In *Proceedings of the Seventh Workshop on Uncertainty Processing (WUPES'06)*, Mikulov, Czech Republic, 111-137, 2006.

TECHNICAL REPORT
R-316
September 2006

A New Graphical Model for the Representation of Marginalized DAG-Representable Relations*

Azaria Paz
Computer Science Department
Technion IIT HAIFA
and The Netanya Academic College
paz@cs.Technion.AC.IL

Abstract

A new model for representing PD-induced relations that are derived from DAG-representable relations through marginalization over a subset of their variables is introduced. The new model requires polynomial space and a polynomial algorithm is given for testing whether a given triplet is represented in the model. In addition a polynomial algorithm is derived for testing whether a marginalized DAG-representable relation is DAG-representable.

1 Introduction

This paper investigates Probabilistic Distribution (PD) induced independency relations which are representable by Directed Acyclic Graphs (DAGs), and are marginalized over a subset of their variables. PD-induced relations have been shown in the literature to be representable as relations that can be defined on various graphical models. All those graphical models have two basic properties: They are compact, i.e., the space required for storing such a model is polynomial in the number of variables, and they are decidable, i.e., a polynomial algorithm exists for testing whether a given independency is represented in the model. In particular, two such models will be encountered in this paper; the DAG model and the Annotated Graph (AG) model. The reader is supposed to be familiar with the DAG-model which was studied extensively in the literature. An ample introduction to the DAG model is included in Pearl [7, 1988], Pearl [8, 2000], and Lauritzen [3, 1996].

*The main part of this work was done while the author visited the Cognitive Systems Laboratory at UCLA and was supported in part by grants from Air Force, NSF and ONR (MURI).

The AG-model in a general form was introduced by Paz, Geva, and Studeny in [6, 2000] and a restricted form of this model, which is all we need for this paper, was introduced by Paz [4, 2003a] and investigated further in Paz [5, 2003b]. For the sake of completeness, we shall reproduce here some of the basic definitions and properties of those models which are relevant for this paper.

Given a DAG-representable PD-induced relation it is often the case that we need to marginalize the relation over a subset of variables. Unfortunately it is seldom the case that such a marginalized relation can be represented by a DAG, which is an easy to manage and a well understood model. The main result of this paper is the introduction of a new model of representation, the Generalized Annotated Graph (GAG) model which is both compact and decidable and can represent any marginalized DAG-representable relation induced by a PD. Moreover, based on the previous paper of the author [5, 2003b], a decision procedure is provided for checking whether a marginalized DAG-representable relation is itself DAG-representable. A different approach to marginalized relations as above can be found in Wermuth and Cox [12, 2000].

2 Preliminaries

2.1 Definitions and notations

UGs will denote undirected graphs $G = (V, E)$ where V is a set of vertices and E is a set of undirected edges connecting between two vertices. Two vertices connected by an edge are adjacent or neighbors. A path in G of length k is a sequence of vertices $v_1 \dots v_{k+1}$ such that (v_i, v_{i+1}) is an edge in E for $i < 1, \dots, k$. A DAG is an acyclic directed graph $D = (V, E)$ where V is a set of vertices and E is a set of directed arcs connecting between two vertices in V. The indegree (outdegree) of a vertex v in D is the number of arcs in E directed into (out of) v. A trail of length k in D is a sequence $v_1 \dots v_{k+1}$ of vertices in V such that (v_i, v_{i+1}) is an arc in E for $i = 1 \dots k$. If all the arcs on the trail are directed in the same direction then the trail is called a directed path. If a directed path exists in D from v_i to v_j then v_j is a descendant of v_i and v_i is a predecessor or ancestor of v_j. If the path is of length one then v_i is a *parent* of v_j who is a child of v_i.

The skeleton of a DAG is the UG derived from the DAG when the orientations of the arcs are removed. A pattern of the form $v_i \rightarrow v_j \leftarrow v_k$ is a collider pattern where v_j is the collider. If there is no arc between v_i and v_k then v_j is an uncoupled collider. A collider is maximal if no directed path exists from it to any other collider. The moralizing procedure is the procedure generating a UG from a DAG, by first joining both parents of uncoupled colliders in the DAG by an arc, and then removing the orientation of all arcs. The edges resulting from the coupling of the uncoupled collider are called moral edges. As mentioned in the introduction UG's and DAG's represent PD-induced relations whose elements are triplets $t = (X; Y|Z)$ over the set of vertices of the graphs. For a given triplet t we denote by $v(t)$ the set of vertices $v(t) = X \cup Y \cup Z$. Two

graph models are equivalent if they represent the same relation.

2.2 DAG-model

Let $D = (V, E)$ be a DAG whose vertices are V and whose arcs are E. D represents the relation $R(D) = \{t = (X; Y|Z)|t \in D\}$ where X, Y, Z are disjoint subsets of V, the vertices in V represent variables in PD, t is interpreted as "X is independent of Y given Z" and $t \in D$ means: t is represented in D. To check whether a given triplet t is represented in D we use the Algorithm L1 below due to Lauritzen et al. [2, 1990].

Algorithm L1:
Input: $D = (V, E)$ and $t = (X; Y|Z)$.

1. Let V' be the set of ancestor of $v(t) = X \cup Y \cup Z$ and let $D'(t)$ be the subgraph of D over V'.
2. Moralize $D'(t)$ (i.e., join all uncoupled parents of uncoupled colliders in $D'(t)$). Denote the resulting graph by $D''(t)$.
3. Remove all orientations in $D''(t)$ and denote the resulting UG by $G(D''(t))$.
4. $t \in G(D''(t))$ iff $t \in D$.

Remark 1 *$t \in G$ where G is a UG if and only if Z is a cutset in G (not necessarily minimal) between X and Y.*

The definition above and the L1 Algorithm show that the DAG model is both compact and decidable.

2.3 Annotated Graph – model

Let $D = (V, E)$ be a DAG. We derive from D an AG $A = (G, K)$ where G is a UG And K is a set of elements $K = \{e = (d, r(d))\}$ as follows: G is derived from D by moralizing D and removing all orientations from it.

For every moral edge d in G we put an element $e = (d, r(d))$ in K such that $d(a, b)$, the domain of e, is the pair of endpoints of the moral edge and $r(d)$, the range of e, is the set of vertices including all the uncoupled colliders in D whose parents are a and b, and all the successors of those colliders. Notice that d denotes both a moral edge and the pair of its endpoints. The relation $R(A)$ defined by the AG A is the relation below:

$$R(A) = \{t = (X; Y|Z)|t \in A\}$$

In order to check whether $t \in A$ we use the algorithm L2 due to Paz [4, 2003a] below.

Algorithm L2
Input: An AG $A = (G, K)$.

1. For every element $e = (d, r(d))$ in K such that $r(d) \cap v(t) = \emptyset$ ($v(t) = X \cup Y \cup Z$). Disconnect the edge (a, b) in G corresponding to d and remove from G all the vertices in $r(d)$ and incident edges. Denote the resulting UG by $G(t)$.

2. $t \in A$ if and only if $t \in G(t)$.

Remark 2 *It is clear from the definitions and from the L2 Algorithm that the AG model is both compact and decidable. In addition, it was shown in [4, 2003a] that the AG model has the following uniqueness property: $R(A_1) = R(A_2)$ implies that $A_1 = A_2$ when A_1 and A_2 are AG's. This property does not hold for DAG models where it is possible for two different (and equivalent) DAGs to define the same relation. In fact the AG (D) derived from a DAG D represents the equivalence class of all DAGs which are equivalent to the given DAG D.*

Remark 3 *The AGs derived from DAG's are a particular case of AGs as defined in Paz et al. [6, 2000] and there are additional ways to derive AGs that represent PD-induced relations which are not DAG-representable. Consider e.g., the example below. It was shown by Pearl [7, 1988 Ch. 3] that every DAG representable relation is a PD-induced relation. Therefore the relation defined by the DAG in Fig. 1 represents a PD-induced relation.*

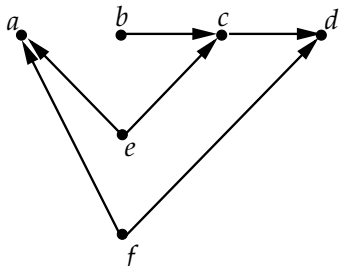


Figure 1: DAG representing relation

If we marginalize this relation over the vertices a, b, c and d we get another relation, PD-induced, that can be represented by the AG A in Fig. 2, under the semantics of the L2 Algorithm, with $R(A) = \{(a; b|\emptyset), (b; d|c)+ \text{symmetric images}\}$. But $R(A)$ above cannot be represented by a DAG. This follows from the following lemma that was proven in [5, 2003b].

