Equivalence of Causal Models with Latent Variables

by

Peter Spirtes and Thomas Verma

October 1992

Report CMU-PHIL-33

Carnegie Mellon

Philosophy
Methodology
Logic

Pittsburgh, Pennsylvania 15213-3890

Equivalence of Causal Models with Latent Variables

by Peter Spirtes and Thomas Verma

Report CMU-Phil-33

1 Introduction

In this paper, we will investigate when it is impossible to determine which of two causal structures generated a given probability distribution, given only the set of conditional independence and dependence relations true of the distribution.

Causal relations among a set of random variables **V** are represented by a directed acyclic graph over **V**, where there is an edge from A to B if and only if A is an immediate cause of B relative to **V**. If there is a directed path from A to B in the causal graph, we will say that A is a (possibly indirect) cause of B. (In what follows, we will capitalize random variables, and boldface any sets of variables. We will use the terms "vertices in a graph" and "variables in a graph" interchangeably.)

A directed acyclic graph G over a set of random variables **V** can also be used to represent the set of probability distributions over **V** that satisfy the following two conditions:

Markov Condition: Let **Parents**(X) be the set of parents of X in G (i.e. the set of Z such that Z -> X is in G) and **Descendants**(X) be the set of descendants of X in a graph G (i.e. the set of Z such that there is a directed path from X to Z in G.) A directed acyclic graph G and a probability distribution P on the vertices **V** of G satisfy the Markov condition if and only if for every X in **V**, X and **V**\({X} ∪ **Descendants**(X)) are independent conditional on **Parents**(X).

Faithfulness Condition: If G is a directed acyclic graph G and P is a distribution over the set of vertices **V** in G, then P is faithful to G if and only if every conditional independence relation true in P is entailed by the Markov condition for G.

Suppose we make the assumption that the probability distribution generated by a causal structure satisfies the Markov and Faithfulness Conditions for the causal structure. (For a justification of this assumption see Spirtes, Glymour, and Scheines forthcoming.) Then there are a number of algorithms (Pearl and Verma 1990, Spirtes, Glymour and Scheines 1990, Spirtes, Glymour, and Scheines 1990, Spirtes, Glymour and Scheines, forthcoming) that can find the set of causal structures compatible with the observed conditional independence relations in a distribution. Under these assumptions, what are the limitations of any method of causal inference that uses only conditional independence and dependence relations to construct a set of causal structures? The answer to this question is that no such method can distinguish between any two causal graphs G_1 and G_2 if satisfying the Markov and Faithfulness Conditions for G_1 entails the observed conditional dependencies and independencies, and satisfying the Markov and Faithfulness Conditions for G_2 entails the observed conditional dependencies and independencies; let us call this condition **faithful entailment equivalence**. Faithful entailment equivalence between directed acyclic graphs is equivalent to a number of other interesting relationships between graphs including: **strong Markov equivalence** (the set of distributions satisfying the Markov Condition for G_1 equals the set of distributions satisfying the Markov Condition for G_2), **strong faithful equivalence** (the set of distributions satisfying the Markov and Faithfulness Conditions for G_1 equals the set of distributions satisfying the Markov and Faithfulness Conditions for G_2), and **weak faithful equivalence** (there exist distributions satisfying the Markov and Faithfulness Conditions for G_1 and G_2.)

There is a way to characterize when two directed acyclic graphs satisfy faithful entailment equivalence that follows from results proved in Verma and Pearl (1990). (See Spirtes, Glymour, and Scheines forthcoming.) One way of phrasing the answer is this. A pattern is a graph which contains a mixture of directed and undirected edges. Let a pattern for a directed acyclic graph G be a graph with the same vertices as G, the same adjacencies as G, and each edge A - B is undirected except if there is an undirected subgraph $A \rightarrow B \leftarrow C$ of G, where A and C are not adjacent in G; those edges are also oriented as $A \rightarrow B \leftarrow C$ in the pattern. Then two directed acyclic graphs are weakly faithful equivalent if and only if they have identical patterns.

Let us now weaken the assumptions we have made for causal inference. Suppose that the observed distribution does not necessarily satisfy the Markov and Faithfulness Conditions for some directed acyclic graph G representing the causal structure, but that it is the marginal of a distribution that satisfies the Markov and Faithfulness Conditions for some directed acyclic graph G that represents the causal structure. By observing the conditional independence relations in a distribution, we can construct the class of latent variable models that are compatible with the conditional independence relations in the observed marginal distribution. This is the strategy used by the FCI Algorithm (see Spirtes 1992 and Spirtes, Glymour, and Scheines forthcoming.)

