters using transformations of arc-reversal and node-removal [22]. A revised algorithm of Shachter [23] also detects irrelevant variables, and it appears that the outcome of this algorithm is identical to ours. In our approach, we maintain a clear distinction between the following two tasks: (1) declarative characterization of the independencies encoded in the network (i.e., the D-separation criterion) and (2) procedural implementation of the criterion defined in (1). Such separation facilitates a formal proof of the algorithm's soundness, completeness, and optimality. In Shachter's treatment, task (1) is inseparable from (2). The axiomatic basis on which our method is grounded also provides means for extending the graphical criteria to other notions of independence, such as data base and correlational dependencies.

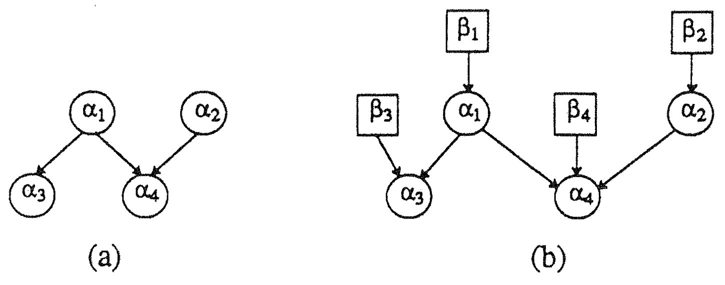

FIG. 11.

ACKNOWLEDGMENT

We thank Eli Gafni for his help in developing Algorithm 1 and Azaria Paz and Ross Shachter for many stimulating discussions.

REFERENCES

[1] H. Cramér, *Mathematical Methods of Statistics*, Princeton University Press, Princeton, NJ (1964).
[2] A. P. Dawid, Conditional independence in statistical theory. *J. R. Stat. Soc. B* **41**(1) (1979) 1-31.
[3] O. D. Duncan, *Introduction to Structural Equation Models*, Academic Press, New York (1975).
[4] S. Even, *Graph Algorithms*, Computer Science Press, Potomac, MD (1979).
[5] R. Fagin, Multivalued dependencies and a new form for relational databases. *ACM Trans. Database Syst.* **2**(3) (1977) 262–278.
[6] R. Fagin and M. Vardi, The theory of data dependencies—A survey. *Proc. Symp. Appl. Math.* **34** (1986) 19–68.
[7] E. Gafni, Personal communication (1988).
[8] D. Geiger, Graphoids—A qualitative framework for probabilistic inference. Ph.D dissertation, UCLA (1990).
[9] D. Geiger and J. Pearl, On the logic of causal models. *Proceedings of the 4th Workshop on Uncertainty in AI*, St. Paul, MN, August (1988) 136–147.
[10] D. Geiger and J. Pearl, Logical and algorithmic properties of conditional independence. Technical Report 870056 (R-97), UCLA Cognitive Systems Laboratory February (1988) To appear in *Ann. Statist.*
[11] C. Glymour, R. Scheines, P. Spirtes, and K. Kelly, *Discovering Causal Structure*, Academic Press, New York (1987).
[12] R. A. Howard and J. E. Mathson, Influence diagrams. *Principles and Applications of Decision Analysis*, Strategic Decisions Group, Menlo Park, CA (1981).
[13] V. Isham, An introduction to spatial point processes and Markov random fields. *Int. Stat. Rev.* **49** (1981) 21–43.
[14] S. L. Lauritzen, *Lectures on Contingency Tables*, 2nd ed., University of Aalborg Press, Aalborg, Denmark (1982).
[15] S. L. Lauritzen and D. J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems. *J. R. Stat. Soc.* **50**(2) (1988) 157–224.
[16] S. L. Lauritzen, A. P. Dawid, B. N. Larsen, and H. G. Leimer, Independence properties of directed Markov fields. *Networks* **20** (1990) 491–505.
[17] S. M. Olmsted, On representing and solving decision problems. Ph.D. Thesis, EES Dept., Stanford University (1983).
[18] J. Pearl, Fusion, propagation and structuring in belief networks. *Artificial Intell.* **29**(3) (1986) 241–288; also in [19].
[19] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, CA (1988).
[20] J. Pearl and A. Paz, Graphoids: A graph-based logic for reasoning about relevance relations. *Advances in Artificial Intelligence-II* (B. Du Boulay et al., Eds.), North-Holland, Amsterdam (1987) 357–363.
[21] J. Pearl and T. Verma, The logic of representing dependencies by directed acyclic graphs. *Proceedings of the AAAI*, Seattle, Washington, July (1987) 374–379.
[22] R. D. Shachter, Probabilistic inference and influence diagrams. *Operations Res.* **36** (1988) 589–604.
[23] R. D. Shachter, An ordered examination of influence diagrams. *Networks* **20** (1990) 535–564.

[24] G. Shafer, P. P. Shenoy, and K. Mellouli, Propagating belief functions in qualitative Markov trees. *Int. J. Approximate Reasoning* **1**(4) (1987) 349–400.
[25] J. Q. Smith, Influence diagrams for statistical modeling. *Ann. Stat.* **17**(2) (1989) 654–672.
[26] W. Spohn, Stochastic independence, causal independence, and shieldability. *J. Phil. Logic* **9** (1980) 73–99.
[27] M. Vardi, Fundamentals of dependency theory. *Trends in Theoretical Computer Science* (E. Borger, Ed.), Computer Science Press, Rockville, MD (1988) 171–224.
[28] T. S. Verma, Some mathematical properties of dependency models. UCLA Cognitive Systems Laboratory, Technical Reports R-103 (1987).
[29] T. Verma and J. Pearl, Causal networks: Semantics and expressiveness. *Proceedings of the 4th Workshop on Uncertainty in AI*, St. Paul, MN (1988) 352–359.

Received December 1989
Accepted March 1990.