Lemma 1 *Let $(G(D), K(D))$ be the annotated graph representation of a DAG D. $K(D)$ has the following properties:*

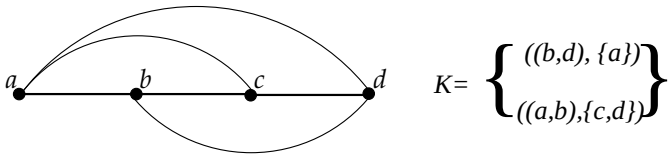


Figure 2: AG A representing a marginalized relation

1. *For every element $((a,b),r) \in K(D)$, there is a vertex $v \in r$ which is a child of both a and b and every vertex $w \in r$ is connected to some vertex v in r whose parents are both a and b.*
2. *For any two elements $(d_1,r_1),(d_2,r_2)$ in $K(D)$, if $d_1 = d_2$ then $r_1 = r_2$.*
3. *For every $((a,b),r) \in K(D)$, (a,b) is an edge in $G(D)$.*
4. *The set of elements $K(D)$ is a poset (=partially ordered set) with regards to the relation "$\succeq$" defined as follows: For any two elements (d_p,r_p) and (d_q,r_q). If $d_p \cap r_q \neq \emptyset$ then $(d_p,r_p) \succ (d_q,r_q)$, in words "$(d_p,r_p)$ is strictly greater than (d_q,r_q)". Moreover $(d_p,r_p) \succ (d_q,r_q)$ implies that $r_p \subset r_q$.*
5. *For any two elements (d_1,r_1) and (d_2,r_2) If $r_1 \cap r_2 \neq \emptyset$ and r_1, r_2 are not a subset of one another, then there is an element (d_3,r_3) in $K(D)$ such that $r_3 \subseteq r_1 \cap r_2$.*

As is easy to see the annotation, K in Fig. 2 does not satisfy the condition 4 of the lemma since the first element in K is bigger than the second but it's range is not a subset of the range of the second element. Therefore A is not DAG-representable.

Remark 4 *An algorithm is provided in [4, 2003a] that tests whether a given AG, possibly derived from a marginalized DAG relation, which satisfies the (necessary but not sufficient) conditions in lemma 1 above, is DAG-representable. The main result of this work is to provide a polynomial algorithm which generates a "generalized annotated graph" representation (concept to be defined in the sequel) which is both compact and decidable. In some cases the generalized annotated graph reduces to a regular annotated graph which satisfies the condition of lemma 1. If this is the case than, using the testing algorithm in [5, 2003b] we can check whether the given AG is DAG-representable. It is certainly not DAG-representable if the generalized annotated graph is not a regular AG or is a regular AG but does not satisfy the conditions of lemma 1.*

Remark 5 *When a given AG A is derived from a DAG then the annotation set $K = \{(d, r(d))\}$ can be interpreted as follows: The edge (a,b), in G, corresponding to d, (a moral edge) represents a conditional dependency. That is: there is some set of vertices, disjoint of $r(d)$, S_{ab} such that $(a_i b|S_{ab})$ is represented in A but a and b become dependent if any proper subset of $r(d)$ is observed i.e., $\neg(a;b|S)$ if $\emptyset \neq S \subseteq r(d)$.*

3 Deriving Annotated Graphs Which are Equivalent to Marginalized DAGs

3.1 Preliminary simplification

Let $A(D) = (G(D), K(D))$ be an AG derived from a given DAG D and equivalent to D, and $R(D)$ is to be marginalized over a subset S of it's variables. We want to find a proper representative for the marginalized relation $R_S(D)$. As a preliminary observation we notice that we may assume that all the vertices in $V\backslash S$, in D, are ancestors of the vertices in S, given that we are interested in $R_S(D)$ only. Otherwise we may reduce D and the corresponding $A(D)$ to a simpler and equivalent, over S, DAG and AG, D' and $A(D')$ as follows:

1. Remove from D and from $G(D)$ all vertices which are not ancestors of S in D.

2. For any element $e = (d, r)$ in $K(D)$ such that r is in $V\backslash S$, remove e from $K(D)$ and disconnect the edge corresponding to e in $K(D)$. Denote the resulting DAG and AG by D' and $A(D')$.

   To show that $A(D')$ represents the same relation as $A(D)$ when marginalized over S we observe that if $e = (d, r)$ in $K(D)$ has the property mentioned in step (2) above then the vertices in r cannot include ancestors of vertices in S: Any vertex in S that has an ancestor in r must be included in r contrary to the fact that $r \in V\backslash S$. Therefore the vertices in r are removed from $G(D)$ in step (1). It follows now, by the L1 Algorithm and it's equivalence to the L2 Algorithm that, over S, $A(D)$ and $A(D')$ represent the same relation.

As a consequence of the above argument it is also clear that an annotated graph $A(D)$ such that all the vertices in $V\backslash S$ are ancestors of vertices in S, has the property that every range of an element in $K(D)$ intersects S (see step 2 of the reduction procedure above). We shall assume hereforth that the given annotated graph $A(D)$, to be marginalized over a subset of vertices S, has the following two properties:

(i) All vertices in D belonging to the set $V\backslash S$ are ancestors of the vertices in S.

(ii) Every range of an element in $K(D)$ intersects S.

We proceed now with the task of constructing a generalized annotated graph (the term will be explained in the sequel) which is equivalent to the marginalization of D over S. We denote this generalized annotated graph by $A_S(D) = (G_S(D), K_S(D))$.

3.2 Procedure M: For constructing $A_S(D)$

3.2.1 Constructing the UG $G_S(D) = (V_S(D), E_S(D))$

Necessarily we must have that $V_S(D) = S$. The derivation of $E_S(D)$ requires a more complex setup. The edges in $E_S(D)$ are separated into strong edges and weak edges where the weak edges are intended to correspond to the domains of the elements in $K_S(D)$. The definition of those edges is given below:

(a) If $a \to b$ is an arc in D then a—b is a strong edge in $E_S(D)$.

(b) Let (u, v) be a pair of vertices in S that are nonadjacent in D. If there is a path in $G(D)$ over $(V \backslash S) \cap$ (ancestor of $\{u, v\}$), connecting u to v and such that the range of every moral edge on the path (if any) intersects $\{u, v\}$ then set u—v to be a strong edge in $E_S(D)$.

(c) Let (u, v) be a pair of vertices in S that are not adjacent in D, and do not satisfy the condition (b) above. If (u, v) is an edge in $G(D)$ or if there is a path in $G(D)$ over $(V \backslash S) \cap \{u, v\}$ connecting u to v, then (u, v) is set as a weak edge in $E_S(D)$. Notice that the first part of the condition is a particular case of the second part when the path consists of a single (moral) edge. The definition of $G_S(D)$ is now completed.

Lemma 2 *If (u, v) is a strong edge in $G_S(D)$ then for any $Z \subseteq S$, $(u; v|Z)$ is not represented in D.*

Proof: The lemma trivially holds for (u, v) satisfying property (a). If (u, v) satisfies (b) and the path connecting u to v, assumed to exist in $G(D)$, includes no moral edges then there is a trail in D corresponding to the path in $G(D)$ having the property that all the vertices on it are ancestors of u or v or both. Therefore when the L1 Algorithm checks whether $(u, v; Z)$ is represented in D, no vertex on the trail is removed. So the trail is transformed into a path in the UG generated by the L1 Algorithm for D and $(u, v; Z)$. The path is not intercepted by any $Z \subseteq S$ since all the vertices on the path are in $V \backslash S$. Thus the L1 Algorithm will decide that $(u, v; Z)$ is not represented in D for any $Z \subseteq S$. If the path includes moral edges then, due to the fact that the ranges of those moral edges intersect $\{u, v\}$, those moral edges will be added (due to the moralization procedure) to the broken trail in D corresponding to the path in $G(D)$ when the L1 Algorithm generates the UG corresponding to D and $(u, v; Z)$. Using the above argument we get again that the L1 Algorithm will decide that $(x, y; Z)$ is not represented in D for any $Z \subseteq S$. □

The above lemma shows that if u and v are connected by a strong edge then they cannot be separated. As mentioned before, weak edges are intended to correspond to the domains of the elements in $K_S(D)$. It turns out however that, for marginalized DAG's it is often impossible to represent them as an annotated graph with (d, r) element in the form shown in the previous sections. We found it necessary therefore to define more general elements (d, G_d) where G_d is an undirected graph itself whose set of vertices is a subset of S.

The construction of those elements, as well as the semantics for deciding whether a triplet is represented in the generalized annotated graph will be shown subsequently. Let u—v be a weak edge in $G_S(D)$. For every such edge we construct an element (d, G_d) where $d = \{u, v\}$ and G_d is a graph whose construction is given in the next section.