Let **O** be the set of variables in the observed marginal distribution. Under these assumptions, what are the limitations of any method of causal inference that uses only conditional independence relations to construct a set of latent causal graphs that are compatible with the conditional independence relations in the observed marginal distribution? Clearly, no such method can distinguish between any two causal graphs G_1 and G_2 if satisfying the Markov and Faithfulness Conditions for G_1 entails the observed conditional independence and dependence relations (i.e. just those involving members of **O**) and satisfying the Markov and Faithfulness Conditions for G_2 entails the observed conditional independence and dependence relations; we will call this **faithful entailment equivalence over O**. (Similarly, the other notions of equivalence presented above can be relativized to **O** in an analogous way.) In this paper we will describe an algorithm for deciding when two directed acyclic graphs satisfy the faithful entailment equivalence relationship over **O** that is polynomial in the number of variables in the graphs.

The general strategy that we will use to decide faithful entailment equivalence over **O** is similar to the one used to decide faithful entailment equivalence, albeit much more complex. We will describe an algorithm that given a directed acyclic graph and a set of variables **O** included in the vertices of the graph constructs an object called a POIPG over **O** for the graph. Two directed acyclic graphs are faithful entailment equivalent over **O** if and only if the algorithm constructs the same POIPG over **O** for each graph.

Note that faithful entailment equivalence over **O** does not entail strong faithful equivalence over **O**. Even if two directed acyclic graphs G_1 and G_2 satisfy the weak faithful equivalence relation over **O**, it may be that G_2 entails non-independence constraints on marginal distributions that G_1 does not. In that case, if $P(\mathbf{O})$ is the marginal of a distribution $P(\mathbf{V})$ that satisfies the Markov and Faithfulness Conditions for G_1, $P(\mathbf{O})$ may violate the non-independence constraints entailed by satisfying the Markov and Faithfulness Conditions for G_2; hence no distribution satisfying the Markov and Faithfulness Conditions for G_2 has marginal $P(\mathbf{O})$. For an example of this, see Verma and Pearl (1991). We do not know whether faithful entailment equivalence over **O** entails weak faithful equivalence over **O**.

2. Preliminary Definitions

We will need the following series of definitions to describe the algorithm.

A directed graph <**V**,**E**> is an ordered pair of a set of vertices **V**, and a set of directed edges **E**, where members of **E** are ordered pairs of distinct members of **V**; there is at most one edge between any pair of variables. In a directed graph G, we will write X -> Y if there is an edge from X to Y in G, and we will say that X is a **parent** of Y. X and Y are **adjacent in a directed graph** G if and only if either X -> Y or Y -> X in G. In a directed graph G, an **undirected path** U from X to Y (which we sometimes write as $\langle X, \ldots, Y\rangle$) is a sequence of vertices starting with X and ending with Y such that for every pair of vertices A and B that are adjacent to each other in the sequence, A and B are adjacent in G ; X and Y are the **endpoints** of U. If A and B are distinct vertices on undirected path $U = \langle X, \ldots, Y\rangle$, the subsequence of U which is equal to $\langle A,\ldots,B\rangle$ is a subpath of U denoted by $U(A,B)$. An edge $A \rightarrow B$ is **on undirected path** U if and only if $A \rightarrow B$ is in G, and A and B are adjacent on U. A path consisting of a single vertex is an **empty path.** A path is **acyclic** if no vertex in the graph occurs on the path more than once. When we mention an undirected path U, we will assume it is acyclic unless explicitly stated otherwise. An undirected path U between X and Y is **into** X if there is some Z on U such that there is an edge $Z \rightarrow X$ on U, and it is **out of** X if there is some Z on U such that there is an edge $X \rightarrow Z$ on U. In a directed acyclic graph G, a **directed path** P from X to Y is a sequence of vertices starting with X and ending with Y such that for every pair of variables A and B that are adjacent to each other in the sequence in that order, the edge $A \rightarrow B$ occurs in G. A **directed graph is acyclic** if and only if it contains no cyclic directed paths. X and Y are **adjacent on path** P (as distinct from adjacent in the graph) if and only if X and Y are adjacent in the sequence P. If an undirected path U contains an edge between X and Y, and an edge between Y and Z, the two edges **collide** at Y if and only if $X \rightarrow Y$ and $Z \rightarrow Y$ in G. On an undirected path U, Y is a **collider** if and only if there exist edges $X \rightarrow Y$ and $Z \rightarrow Y$ in U; it is an unshielded collider on U if it is a collider on U, and Z and X are not adjacent in G. X is an **ancestor** of Y and Y is a **descendant** of X if and only if there is a directed path from X to Y. (We count the sequence consisting of a single vertex $\langle X\rangle$ as a directed path from X to X, so X is its own ancestor and descendant, although it is not its own parent or child.) X, Y, and Z form **triangle** X-Y-Z in G if and only if X is adjacent to Y, Y is adjacent to Z, and Z is adjacent to X in G. A **trek** between distinct vertices X and Y is an unordered pair of directed paths from some variable Z to X and Y respectively that intersect only at Z; Z is the **source of the trek.** (One of the directed paths may be an empty path, in which case Z may equal X or Y.)