3.2.2 The construction of $K_S(D)$

During the following construction some moral edges will be marked and some pairs of consecutive edges (i.e., two edges having a common vertex) will be designated as a forbided transition.

Notation and Definition: The graph $G(\overline{S}, d)$ denotes the subgraph of $G(D)$ over the union of the vertices in d and the vertices in $V \backslash S$, where $d = (u, v)$ is a weak edge in $G_S(D)$. A legal path is a path in $G(S, d)$ which does not include marked edges, does not include forbidden transitions, and may include the vertices u and v only as end vertices of the path (i.e., it does not pass through them).

3.2.2.1 The construction of $\hat{G}_d$

For every weak edge $d = (u, v)$ in $G_S(D)$ we construct first an intermediary graph $\hat{G}_d = (\hat{V}_d, \hat{E}_d)$ as below:

1. If (a, b) is a moral edge in $G(\overline{S}, d)$ such that the range of the domain (a, b) in $K(D)$ has nonempty intersection with $\{u, v\}$ then reset (a, b) as a regular (i.e., nonmoval) edge in $G(\overline{S}, d)$, in connection with the algorithms and procedures described in the sequel, in this section.

2. For every remaining moral edge (a, b) in $G(\overline{S}, d)$ and for every vertex c in $G(\overline{S}, d)$ such that c is a collider in D whose parents are a and b set the pair of edges (a, c) and (c, b) as a forbided transition pair in $G(\overline{S}, d)$.

3. Allocate weights to the edges in $G(\overline{S}, d)$ such that moral edges get weight 1 and all other edges get weight 0.

4. Construct $\hat{G}_d = (\hat{V}_d, \hat{E}_d)$ as follows:

   (a) The vertices u and v are in $\hat{V}_d$.

   (b) Set all other vertices in $\hat{V}_d$ and all edges in $\hat{E}_d$ according to the algorithm below.

Algorithm

0. Create a set W of vertices, $W = \emptyset$

Begin

1. While a legal path connecting u to v through $G(\overline{S}, d)$ exists
   do begin

1.1 Find a legal path of minimal weight connecting u to v through $G(\overline{S}, d)$.

Remark 6 *We will show that at least one such path must exist and that every such path must have moral edges on it.*

1.2 Let (p, q) be the moral edge on the above legal path closest to u with p closer to u than q.

- Create a vertex $v_{u,p}$ in $\hat{V}_d$ and an edge in $\hat{E}_d$ connecting u to v_{up}.
- If $q = v$ then create an edge in $\hat{E}_d$ connecting $v_{u,p}$ to v, else reset $W = W \cup q$
- Mark the edge (p, q).

end (while).

2. Unmark all marked edges

3. While W is not empty do begin

3.1 Remove a vertex w from W and reset $W := W \backslash w$.

3.2 Find a minimal weight legal path in $G(\overline{S}, d)$ connecting w to v.

3.3 If no such path exists, then unmark all the marked edges and go to 3.

3.4 If the path has weight 0 then create edges in $\hat{E}_d$ connecting all vertices v_{pq} in $\hat{V}_d$ such that $q = w$ to v, unmark all marked edges and go to 3.

3.5 Let (r', q') be the moral edge on the part found in step 3.2 closest to w with p' closer to w then q'.

- Create a vertex $v_{w,p'}$ in $\hat{V}_d$ and edges in $\hat{E}_d$ connecting all vertices $v_{p,q}$ in $\hat{V}_d$ such that $q = w$ to $v_{w,p'}$
- If $q' = v$ then create an edge in $\hat{E}_d$ connecting $v_{w,p'}$ to v, else reset $W = W \cup q'$.
- Mark the edge (p', q').

3.6 Go to 3.2

4. End of algorithm

3.2.2.2 An Example

The graph D shown below is borrowed from the paper of Verma and Pearl [11, 1991] and we want to marginalize this graph over $\{a, b, c, e, f\} = S$.

The annotated graph representation of D is shown in Fig. 4.

Notice that all ranges in $K(D)$ have nonempty intersection with S.

The graph $G_S(D)$ is shown in Fig. 5 where strong edges are represented by solid lines and weak edges by broken lines.

Notice the following:

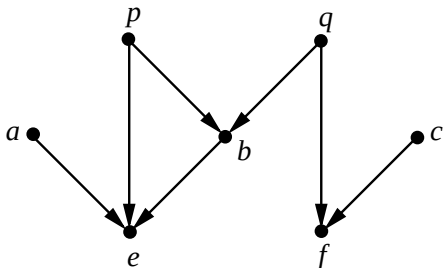


Figure 3: Graph D

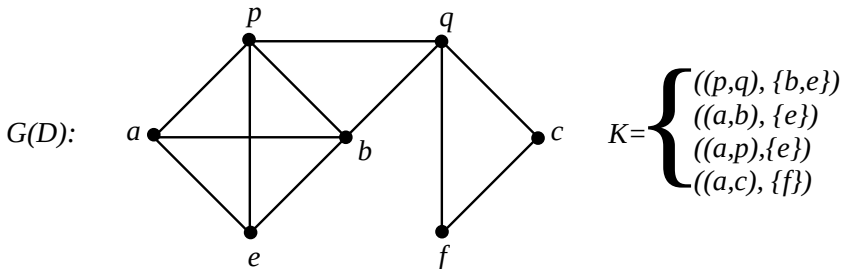


Figure 4: The annotated graph $A(D)$

- The edges $(a, e), (f, c)$ and (e, b) are strong edges due to criterion (a) of strong edges.
- The edge (e, f) is a strong edge due to the path e—p—q—f in $G(D)$ where the range of the moral edge p—q on the path includes the vertex e.
- The edges (b, f) is a strong edge due to the path b—q—f in $G(D)$.
- The edges (a, c) is a weak edge due to the path a—p—q—c in $G(D)$ etc.

The graphs $\hat{G}_d$ are shown in Fig. 6.

Notice that the moral edge (p, q) in $G(D)$ is treated as a regular edge in the construction of $\widehat{G_{(e,c)}}$, due to the fact that the range of (p, q) includes the vertex e (see step 1 of the construction algorithm for $\hat{G}_d$), and is not included therefore as a vertex in $\hat{G}_{(e,c)}$.

3.2.2.3 A property of the graphs $\hat{G}_d$

Definition 1 *Let (u, v) be a weak edge in $G_S(D)$. A moral edge (a, b) in $G(\overline{S}, d)$ will be called proper with regard to (u, v) iff the range corresponding to the domain (a, b) in $K(D)$ has empty intersection with $\{u, v\}$.*

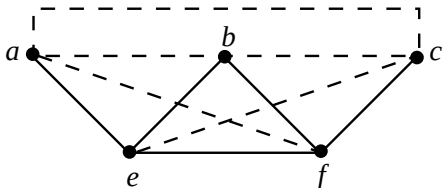


Figure 5: The graph $G_S(D)$

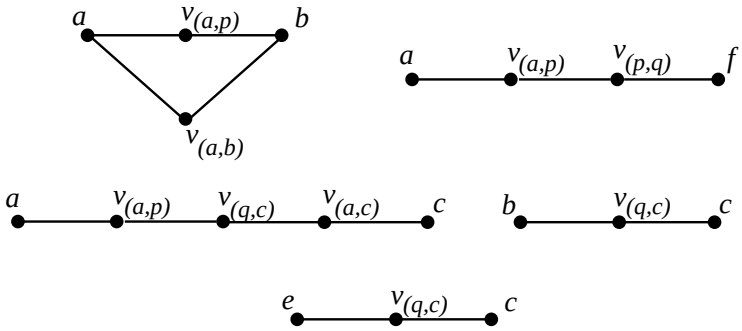


Figure 6: The graphs $\hat{G}_d$

Lemma 3 *Let (u, v) be a weak edge in $G_S(D)$ for a given marginalized DAG D over S. Then the following properties hold true:*

(i) There exists at least one legal path connecting u to v in $G(\overline{S}, d)$.

(ii) Every legal path as in (i) above includes at least one proper moral edge with regard to (u, v).

(iii) For every legal path π satisfying (i) and (ii) above there is a unique corresponding path π' in $\hat{G}_{(u,v)}$, connecting u to v, such that every vertex v_e on π' corresponds to a proper moral edge e on π and, if v_e precedes $v_{e'}$ on π' then e precedes e' on π.