Verma and Pearl (see Pearl 1988) have shown how to calculate the conditional independence relations that are entailed by distributions satisfying the Markov condition for a graph G using the d-separability relation. In graph G, an undirected path U between distinct vertices X and Y not in **S** **d-connects variables** X **and** Y **given a set of vertices S** not containing X or Y if and only if (i) every collider on U has a descendent in **S** and (ii) no other vertex on U is in **S**. Distinct vertices X and Y are **d-separated given a set S** not containing X and Y if and only if no path d-connects X and Y given **S**.

Disjoint sets of vertices **X** and **Y** are d-separated given **S** in G if and only if every member of **X** is d-separated from every member of **Y** given **S** in G. If distribution P satisfies the Markov and Faithfulness Conditions, then for disjoint sets of vertices **X**, **Y**, and **S**, **X** is independent of **Y** conditional on **S** if and only if **X** is d-separated from **Y** given **S** in G (Pearl 1988).

Given a directed acyclic graph G over a set of variables **V**, and **O** a subset of **V**, Verma and Pearl (1991) have characterized the conditions under which two variables in **O** are not d-separated given any subset of $\mathbf{O}\backslash\{A,B\}$. If G is a directed acyclic graph over a set of variables **V**, **O** is a subset of **V** containing A and B, and $A \neq B$, then an undirected path U between A and B is an **inducing path over O** if and only if every member of **O** on U except for the endpoints is a collider on U, and every collider on U is an ancestor of either A or B. We will sometimes refer to members of **O** as **observed** or **measured** variables; variables not in **O** are **latent variables.**

Theorem 1: If G is a directed acyclic graph with vertex set **V**, and **O** is a subset of **V** containing A and B, then A and B are not d-separated by any subset **Z** of $\mathbf{O}\backslash\{A,B\}$ if and only if there is an inducing path over the subset **O** between A and B.

The inducing paths relative to **O** in a graph G over **V** can be represented in the following structure described (but not named) in Verma and Pearl (1990). G' is an **inducing path graph over O for directed acyclic graph** G if and only if

(i) **O** is a subset of the vertices in G;

(ii) there is an edge A <-> B in G' if and only if A and B are in **O**, and there is an inducing path in G between A and B over **O** that is into A and into B;

(iii) there is an edge A -> B in G' if and only if A and B are in **O**, there is no edge A <-> B in G', and there is an inducing path in G between A and B over **O** that is out of A and into B; and

(iv) there are no other edges in G'.

The inducing path graph over **O** of a directed acyclic graph G also contains no directed cycles. It can be shown that in an inducing path graph, A -> B entails that *every* inducing path over **O** between A and B is out of A and into B; A <-> B entails that there exists an inducing path over **O** that is into A and into B, but is not incompatible with there being an inducing path over **O** that is out of A and into B. A <-> B occurs in an inducing path graph

G' when in G there is a latent common cause of A and B (i.e. there is a trek between A and B whose source is not in $\mathbf{O}$.)

We can extend the concept of d-separability to inducing path graphs if the only kinds of edges that can occur on a directed path are edges with one arrowhead, but undirected paths may contain edges with either single or double arrowheads. The following theorem is proved in Spirtes, Glymour, and Scheines (forthcoming).

Theorem 2: If G is a directed acyclic graph, G' is the inducing path graph for G over $\mathbf{O}$, and $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{S}$ are disjoint sets of variables included in $\mathbf{O}$, then $\mathbf{X}$ and $\mathbf{Y}$ are d-separated given $\mathbf{S}$ in G' if and only they are d-separated given $\mathbf{S}$ in G.

If G' is an inducing path graph over $\mathbf{O}$ and $A \neq B$, let $V \in$ **D-SEP**(A,B) if and only if $A \neq V$ and there is an undirected path U between A and V such that every vertex on U except for the endpoints is a collider on U and every vertex on U is an ancestor of A or B. Theorem 2 is proved in Spirtes, Glymour, and Scheines (forthcoming).

Theorem 3: If G is a directed acyclic graph, G' is the inducing path graph for G over $\mathbf{O}$, A and B are in $\mathbf{O}$, A is not an ancestor of B, and A and B are not adjacent in G', then A and B are d-separated given **D-SEP**(A,B).

In an inducing path graph either A is not an ancestor of B or B is not an ancestor of A. Thus we can determine whether A and B are adjacent in an inducing path graph without determining whether A and B are dependent conditional on *all* subsets of $\mathbf{O}$.

A **partially oriented inducing path graph** can contain several sorts of edges: A -> B, A o-> B, A o-o B, or A <-> B. Note that there is no mark at the "A" end of A -> B; for convenience we say that the "A" end of A -> B is the empty mark, which we denote EM. We use "*" as a metasymbol to represent any of the three kinds of ends (EM (the empty mark), ">", or "o"); the "*" symbol itself does not appear in a partially oriented inducing path graph. (We also use "*" as a metasymbol to represent the two kinds of ends (EM or ">") that can occur in an inducing path graph.)