Proof: By definition, since (u, v) is a weak edge, there is a path in $G(\overline{S}, d)$ connecting u to v. If the path is not legal then it contains forbidden transitions. Any such forbidden transition a—c—b corresponds to an uncoupled collider configuration $a \to c \leftarrow b$ in D. Therefore a—b is a moral edge on $G(D)$ which short circuits the vertex c on the path. Replacing the forbided transitions by the corresponding moral edges results in a legal path thus proving properties (i). To prove (ii), let π be a legal path connecting u to v in $G(\overline{S}, d)$. If π includes no moral edges then, as it excludes formiden transitions, it must correspond to a trail in D such that all the vetices on it (in D) are ancestors of u of v or both

implying that (u, v) is a strong edge, contrary to our assumption. On the other hand, if π includes moral edges but all of them are not proper with regard to (u, v) then we can show again that all the vertices on π are ancestors of either u or v thus rendering (u, v) a strong edge contrary to assumption. To show this let (a, b) and (a', b') be two consecutive nonproper moral edges on π. As the path is legal, the subpath from b to a' must have all its vertices ancestors of b or a' or both. Therefore, given that (a, b) and (a', b') are non proper we must have that b and a' are ancestors of u of v or both. It follows that all the vertices on the subpath from b to a' are ancestors of u or v or both. The same argument works when we consider the subpath from u to a, where (a, b) is the closest to u nonproper moral edge on π, and the subpath from b' to v where (a', b') is the closest to v nonproper moral edge on π. It follows that π must include some proper moral edges.

This argument also applies when there is only one nonproper moral edge on π. To prove (iii) let π be a path as in the proof of (ii) above. Consider the first while loop in the algorithm describing the construction of the graph $\hat{G}_{(u,v)}$ (step 1 part (4) section 3.2.2.1). Assume that this while loop generates a total of k vertices, say, $v_1, v_2, \ldots, v_k$ in $\hat{G}_{(u,v)}$ all adjacent to u in $\hat{G}_{(u,v)}$ and corresponding to the proper moral edges $e_1, \ldots, e_k$ in $G(\overline{S}, d)$. Then one of the edges $e_1, \ldots, e_k$ must be included in π. Otherwise, the existence of the path π connecting u to v and not including any of the proper moral edges $e_1, \ldots, e_k$ implies the existence of a minimum weight path from u to v not including $e_1, \ldots, e_k$, but including proper moral edges (by property (i)). This would have forced the while loop to execute an additional iteration which would have added to the set $v_1, \ldots, v_k$ an additional vertex, representing a proper moral edge on that minimal weight path. To complete the proof of property (iii) we can now use an inductive argument: Assume that we found a partial path in $\hat{G}_{u,v} : u, w_1, w_2, \ldots, w_j$ such that $w_1, \ldots, w_j$ correspond to proper moral edges on $\pi, e_1, \ldots, e_j$. If w_j is connected to v in $\hat{G}_{(u,v)}$ then we are done. Otherwise, step 3.5 in the construction algorithms shown in section 3.2.2.1 will generate a set of vertices, all connected to w_j in $\hat{G}_{(u,v)}$, corresponding to proper moral edges. Using a similar argument as the one used above we can show that one of those newly generated vertices corresponds to a proper moral edge on π subsequent to e_j on π. Continuing that way we can find the unique path π' in $\hat{G}_{u,v}$ having the property claimed in (iii). □

Corollary 1 *Let C be a vertex-cut set in $\hat{G}_{(u,v)}$ separating u from v. Then E_C the set of proper moral edges, corresponding to C in $G(\overline{S}, d)$ are an edge-cut separating u from v in $G(\overline{S}, d)$, over the legal paths (i.e., all legal paths are cut by the edge cut). Moreover, if E' is a subset of the set of proper moral edges which is a minimal edge cut over the legal paths in $G(\overline{S}, d)$ then all the edges in E' are represented as vertices in $\hat{G}_d$ and those vertices form a vertex-cutset in $\hat{G}_d$ between u and v.*

Proof: The first part of the corollary follows directly from property (iii) in the above lemma. To prove the second part we notice first that if e is a proper

moral edge in $G(\overline{S}, d)$ and e belongs to a minimal edge cut then e must generate a vertex in $\hat{G}_{(u,v)}$. To prove this we use the following argument. Since e belongs to a minimal edge cut over the legal paths there must be a minimal weight legal path including e. Following the construction algorithm in the previous section we must have that either e or another proper moral edge on the path, preceding e, generates a corresponding vertex in $\hat{G}_d$. If it is e then we are done, otherwise that preceding edge, say (a, b) will dictate the insertion of b into W and, when b is processed subsequently in step 3, we can use the same argument, for the legal subpath from b to v, that includes e as a proper moral edge on it, in order to show that either e or another proper moral edge on the subpath, generates a corresponding vertex on $\hat{G}_d$. Repeating this argument finitely many times we conclude that eventually e will generate a corresponding vertex in $\hat{G}_d$. Now let $C(E')$ be the set of vertices corresponding to E' in $\hat{G}_d$. If $C(E')$ is not a cutset between u and v in $\hat{G}_d$ then there is a path in $\hat{G}_d$ from u to v not passing through $C(E')$. This path corresponds to a legal path in $G(\overline{S}, d)$ not including any of the edges in E', which contradicts the assumption that E' is an edge cut over the legal paths in $G(\overline{S}, d)$. □

3.2.2.4 Deriving the graphs G_d out of the graphs $\hat{G}_d$

Let $\hat{G}_d$ be the intermediary graph constructed, as shown in section 3.2.2.1 for a given weak edge $d = (u, v)$ in $G_S(D)$. The vertices in $\hat{G}_d$, except u and v, correspond to proper moral edges in $G(\overline{S}, d)$ and those moral edges are the domains of some elements in $K(D)$. To derive the graph G_d out of $\hat{G}_d$ we follow the steps below:

1. For every vertex v_e, except u and v, in $\hat{G}_d$ find the range of the element in $K(D)$ whose domain is e and denote it by $r(e)$.
2. Find $r(e) \cap S$ and denote $r_S(e) = r(e) \cap S$
3. For every vertex v_e in $\hat{G}_d$ substitute the set of vertices $r_S(e)$.
4. If v_e is adjacent to $v_{e'}$ in $\hat{G}_d$ then connect all the vertices in $r_S(e)$ to all the vertices in $r_S(e')$. Also connect u to all the vertices in $r_S(e)$ if v_e is adjacent to u and do the same for v.
5. By construction, all the vertices in any set $r_S(e)$, as above, are in $G_S(D)$. If some vertex w is included in several sets $r_S(e)$ for different e's then merge all those occurrences of w into a single vertex and connect this vertex to all the vertices that were adjacent to some occurrence of w before the merger.
6. Denote the resulting graph by G_d and set this graph as the annotation of the weak edge $d = (u, v)$ in $G_S(D)$.

Remark 7 *The following relation exists between $\hat{G}_d$ and G_d where $d = (u, v)$. If C is a cutset between u and v, in G_d, then there is a cutset C' between u and*

v in $\hat{G}_d$ such that the following inclusion holds:

$$c'' = \{\bigcup r_S(e) : v_e \in C'\} \subseteq C$$

This follows from the fact that, by step 4, if v_e is adjacent to $v_{e'}$ in $\hat{G}_d$ then all vertices in $r_S(e)$ are connected to all vertices in $r_S(e')$ and therefore if some vertex $v_1 \in r_S(e_1)$ is connected to some vertex $v_2 \in r_S(e_2)$ through some vertex $v_3 \in r_S(e_3)$ in G_d then all the vertices in $r_S(e_3)$ must belong to a cutset in G_d separating v_1 from v_2.

3.2.2.5 Definitions of $K_S(D)$ and $A_S(D)$

An element $(d, r(d))$ will be called degraded if it's domain d is empty. For every element in $K(D)$ construct a degraded element $(\emptyset, r_S(d))$ where $r_S(d) = r(d) \cap S$. As mentioned in section 3 we assume that for all elements $(d, r(d))$ in $K(D), r(d) \cap S \neq \emptyset$. The set of elements $K_S(D)$ is now defined as below:

$$\begin{aligned} K_S(D) &= \{(\emptyset, r_S(d)) : (d, r(d)) \in K(D)\} \bigcup \\ &\quad \{(d, G_d) : d \text{ is a weak edge in } G_S(D)\} \end{aligned}$$

Finally we define

$$A_S(D) = (G_S(D), K_S(D)).$$

$A_S(D)$ is the annotated graph representation of a given DAG D marginalized over a subset S of it's vertices.