A partially oriented inducing path graph π for directed acyclic graph G with inducing path graph G' over $\mathbf{O}$ is intended to represent the adjacencies in G', and some of the orientations of the edges in G' that are common to all inducing path graphs with the same d-connection relations as G'. If G' is an inducing path graph over **O**, **Equiv**(G') is the set of inducing path graphs over the same vertices with the same d-connections as G. Every inducing path graph in **Equiv**(G') shares the same set of adjacencies. We use the following definition:

π is a **partially oriented inducing path graph (or POIPG) of directed acyclic graph** G **with inducing path graph** G' **over O** if and only if

(i) if there is any edge between A and B in π, it is one of the following kinds:

A -> B, B -> A, A o-> B, B o-> A, A o-o B, or A <-> B;

(ii) π and G' have the same vertices;

(iii) π and G' have the same adjacencies;

(iv) if A <-> B is in π, then A <-> B is in every inducing path graph in **Equiv**(G');

(v) if A -> B is in π, then A -> B is in every inducing path graph in **Equiv**(G');

(vi) if A *-* B *-* C is in π, then the edges between A and B, and B and C do not collide at B in any inducing path graph in **Equiv**(G');

(v) if A o-> B is in π, then in every inducing path graph X in **Equiv**(G'), either A -> B or A <-> B in X;

(vi) if A o-o B is in π, then in every inducing path graph X in **Equiv**(G'), either A -> B, A <- B, or A <-> B in X.

Note that an edge A *-o B does not constrain the edge between A and B either to be into or to be out of B in any subset of **Equiv**(G'). If G is a directed acyclic graph G, and G' an inducing path graph of G over **O**, any POIPG for G over **O** is also a POIPG over **O** for any directed acyclic graph whose inducing path graph over **O** is G'; hence we also speak of the POIPG over **O** of G'.

In a graph G, a variable X on a path U is **hidden** if and only if there are distinct vertices Y and Z adjacent to X on U, and Y and Z are adjacent in G; the edge between Y and Z **hides** X. In a graph G', if U is an undirected path between X and Y, Z is the vertex adjacent to Y on U, and Z is a hidden vertex on U, then U is a **semi-inducing path** between X and Y for Z if and only if X is not adjacent to Y, every vertex on U except the endpoints and possibly Z is a collider on U, and for every vertex V on U except for the endpoints and possibly Z there is an edge V -> Y in G.

2. The Faithful Entailment Equivalence Algorithm(Input: G_1, G_2, O; Output *FE*)

The input to the Faithful Entailment Equivalence Algorithm is two directed acyclic graphs G_1 and G_2 and a set **O** of vertices that is a subset of the vertices in both graphs. The output is a boolean variable *FE* that is true if and only if G_1 and G_2 are faithful entailment equivalent over **O**.

A.) Form the inducing path graph for G_1 over **O** using the Inducing Path Graph Algorithm with inputs G_1 and **O** and output G_1'.
B.) For each pair of variables *A* and *B* that are not adjacent in G_1, using the Find D-Sep Algorithm with inputs G_1', *A*, and *B*, find a set of variables **S** such that *A* and *B* are d-separated given **S** in G_1. Set **Sepset**[*A*,*B*] and **Sepset**[*B*,*A*] equal to **S**.
C.) Form the POIPG π_1 for G_1 over **O** by calling the POIPG Algorithm with inputs G_1, G_1', **O**, and **Sepset**.
D.) Form the inducing path graph for G_2 over **O** using the Inducing Path Graph Algorithm with inputs G_2 and **O** and output G_2'.
E.) Form the POIPG π_2 for G_2 over **O** by calling the POIPG Algorithm with inputs G_2, G_2', **O**, and **Sepset**.
F.). *FE* = TRUE if and only if only if $\pi_1 = \pi_2$.

3.1 The Inducing Path Graph Algorithm(Input: *G*, O: Output: *G*')

The input to the Inducing Path Graph Algorithm is a directed acyclic graph *G* with vertex set **V**, and the output is the inducing path graph of *G* over **O**, where **O** is a subset of **V**. *G*' is initialized to have no edges.

A). Determine for each pair of vertices *I*, *J* whether there is a directed path from *I* to *J* in *G*.
B). Determine for each pair of vertices *I*, *J* whether there is a directed path in *G* from *I* to *J* that does not contain any member of **O** except possibly for the endpoints.
C). Determine for each pair of vertices *I*, *J* whether there is a trek between *I* and *J* that is directed into *I* and *J* and that contains no members of **O** except possibly for the endpoints.
D). For each pair of variables *K* and *L* in **O**
    for each pair of variable *I* and *J* in **V** that are ancestors of *K* or *L* in *G*
        determine whether there is an undirected path between *I* and *J* that is into *I* and into *J* such that each collider on the path is an ancestor of *K* or of *L*, and each member of **O** on the path except for the endpoints is a collider on the path;

if there is such an undirected path between K and L, add the edge K <-> L to G'.

E). For each pair of variables K and L in **O** not connected by a double-headed arrow

for each ordered pair of variables I and J in **V**

determine whether there is an undirected path between I and J that is not into I such that each collider on the path is an ancestor of K or of L, and each member of **O** on the path except for the endpoints is a collider on the path;

if there is such an undirected path between K and L, add the edge K -> L.

Steps D) and E) determine whether K is adjacent to L in the inducing path graph, and what kind of edge connects them. Step A), B), and C), are simply results used in D) and E). We discuss each step in more detail below.

Lemma 1: There is a trek between a pair of variables I and J that is into I and into J and that contains no members of **O** with the possible exception of the endpoints if and only if there is a vertex K not in $\mathbf{O} \cup \{I,J\}$ such that there is a directed path from K to I that contains no member of $\mathbf{O} \cup \{I,J\}$ except for the endpoint I, and a directed path from K to J that contains no member of $\mathbf{O} \cup \{I,J\}$ except for the endpoint J.

Proof. The "only if" direction is trivial. For the "if" direction let M be the last point of intersection of the two directed paths. M is not equal to I or J because it occurs on both paths. The subpaths from M to I and M to J form a trek between I and J that is into I and into J and that contains no member of **O** with the possible exception of the endpoints. Q.E.D.

Lemma 2: There is an undirected path U between variables M and N that is into M if M is in **O** and into N if N is in **O**, such that the endpoints and every collider on U is an ancestor of I or J and every member of **O** on U except for the endpoints is a collider on U, if and only if there is an acyclic sequence of vertices from M to N such that for any pair of vertices K and L adjacent in the sequence, K and L are ancestors of I or J, and there is a trek between K and L that is into K if K is in **O** and into L if L is in **O**, and that contains no member of **O** with the possible exception of the endpoints.

Proof. The "only if" direction is trivial. We will now prove the "if" direction. Suppose that there is a sequence of vertices such that for any pair of vertices K and L adjacent in the sequence, K and L are ancestors of I or J, there is a trek between K and L that is into K if K is in **O** and into L if L is in **O** and contains no member of **O** with the possible exception of the endpoints. It is possible to form an undirected path U between M and N out of edges in the sequence of treks. Because each trek with an endpoint Q in **O** is into Q, and each vertex in **O** occurs only as an endpoint of a trek it follows that the only edges containing a vertex in **O** are into that vertex. Hence if a vertex on U is in **O** it is a collider on U or an endpoint of U. Because the endpoints of each trek are ancestors of either I or J, every vertex on each trek is an ancestor of I or J. Hence every collider on U is an ancestor of I or J. U is into M if M is in **O** because if M is in **O** every edge on every trek that contains M is into M. Similarly U is into N if N is in **O** . Q.E.D.

Lemma 3: For each I and J in **O** such that there is no inducing path between I and J over **O** that is into I and into J, there is an inducing path over **O** between variables I and J in **O** that is out of I if and only if either there is a directed path from I to J that contains no members of **O** except for the endpoints, or there is a vertex K such that there is a directed path D from I to K that contains no member of **O** except for the endpoints, and an undirected path U between K and J that is into K and into J, such that every member of **O** that is on U except for the endpoints is a collider on U, and the endpoints and every collider on U is an ancestor of either I or J.

Proof. The "only if" direction is trivial. We will now prove the "if" direction. If there is a directed path from I to J that contains no member of **O** except for the endpoints then it is trivial that there is an inducing path between I and J over **O**. Suppose then that there is a directed path D from I to K, and an undirected path U between K and J that is into K, and into J, such that every member of **O** that is on U except for the endpoints is a collider on U, and every collider on U is an ancestor of either I or J. I is not on U because otherwise $U(I,J)$ is an inducing path between I and J over **O** that is into I and into J, contrary to the hypothesis. Let M be the first vertex on D which is also on U. M is on a trek between some pair of vertices A and B on U, where B is between A and J. If M is on the branch of the trek with sink B, then M is not a collider or a member of **O** on the concatenation of $D(I,M)$ and $U(M,J)$, so the concatenation is an inducing path over **O** between I and J which is out of I. If M is not on the branch of the trek with sink B, it is on the branch of the trek with sink A. In that case M is a collider on the concatenation of $D(I,M)$ and $U(M,J)$ that has a descendant in I or J, so the concatenation is an inducing path between I and J over **O** that is out of I. Q.E.D.