3.2.2.6 An example (continued)

Consider again the example in section 3.2.2.2. The annotated graph representation of the DAG D in Fig. 3 is shown in Fig. 7. The graph $G_S(D)$ is shown in Fig. 8 where $S = \{a, b, c, e, f\}$. Those graphs are reproduced here for the benefit of the reader:

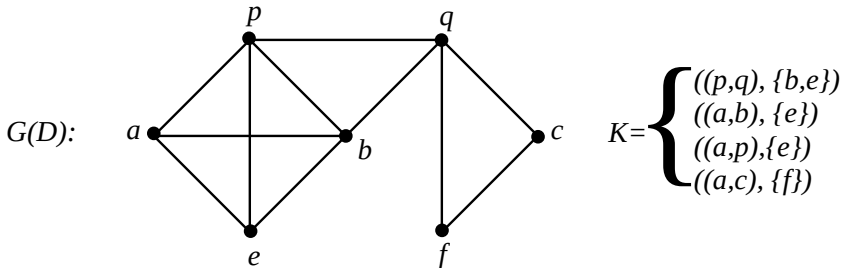


Figure 7:

The derivation of the graphs G_d are shown below:
In the same way we get for the other weak edges in$G_S(D)$.

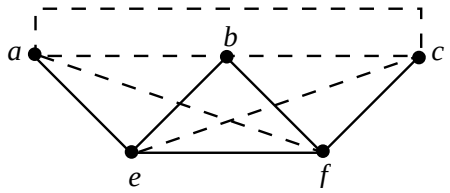
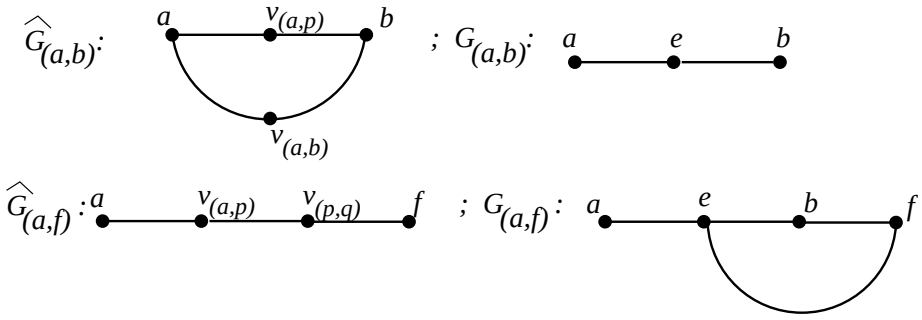


Figure 8:



The degraded elements are found to be:

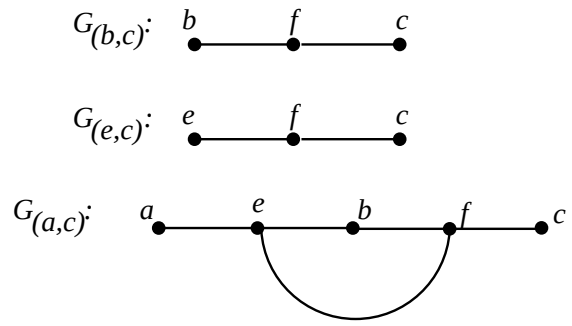
$$(\emptyset, \{b, e\}), (\emptyset, \{e\}), (\emptyset, \{f\})$$

The set $K_S(D)$ therefore includes all the above degraded elements and all the elements of the form (d, G_d) where $d = (a, b), (a, f), (a, c), (b, c), (e, c)$.

3.2.2.7 A property of the annotated graph representation of marginalized DAG's

Lemma 4 *Let $A(D) = (G(D), K(D))$ be the annotated graph representation of a given DAG D. Let $A_S(D) = (G_S(D), K_S(D))$ be the annotated graph representation of the marginalized relation represented by D over a subset S of the vertices of $G(D)$. Let $t = (X; Y|Z)$ be a triplet such that $v(t) = \{X \cup Y \cup Z\}$ is a subset of S. Let $G'(D)$ be the graph derived from $G(D)$ when the L2 Algorithm is applied to $A(D)$ for the triplet t and let (d, G_d) be a nondegraded element in $K_S(D)$ where $G_d = (V_d, E_d)$ and $d = (u, v)$. Then all the paths in $G'(\overline{S}, d)$ between u and v are disconnected if and only if $(S \backslash v(t)) \cap (V'_d \backslash \{u, v\})$ is a cutset between u and v in G_d.*

Proof: We show first that if and only if all legal paths in $G'(\overline{S}, d)$ are disconnected then all paths are disconnected. Trivially, we need to prove the "if" case only. To this end, let π be a path in $G(\overline{S}, d)$ connecting u to v and assume that π is not legal. Then it must include forbidden transitions. Any such transition a—c—b corresponds to an uncoupled collider $a \rightarrow c \leftarrow b$ in D. Therefore a—b is



a moral edge in $G(D)$ corresponding to an element in $K(D)$ with range $r(a,b)$, and one can construct a legal path π' in $G(\overline{S},d)$ short circuiting all forbidden transitions by their corresponding moral edges. As shown in the proof of part (ii) of lemma 3, at least one of the moral edges on π', short circuiting the forbidden transitions on π must be proper. Recall that the range $r(a,b)$ corresponding to such a proper moral edge does not intersect $\{u,v\}$. Now if all legal paths between u and v in $G(\overline{S},d)$ are disconnected by the L2 Algorithm then the path π' is disconnected which means that a proper moral edge on π' is disconnected. Let (a',b') be the proper moral edge on π' disconnected by the L2 Algorithm. The condition for this to happen is that $r(a',b')\cap v(t)=\emptyset$. But if this condition holds then the L2 Algorithm will also remove $r(a',b')$ from $G(D)$ thus disconnecting π. To complete the proof of the lemma it suffices therefore to show, based on the above argument, that if and only if the conditions of the lemma hold then all legal paths in $G'(\overline{S},d)$ between u and v are disconnected. We proceed now to prove this claim. Assume first that all legal paths between u and v in $G(\overline{S},d)$ are disconnected when the L2 Algorithm is applied to $A(D)$ for the triplet t. Then there is a set E_C of proper moral edges in $G(\overline{S},d)$, removed by the L2 Algorithm, which is a minimal edge-cut over the legal paths between u and v in $G(\overline{S},d)$. By lemma 3 this set of edges correspond to a cutset C between u and v in $\hat{G}_d$ and by the construction of G_d the set of vertices C generates a set of vertices C' defined as below.

$$C'=\{\bigcup r_S(a,b) : w_{ab}\in\hat{G}_d,(a,b)\in E_C\}$$

and C' is a cutset between u and v in G_d, with $C'\subseteq S$. Now the condition for the removal of the set of edges E_C by the L2 Algorithm holds if and only if $r(a,b)\subseteq V(D)\backslash v(t)$ for all the edges $(a,b)\in E_C$. Also as $v(t)$ is a subset of S the above condition implies that $r_S(a,b)\subseteq S\backslash v(t)$ for all $(a,b)\in E_C$ so that $C'\subseteq S\backslash v(t)$. Furthermore, as C' is a cutset between u and v in G_d with $C'\subseteq V_d$ we have that $(S\backslash v(t))\cap V_d$ is a cutset between u and v in G_d as required. Assume now that $(S\backslash v(t))\cap(V_d\backslash\{u,v\})=C''$ is a cutset between u and v in G_d. Then, by remark 6, there must be a cutset $\overline{C}$ in $\hat{G}_d$ between u and v such that for every vertex v_e in $\overline{C}$, $r_S(e)\subseteq C''$. By lemma 3 the cutset $\overline{C}$ in $\hat{G}_d$ corresponds to an edge cut $E_{\overline{C}}$ over the legal paths in $G(\overline{S},d)$. Let e be an

edge in $E_{\overline{C}}$. Then e corresponds to a vertex v_e in $\overline{C}$ and, as mentioned above $r_S(e) \subseteq S \backslash v(t)$. Now $v(t) \subseteq S$ and therefore $r(e) \subseteq V \backslash v(t)$ which implies that the condition for the removal of e from $G(\overline{S}, d)$ exists for all $e \in E_{\overline{C}}$ which is an edge cut over the legal paths between u and v in $G(\overline{S}, d)$. □

3.3 A semantics for defining the relation represented by the annotated graph derived from a marginalized DAG

3.3.1 Algorithm MDT

Let $A_S(D) = (G_S(D), K_S(D))$ be the annotated graph representation of a DAG D marginalized over a subset S of it's vertices. Let $t = (X; Y|Z)$ be a triplet over S and let $v(t)$ be the set of vertices $v(t) = X \cup Y \cup Z$. To ascertain whether t is represented in $A_S(D)$ apply the procedure below:

1. For every degraded element $(\emptyset, r_S(d))$ in $K_S(D)$, if $r_S(d) \cap v(t) = \emptyset$ then remove the vertices in $r_S(d)$ and incident edges from $G_S(D)$.

2. For every nondegraded element (d, G_d) in $K_S(D)$ where $d = (u, v)$. Iff $(S \backslash v(t)) \cap (V_d \backslash \{u, v\})$ is a cutset between u and v in G_d then remove the edge (u, v) from $G_S(D)$.