We repeatedly use the following Path Sum Algorithm described in Aho, Hopcroft and Ullman (1974). C_{ij}^{k} represents the sum over all paths of the product of labels on each path from V_i to V_j that with the possible exception of the endpoints contains only vertices in the set $\{V_1,\ldots,V_k\}$. We use Boolean addition in the algorithm, i.e. $1 + 1 = 1$. The input to the algorithms is a directed acyclic graph whose edges are labeled by a real number. The cost of a path is the product of the labels of the edges on the path. The output is a cost function c, where $c(I,J)$ represents the sum of the costs of all directed paths from I to J. The Path Sum Algorithm is $O(n^3)$.

Path Sum Algorithm(Input: G: Output: c)

A.) for $1 \leq i,j <= n$ and $i \neq j$ do $C_{ij}^{0} \leftarrow l(V_i, V_j)$

B.) for $k := 1$ to n do

for $1 <= i,j <= n$ and $i \neq j$ do

$$C_{ij}^{k} \leftarrow C_{ij}^{k-1} + C_{ik}^{k-1} * C_{kj}^{k-1}$$

C.) for $1 \leq i,j \leq n$ do $c(V_i, V_j) \leftarrow C_{ij}^{n}$

Step A) of the Inducing Path Algorithm can be performed by setting $l(V_i,V_j) = 1$ if there is an edge between V_i and V_j in directed acyclic graph G and 0 otherwise. There is a directed path from A to B in G if and only if $c(A,B) = 1$. If there are n vertices in a directed acyclic graph G, then the algorithm is $O(n^3)$.

Step B) can be performed by ordering the variables so that all of the members of **O** are last in the ordering. Set $l(V_i,V_j) = 1$ if there is an edge between V_i and V_j in directed acyclic graph G and 0 otherwise. Perform the Path Sum Algorithm except in step B) the inner for loop runs only to $n - m$, instead of to n. There is a directed path from A to B in G that with the possible exception of the endpoints contains no member of **O** if and only if $c(A,B) = 1$. This step is $O(n^3)$.

Step C) is based upon lemma 1. First order the variables so that I, J, and all of the variables in **O** follow all of the other variables. Then set $l(K,L) = 1$ if and only if there is a directed edge from K to L, and apply the Path Sum Algorithm, replacing n in the inner for loop of step B) of the Algorithm by n minus the number of vertices in $\mathbf{O} \cup \{I,J\}$. $c(K,L) = 1$ if and only if there is a directed path from K to L that contains no vertices in $\mathbf{O} \cup \{I,J\}$ with the possible exception of the endpoints. Then there is a trek between I and J that is into I and into J and contains no members of **O** except for the endpoints if and only if there is a vertex $K \neq I, J$ such that $c(K,I) = c(K,J) = 1$. This is $O(n^5)$ because it requires applying the Path Sum Algorithm $O(n^2)$ times (once for each pair of vertices I, J).

Step D) is based on lemma 2. For a given pair of variables I and J in **O**, in order to determine if there is an inducing path between I and J that is into I and into J, set $l(K,L) = 1$ if K is an ancestor of I or J, L is an ancestor of I or J, and there is a trek between K and L that is into K if K is in **O** and into L if L is in **O** and contains no members of **O** with the possible exception of the endpoints (determined from step C) and 0 otherwise. Then apply the Path Sum Algorithm. Then set A <-> B if and only if $c(A,B) = 1$. This is $O(m^2n^3)$ because it requires applying the Path Sum Algorithm $O(m^2)$ times (once for each pair of vertices in **O**).

Step E) is based on lemma 3. First set $c(I,J) = 0$. For a given pair of variables I and J in **O** for which there is no inducing path over **O** between I and J into I and into J, determine if there is a directed path from I to J that contains no member of **O** except for the endpoints (from step B); if there is set $c(I,J) = 1$. If there is not, determine if there is a vertex K not equal to I or J, such that there is a directed path from I to K that does not contain any members of **O** except possibly for the endpoints (determined from step B), and an undirected path U from K to J that is into K and into J and on which the endpoints and every collider is an ancestor of I or J, and every member of **O** on U except for the endpoints is a collider (determined from step D). If there is then reset $c(I,J) = 1$. This is $O(n^3)$. Then set A -> B if and only if $c(A,B) = 1$.

The algorithm is dominated by step C), which is $O(n^5)$.

3.2 The Find D-SEP Algorithm

The input to the Find D-SEP Algorithm is an inducing path graph G' and two vertices I and J in G'. The output **S** is equal to **D-SEP**(I,J).

Find D-SEP Algorithm(Input: G',I,J: **Output:** S)

Set $l(K,L) = 1$ if there is an edge K <-> L in G', and K and L are ancestors of I or J.
Call Path Sum Algorithm(G',c).
If $V \neq I$ and V is a parent of I or a parent of some vertex K for which $c(I,K) = 1$, add V to **S**. If $V \neq I$ and $c(I,V) = 1$, add V to **S**.

If I is not an ancestor of J, and I and J are not adjacent in G', then every member V of **D-SEP**(I,J) is on an undirected path between I and V that can be divided into two subpaths. The first (possibly empty) subpath is a path that consists entirely of double-headed arrows between I and some vertex X (possibly equal to V) such that each vertex on the path is an ancestor of I or J. The second subpath, if it exists, is a single edge from V to X. The output of the Path Sum Algorithm in this case sets $c(K,L) = 1$ if and only if there is an undirected path between K and L that consists of all double-headed arrows that are ancestors of I or J. Then vertices not equal to I that are the endpoints of paths for which $c(I,V) = 1$ are added to **S**, and vertices not equal to I that are parents of endpoints of paths for which $c(I,V) = 1$ are added to **S**. The algorithm is $O(m^3)$ because of the call to the Path Sum Algorithm. It is called $O(m^2)$ times (one for each pair of vertices in **O** that are not adjacent.) Hence Step B of the POIPG Algorithm is $O(m^5)$.

3.3 The POIPG Algorithm(Input: G, O, G', Sepset; Output: π)

The input to the POIPG Algorithm is a directed acyclic graph G; **O**, a subset of the vertices in G; G', an inducing path graph for G over **O**; and **Sepset**, a two dimensional array of vertices such that for each A and B in **O**, **Sepset**$[A,B]$ is in **O**, **Sepset**$[A,B]$ = **Sepset**$[B,A]$, and A and B are d-separated given **Sepset**$[A,B]$ in G. The output is a partially oriented inducing path graph of G relative to **O**. Let n be the number of vertices in G, and m be the number of vertices in **O**.

A.) For each pair of vertices A and B in **O**, place an undirected edge between A and B if and only if A and B are adjacent in G'.

B). For each triple of vertices A, B, C such that the pair A, B and the pair B, C are each adjacent in F but the pair A, C are not adjacent in G', orient A *-* B *-* C as A *-> B <-* C if and only if B is not in **Sepset**$[A,C]$, and orient A *-* B *-* C as $A\ *\text{-}\underline{*\ B\ *}\text{-}*\ C$ if and only if B is in **Sepset**$[A,C]$.

C). repeat

1) If there is a directed path from A to B in π, and an edge A *-* B, orient A *-* B as A *-> B,

2) else if B is a collider along $\langle A,B,C\rangle$ in π, B is adjacent to D, and A and C are not d-connected given D, then orient B *-* D as B <-* D,

3) else if U is a semi-inducing path between A and B for M in π, and P and R are adjacent to M on U, and P - M - R is a triangle, then

if M is in **Sepset**$[A,B]$ then M is marked as a non-collider on subpath $P\ *\text{-}\underline{*\ M\ *}\text{-}*\ R$

else P *-* M *-* R is oriented as P *-> M <- * R.

4) else if $P\ *\text{-}\underline{>M\ *}\text{-}*\ R$ then orient as P *-> M -> R.

until no more edges can be oriented

(When we say orient an edge as P *-> M we mean that the orientation step does not change the P end of the edge between P and M.)

3.3.1 Step A

Step A) is $O(m^2)$.

3.3.2: Step B

Step B) is $O(m^3)$.

3.3.3: Step C

The only difficult part of step C) is finding semi-inducing paths in a partially oriented inducing path graph. The proof that the orientations are correct follows from Theorem 6.4 in Spirtes, Glymour, and Scheines (forthcoming).

3.3.3.1 Step C1

This is $O(m^3)$ using the Path Sum Algorithm.

3.3.3.2 Step C2

This is $O(m^4)$ because it requires examining each quadruple A, B, C, and D of vertices in **O**.

3.3.3.3 Step C3

This can be performed by the following algorithm which determines for a given quadruple of variables A, P, M, and R, if there is a semi-inducing path U between A and R for M on which P and R are adjacent to M. A semi-inducing path between A and R containing subpath P *-* M *-* R can be broken into three parts: the subpath P *-* M *-* R; a (possibly empty) path U between a vertex B and P that does not contain M or R, such that every vertex on U is a parent of R; and an edge A *-> B.

The input to Find Semi-Inducing Paths is an inducing path graph G', and the output is a variable **Semi** that is an array of ordered triples of vertices, such that <A,P,R> is in **Semi**[M] if and only if there is a semi-inducing path for M between A and R, containing subpath <P,M,R>.

Algorithm Find Semi-Inducing Paths(Input: G': **Output**: **Semi**);

A.) For each pair of distinct vertices M and R in G' such that M is adjacent to R,

1.) Call Algorithm Find Collider-Paths(M,R)

2.) For each vertex P such that P is not equal to M or R, P <-* M and P -> R

a.) for each vertex A distinct from M, P, and R, **Semi**[M] = $\langle A,P,R \rangle$ if A is not adjacent to R and A *-> P;

b.) for each pair of vertices A and B distinct from each other and M, P, and R, **Semi**[M] = $\langle A,P,R \rangle$ if $c(B,P) = 1$, A is not adjacent to R, and A *-> B.