3. t is represented in $A_S(D)$ if and only if it is represented in the UG generated from $G_S(D)$ by steps 1 and 2 above.

3.3.2 An example (continued)

Consider again the example in section 3.2.2.2. For the given subset of variables $S = \{a, b, c, e, f\}$ the annotated graph $A_S(D) = (G_S(D), K_S(D))$ is shown in section 3.2.2.6 and the graph $G_S(D)$ is reproduced below:

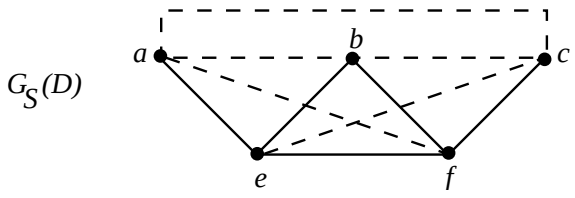


Figure 9: The graph $G_S(D)$

Let t be the triplet $t = (b, c; a|f)$, then $v(t) = \{b, c, a, f\}$. The degraded element $(\emptyset, \{e\})$ satisfies the property in step 1, $\{e\} \cap v(t) = \emptyset$, therefore the vertex e should be removed from $G_S(D)$. The condition in step 2 is satisfied for $G_{(a,b)}, G_{(a,f)}$ and $G_{(a,c)}$ therefore the weak edges $(a, b), (a, f)$ and (a, c) should be removed from $G_S(D)$. The resulting UG is shown below

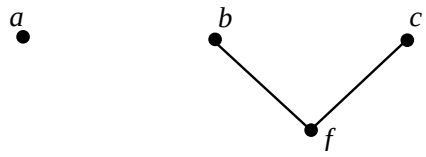


Figure 10: Test Graph

Verifying that t is represented in the test graph we conclude that t is represented in $A_S(D)$.

3.3.3 Correctness – Main Theorem

The testing procedure described in section 3.3.1 will be referred to as the MDT (marginalized DAG test) procedure.

Theorem 1 *Let $A_S(D) = (G_S(D), K_S(D))$ be the annotated graph representation of a DAG D marginalized over a subset S of it's vertices. Let $t = (X; Y|Z)$ be a triplet over S. Then t is represented in D if and only if it is represented in $A_S(D)$ through the MDT procedure.*

Proof: We know that t is represented in D if and only if it is represented in $A(D) = (G(D); K(D))$ through the L2 Algorithm. Denote by G_1 the undirected graph derived from $G(D)$ by the L2 Algorithm for $A(D)$ and t. Then t is represented in $A(D)$ if and only if it is represented in G_1. Let G_2' be the subgraph of G_1 over the vertices $V_1 \cap S$. For any pair of nonadjacent vertices (a, b) in G_2' such that there is a path in G, connecting a to b through $V_1 \backslash V_2$, add the edge (a, b) to E_2'. Denote the resulting graph by G_2. It is easy to show and it is well known that t is represented in G_2 if and only if it is represented in G_1 (recall that $v(t) \subseteq S$). To prove the theorem , it suffices therefore to prove that G_2 is equal to the graph generated from $G_S(D)$ by the MDT procedure. Denote by $G_S(D, t) = (V_S(D, t), E_S(D, t))$ the graph generated from $G_S(D)$ by the MDT procedure for the triplet t. We will show that $V_S(D, t) = V_2$ and $E_S(D, t) = E_2$. Clearly, the vertices in V_2 are the vertices in S except vertices which have been removed from S by the L2 Algorithm. The removed vertices must belong to the range of an element $(d, r(d))$ in $K_S(D)$ and by step 1 of the MDS procedure the vertices in $r_S(d)$ are removed from $G_S(D)$ since $r(d) \cap v(t) = \emptyset$ implies that $r_S(d) \cap v(t) = \emptyset$. Therefore $V_S(D, t) = V_2$. Consider now $E_S(D, t)$. Let $d = (a, b)$ be a strong edge in $G_S(D)$. Then it is an edge also in $G_S(D, t)$. It could be a strong edge in $G_S(D)$ for one of two reasons: (i) (a, b) is an arc in D. Then it is an edge in both E_2 and $G_S(D, t)$. (ii) It is not an arc in D but there is a path in $G(\overline{S}, d)$ such that all the vertices on the path are ancestors of $\{a, b\}$ and the range of every moral edge on the path (if any) has nonempty intersection with $\{a, b\}$. Then this path is not removed when G_1 is created by the L2 Algorithm, since the vertices on the path cannot belong

to a range to be removed by the L2 Algorithm and the element corresponding to moral edges on the path, if any, are not processed by the L2 Algorithm preventing the removal of such moral edges. Therefore (a, b) will be an edge in both E_2 and $E_S(D)$. By step 2 of the MDT procedure d is not an edge in $E_S(D, t)$ if and only if $(S \backslash v(t)) \cap (V_d \backslash \{u, v\})$ is a cutset between u and v in G_d. By lemma 4 this holds true if and only if all paths in $G(\overline{S}, d)$ between u and v are disconnected by the L2 Algorithm. Thus (a, b) is not set as an edge in E_2 when G_2 is created iff it is not an edge in $E_S(D, t)$. We have thus shown that $E_S(D, t) \subseteq E_2$. To show that $E_2 \subseteq E_S(D, t)$ we notice that the edges (a, b) in E_2 are either corresponding to arcs in D, or are created due to the fact that a path in $G(\overline{S}, d)$ exists connecting a to b which corresponds to the condition of a strong edge in $G_S(D)$, or are created due to the fact that a path exists in $G(\overline{S}, d)$ connecting a to b which corresponds to a weak edge in $G_S(D)$, which is not removed by the MDT procedure. □

4 Extensions and Final Comments

4.1 PD-induced relations that cannot be represented by a marginalized DAG

While DAG's are widely used as a model that can represent PD-induced relations one may ask whether it might be possible to represent every PD-induced relation either by a DAG or, assuming the existence of latent variables, by a marginalized DAG. The answer to this question is negative as should be expected. A counterexample is given below.

Consider the following PD-induced relation, over 3 variable x, y, and z, consisting of two triplets only:

$$R = \{(x; y|\emptyset), (x; y|z) + \text{symmetric triplets}\}$$

Then R cannot be represented by a marginalized DAG. To prove this claim assume that there is a DAG D with n variables, including x, y and z such that when D is marginalized over $\{x, y, z\}$, the marginalized DAG represents R. This assumption leads to a contradiction: Since $(x; z|\emptyset)$ and $(y; z|\emptyset)$ are not in R there must be trails π_{xz} and π_{yz} in D with no colliders included in them. Let π_{xy} be the concatenation of the two trails π_{xz} and π_{zy} (which is the trail π_{yz} reversed). Then π_{xy} connects between x and y and has no colliders on it except perhaps the vertex z. If z is a collider then $(x; y|z)$ is not represented in D. If z is not a collider then π_{xy} has no colliders on it and therefore $(x; y|\emptyset)$ is not represented in D. Therefore R cannot be represented by marginalizing D over $\{x, y, z\}$, a contradiction. That R is a PD-induced relation was shown by Milan Studeny [10, private communication, 2000] as follows:

Consider the PD over the binary variables x, y and the ternary variable z. The probability of the three variables for the different values of x, y, z is given

below

$$\begin{array}{ll} p(0,0,0) & = p(0,0,1) = p(1,0,1) = p(1,0,2) = \frac{1}{8} \\ p(0,1,0) & = p(1,1,1) = \frac{1}{4} \\ p(x,y,z) & = 0 \text{ for all other configurations} \end{array}$$

The reader can convince himself that the relation induced by the above PD is the relation $R = \{(x; y|\emptyset), (x; y|z)\}$. Notice however that the relation R above is represented by the annotated graph below

$$G : x—z—y \qquad K = \{((x, y), \{z\})\}$$

see Paz [5, 2003b].

4.2 Space complexity considerations

The number of elements in the set $K(D)$ of the annotated graph representation of a given DAG D is bounded by the number of moral edges in $G(D)$. For every element $(d, r(d))$ in $K(D)$ the range $r(d)$ is a unique subset of the vertices of D, whose removal from $G(D)$ when a triplet t such that $v(t) \cap v(d) = \emptyset$ is tested by the L2 Algorithm, induces the removal of the edge d from $G(D)$. The situation is quite different when we apply the MDT procedure to the annotated graph representation of a marginalized DAG. The space required for storing $A_S(D)$ is still polynomial in the number of vertices of D: the number of degraded elements is bounded by the number of moral edges in D, the number of elements (d, G_d) is equal to the number of weak-edges in $G_S(D)$ and the set of vertices in G_d, for every weak edge d, is a subset of the set of vertices of $G_S(D)$. The time complexity for testing a triplet t by the MDT procedure is also polynomial as is easy to see. On the other hand the condition for the removal of a weak edge d from $G_S(D)$ dictated by the MDT algorithm is a complex condition i.e., d should be removed if $(S \backslash v(t)) \cap (V_d \backslash \{u, v\})$ is a cutset between u and v in G_d, where $d = (u, v)$. Now the number of such cutsets could be exponential so that there may be exponentially many different conditions whose presence dictates the disconnecting of u from v when the MDT procedure is applied to $A_S(D)$ for different triplets t.

This feature is illustrated in the example below – introduced in Geiger et al. [1, 1994]. Consider the DAG $D = (V, E)$ with $V = \{x_i | i \in N\} \cup \{y_i | i \in N\} \cup \{u_{ij} | i, j \in N, i \neq j\} \cup \{v_{ij} | i, j \in N, i \neq j\} \cup \{w_{ii} | i \in N\}$

$$\begin{array}{rlll} E = & \{x_i \leftarrow u_{ij} \rightarrow x_j & | & i, j \in N, i \neq j\} \cup \\ & \{y_i \leftarrow v_{ij} \rightarrow y_j & | & i, j \in N, i \neq j\} \cup \\ & \{x_i \leftarrow w_{ij} \rightarrow y_i & | & i \in N\} \end{array}$$

An ij face of D is shown in Fig. 11:

Let $A(D) = (G(D), K(D))$ be the annotated graph representation of D. An i, j face of $G(D)$ is shown in Fig. 12.

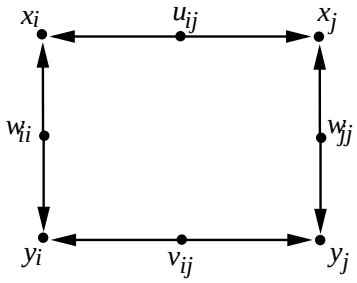


Figure 11: i, j face of D

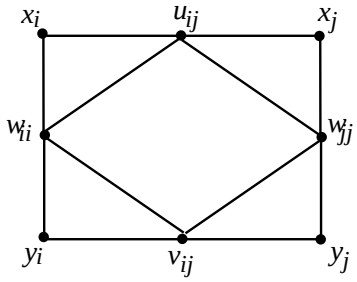


Figure 12: i, j face of $G(D)$

The set of elements $K(D)$ is:

$$K(D) = \{(u_{ij}, w_{ii})\{x_i\})|i, j \in N, i \neq j\} \cup \{(v_{ij}, w_{ii})\{y_i\})|i, j \in N, i \neq j\}$$

Assume now that we want to marginalize D over $S = \{x_i|i \in N\} \cup \{y_i|i \in N\}$. An i, j face of $G_S(D)$ is shown in Fig. 13.

It is easy to verify that the edges $(x_i, x_j), (y_i, y_j), (x_i, y_i)$ and (x_j, y_j) are strong edges while (x_i, y_j) and (x_j, y_i) are weak edges.

Let $d = (i, j)$ for any i and j. In order to construct $G_{(i,j)}$ we observe the following. There are paths connecting x_i to y_i through the removed vertices which can be found in Fig. 12:

$$x_i — u_{ij} — w_{jj} — y_j \text{ and } x_i — w_{ii} — v_{ij} — y_j$$

The condition for disconnecting those paths (consult the MSD procedure) is that both x_i and y_j are not in $v(t)$. But there are exponentially many additional such paths even if both y_i and x_i are not in $v(t)$, illustrated in Fig. 14 where $k \neq i, j$,

Those paths are, fore every $k \neq i, j$:

$$x_i — u_{ik} — w_{kk} — v_{kj} — y_j$$

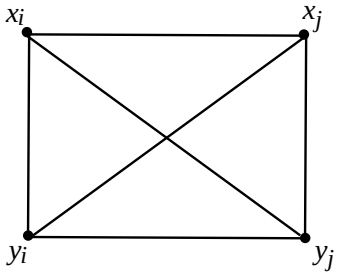


Figure 13: i, j face of $G_S(D)$

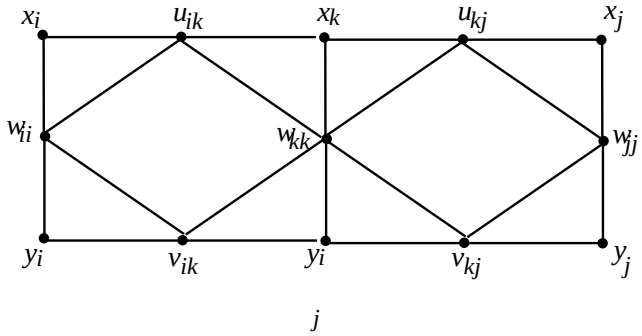


Figure 14: Paths connecting x_i to y_j

The condition for disconnecting those paths is that, for every k, either x_k or y_k is not in $v(t)$. Therefore G_{ij} is as shown in Fig. 15. i.e., $V_{ij} = \{x_i | i \in N\} \cup \{y_i | i \in N\}$ and $E_{ij} = \{x_i, x_j), (x_i, y_i), (x_j, y_j), (y_i, y_j)\} \cup \{(x_i, x_k), (x_k, y_k), (y_k, y_j) | k \neq i, j\}$.

Notice that the number of possible cutsets in this graph is exponential since any subset of vertices including x_j, y_i and at least one of $\{x_k, y_k\}$ for each $k \neq i, j$ is a cutset.

4.3 Marginalized DAG's that can be represented by DAG's

Based on the previous sections it is clear that marginalized DAG's are not always representable by DAG's. It was shown in [5, 2003b] that the annotated graph representation of DAG's is unique. Moreover, the set of elements $K(D)$ in the representation of a DAG D does not contain degraded elements and all the elements in $K(D)$ have the form $(d, r(d))$, where $r(d)$ is a set of vertices whose removal from $G(D)$ (given that $r(d) \cap v(t) = \emptyset$ for a tested triplet) enables the disconnection of the edge corresponding to d in $G(D)$, when the L2 Algorithm

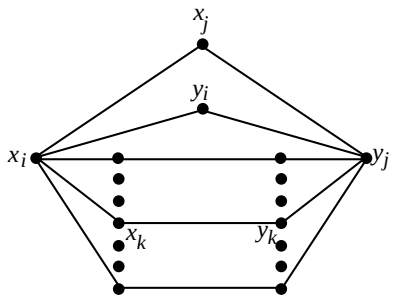


Figure 15: Graph G_{ij}

is applied to $A(D)$, for a triplet t. This feature induces a necessary condition for $A_S(D)$ to be represented by a DAG, namely, $K_S(D)$ must not include degraded elements and all the nondegraded elements (d, G_d) must have the property that there is a unique and simple set of vertices in $V_d \backslash \{u, v\}$ which separates u from v in G_d where $d = (u, v)$. If this necessary condition is satisfied then, based on the MDT procedure, we can reset $K_S(D)$ so that all the elements in this set have the form $(d, r(d))$ where $r(d)$ is equal to the unique cutset separating u from v in G_d, and then apply a polynomial procedure to $A_S(D)$, described in [5, 2003b] which will find a DAG D representing $A_S(D)$, if such exists, or will declare that $A_S(D)$ is not representable by a DAG (even though it satisfies the above necessary condition). In fact, the above necessary condition can be simplified. We will show now that if all the nondegraded elements in $K_S(D), (d, G_d)$, have the property that there is a unique cutset separating u from v in G_d, then all the degraded elements are superfluous and can be discarded. An element that has this property (of unique cutset) will be called simple element.

Claim 1 *If all the nondegraded elements in $K_S(D)$ are simple, then the annoted graphs $A_S(D) = (G_S(D), K_S(D))$ and $A'_S(D) = (G_S(D), K'_S(D))$ are equivalent, where $K'_S(D)$ includes the nondegraded elements of $K_S(D)$ only.*

Proof: let t be any triplet over S. If t is represented in $A'_S(D)$, through the MDT procedure then t is represented in $A_S(D)$ since the degraded elements many induce the removal of additional vertices, disjoint of $v(t)$ and this cannot change the fact that t is represented in the UG generated by the MDT procedure.

To complete the proof assume that t is represented in $A_S(D)$ but it is not represented in $A'_S(D)$. Let $t = (X; Y|Z)$ and let G_t and G'_t be the UG's derived from $G_S(D)$ by the MDT procedure, for testing t, from $A_S(D)$ and $A'_S(D)$ correspondingly. V'_t includes some vertices belonging to degraded elements disjoint of $v(t)$ which are not included in V_t. As t is not represented in $A'_S(D)$ it is not represented in G'_t and therefore there must be a path π from some vertex $x \in X$ to some vertex $y \in Y$ such that all the vertices of a subpath of π are in $V'_t \backslash V_t$. Let $\pi = x, v_1, \ldots, v_k, y$ be such a path. Let π' be the trail corresponding to π in D And let $v_i, \ldots, v_j$ be the subpath of π' in $V'_t \backslash V_t$. Now $V_t \backslash V'_t$ includes only

vertices from degraded elements which, as such belong to ranges of elements in $K(D)$. Therefore we must have that, in D, v_{i-1} is oriented into v_i and v_{j+1} is oriented into v_j where v_{i-1} and v_{j+1} are in V_t, since otherwise v_{i-1} or v_{j+1} or both would also belong to $V_t'\backslash V_t$, contrary to the choice of i and j, this following from the fact that $v_i \rightarrow v_{i-1}$ implies that v_{i-1} is in the same range as v_i and similarly for $v_j \rightarrow v_{j+1}$ (recall that $v_i, \ldots, v_j$ belong to the intersection of some ranges in $K(D)$ with S). This implies that in D, a collider must exist on the subpath $v_i, \ldots, v_j$. If the collider is coupled then we can short circuit the subpath by the arc joining the parents of that collider. Continuing the short circuiting process several times we will get, eventually, a trail connecting v_{i-1} to v_{i+1}, including an uncoupled collider, which is a vertex on the subpath $v_i, \ldots, v_j$.

This implies that a nondegraded element would have been included in both $A_S(D)$ and $A'_S(D)$ which would have been processed by the MDT algorithm, disconnecting π (notice that all the successors of the collider must belong to $V_t'\backslash V_t$ since successors of vertices in some range must belong to the same range). This argument holds for any path connecting x to y through $V_t'\backslash V_t$ and therefore, if t is represented in $A_S(D)$ then it is represented in $A'_S(D)$ thus completing the proof. □

It follows from the above claim that if all the elements in $K_S(D)$ are simple, where $A_S(D)$ is the annotated graph representation of a marginalized DAG, then we can discard all the degraded element from $K_S(D)$. Moreover every simple element (d, G_d) can be replaced by a regular element $(d, r(d))$ where $r(d)$ is the unique cutset disconnecting u from v in G_d, since the condition for disconnecting u from v is that $r(d)$, the unique cutset in G_d, is disjoint of t.

Summing up: If $A_S(D) = (G_S(D), K_S(D))$ is such that all the nondegraded elements in $K_S(D)$ are simple then $K_S(D)$ can be replaced by a regular annotation and then the procedure described in [5, 2003b] can be applied to check whether $A_S(D)$ can be represented by a DAG. This is illustrated in the example below.

4.4 An Example

Let D be the graph shown in Fig. 16

The annotated graph representation of D is shown in Fig. 17.

Marginalizing over $S = \{a, b, c, d\}$ we get the annotated graph shown in Fig. 18.

The degraded element can be discarded. An equivalent DAG can be found and is shown in Fig. 19.

4.5 Checking whether two DAG's are equivalent when both are marginalized over the same subset of their common vertices

This problem was considered in [9, 1992] where a polynomial algorithm is suggested for solving it. It is reasonable to assume that the annotated graph rep-

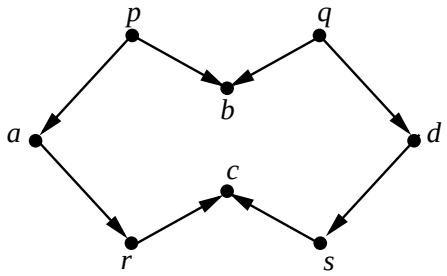


Figure 16: DAG representing relation

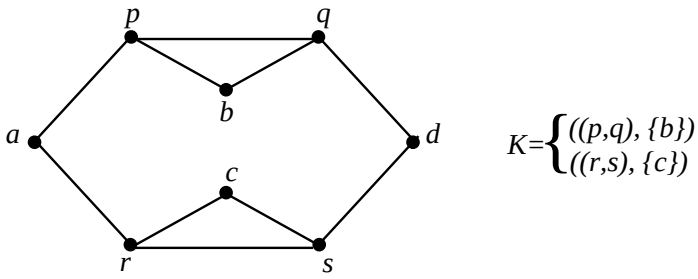


Figure 17: DAG representing relation

resentation may also provide a way for solving the above problem. The tools developed so far may however not be enough for this use. It may be necessary to provide first a means of simplifying the annotated graph representation of marginalized DAG's so as to get some kind of canonical annotated graph which is unique. So far it is not clear and it is probably not true that the annotated graph representing a marginalized DAG as developed in the previous sections is unique and it is reasonable to assume that some simplifications of the representation is possible, by eliminating superfluous degraded elements and by simplifying the G_d graphs. This subject is not pursued further in the work.

4.6 Conditioning

Given a GAG A, the procedure below will derive from A another GAG A' which represents the relation $R(A)$ when conditioned for a subset T of its variables.

Procedure C

Input: a GAG $A = (G, K)$ with

$$K = \{(\emptyset, r_i) : i = i, \ldots, j\} \cup \tag{1}$$

$$\{(d, G_d) : d \in E\} \tag{2}$$

and a subset $T \subset V$ where $G = (V, E)$.

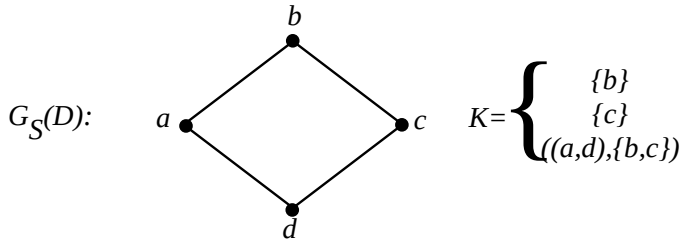


Figure 18: DAG representing relation

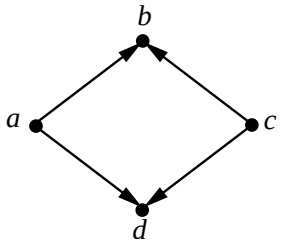


Figure 19: DAG representing relation

1. For every element in $K(\emptyset, r_i)$ such that $r_i \cap T \neq \emptyset$, remove $(\emptyset, r_i)$ from K

2. For every element in K (d, G_d) such that $d \cap T \neq \emptyset$, remove (d, G_d) from K.

3. Remove T and incident edges from G.

4. For any element (d, G_d) not removed in step 2, such that $V_d \cap T \neq \emptyset$ do: For any vertex $v \in V_d \cap T$, remove v from V_d and connect by an edge any two vertices u and w such that both u and w are adjacent to v in G_d.

   Set $A' = (G', K')$ where G' is the graph derived in step 3 and K' is the annotation derived from K after completion of steps 1,2, and 4.

- End of procedure
  The fact that the procedure is correct follows directly from the MDT algorithm and is left to the reader.

Acknowledgment

The author is indebted to Judea Pearl for useful hints and comments.

References

[1] D. Geiger, A. Paz, and J. Pearl. On testing whether an embedded Bayesian network represents a probability model. In R. Lopez de Mantaras and D. Poole, editors, *Uncertainty in Artificial Intelligence 10*, pages 244–252. Morgan Kaufmann, San Mateo, CA, 1994.

[2] S .L. Lauritzen, A. P. Dawid, B. N. Larsen, and H. G. Leimer. Independence properties of directed Markov fields. *Networks*, 20:491–505, 1990.

[3] S.L. Lauritzen. *Graphical Models*. Claredon, Oxford, U.K., 1996.

[4] A. Paz. An alternative version of Lauritzen et al's algorithm for checking representation of independencies. *Journal of Soft Computing*, pages 491–505, 2003a.

[5] A. Paz. The annotated graph model for representing DAG-representable relations – algorithmic approach. Technical Report Technical Report R-312, Computer Science Department, UCLA, 2003b.

[6] A. Paz, R.Y. Geva, and M. Studeny. Representation of irrelevance relations by annotated graphs. *Fundamenta Informaticae*, 48:149–199, 2000.

[7] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA, 1988.

[8] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, 2000.

[9] P. Spirtes and T. Verma. Equivalence of causal models with latent variables. Technical Report CMU-PHIL-33, Carnegie Mellon University, Pittsburgh, Pennsylvania, October 1992.

[10] M. Studeny, 2000. Private communication.

[11] T. Verma and J. Pearl. Equivalence and synthesis of causal models. In L.N. Kanal P. Bonissone, M. Henrion and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 6*, pages 225–268, B.V., 1991. Elsevier Science Publishers.

[12] N. Wermuth and D.R. Cox. A sweep operator for triangular matrices and its statistical applications. Technical Report Research Report, ZUMA 00-04, Universitat Mainz, Center of Survey Research, F.R. Germany, 2000.