Step A2) finds the third part of a semi-inducing path, and steps A2a) and A2b) find the first and second parts of a semi-inducing path. We show below that Algorithm Find Collider-Paths is $O(m^3)$. Step A2) is also $O(m^3)$. These steps are each called $O(m^2)$ times, so the entire algorithm is $O(m^5)$.

The following algorithm searches for the second part of a semi-inducing path. G' is an inducing path graph over **O**, and M and R are vertices in G'. The path U between B and P exists if and only if $c(B,P) = 1$.

Algorithm Find Collider-Paths(Input: G', M, R; **Output:** c);

A.) Order the variables in G' so that M and R come last in the ordering.

B.) For each pair of variables J and K, set $l(J,K) = 1$ if and only if J <-> K in G' and J and K are parents of R.

C.) Apply the Path Sum Algorithm, where the inner for loop in step B) iterates up to only m-2.

Algorithm Find Collider-Paths is $O(m^3)$ because of step C)

3.3.3.4 Step C4

Step C4) is $O(m^3)$.

Each time the repeat loop is entered, it either adds an arrowhead to an arrow, or it marks a vertex on a path of two edges as a non-collider, or it removes a non-colliding mark from a vertex on a path of two edges. Hence there $O(m^3)$ marks that can be placed on the partially oriented inducing path graph, and the repeat loop can be entered at most $O(m^3)$ times. Each time the repeat loop is entered, the dominating action is step C3), which is $O(m^5)$. Hence step C) is $O(m^8)$. Overall then the complexity of the POIPG algorithm is $O(m^8)$.

Appendix: The Correctness and Complexity of the Faithful Entailment Equivalence Algorithm

Lemma 6.6.1 and 6.6.2 are from Spirtes, Glymour, and Scheines (forthcoming.)

Lemma 6.6.1: Suppose that G is a directed acyclic graph, and in G there is a sequence of vertices M starting with A and ending with C, and a set of paths F such that for every pair of vertices I and J adjacent in M there is exactly one inducing path W over $\mathbf{O}$ between I and J in F. Suppose further that if $J \neq C$ then W is into J, and if $I \neq A$ then W is into I, and I and J are ancestors of either A or C. Then in G there is an inducing path T over $\mathbf{O}$ between A and C such that if the path in F between A and its successor in M is into A then U is into A, and if the path in F between C and its predecessor in M is into C then U is into C.

In an inducing path or directed acyclic graph G that contains an undirected path U between X and Y, the edge between V and W is **substitutable** for $U(V,W)$ in U if and only if V and W are on U, V is between X and W on U, G contains an edge between V and W that is not on U, V is a collider on the concatenation of $U(X,V)$ and the edge between V and W if and only if it is a collider on U, and W is a collider on the concatenation of $U(Y,W)$ and the edge between V and W if and only if it is a collider on U.

Lemma 6.6.2: If G' is an inducing path graph for directed acyclic graph G over $\mathbf{O}$, C is a descendant of B in G, and U is an undirected path in G' between X and R containing subpath A *-> B <-> C where A is between X and B, then in G' there is a vertex E on U between X and A inclusive such that the edge between E and C is substitutable for $U(E,C)$ in U. Furthermore the concatenation of $U(X,E)$ and the edge between E and C is into C, and if U is into X, then the concatenation of $U(X,E)$ and the edge between E and C is into X.

Let the **length of a path** be the number of edges on the path. If there is a directed path from a vertex C to a member of $\mathbf{Z}$, let Minlength(C,$\mathbf{Z}$) be the length of a shortest directed path from C to a member of $\mathbf{Z}$. If U is an undirected path that d-connects X and Y given $\mathbf{Z}$, let d-length(U,$\mathbf{Z}$) be the sum over the colliders C on U of Minlength(C,$\mathbf{Z}$). U is a **minimal d-connecting path** between X and Y given $\mathbf{Z}$, if and only if U is a d-connecting path between X and Y given $\mathbf{Z}$, and there is no path W that d-connects X and Y given $\mathbf{Z}$ such that either

(i) the length of W is less than the length of U, or

(ii) the length of W equals the length of U, and d-length(W) is less than d-length(U).

Clearly if there is a path that d-connects X and Y given $\mathbf{Z}$, then there is at least one minimal path that d-connects X and Y given $\mathbf{Z}$.

Lemma 4: In an inducing path graph G, if an undirected path U that d-connects X and Y given $\mathbf{Z}$ contains subpath M *-> A -> R, where M is between X and A, and there is a vertex B on U such that A is between X and B, there is an edge between A and B that is into A and into B that is not on U, U' is the concatenation of $U(X,A)$ the edge between A and B and $U(B,Y)$, A is a collider on U', and B is a descendant of A in G, then U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$.

Proof. See figure 1.

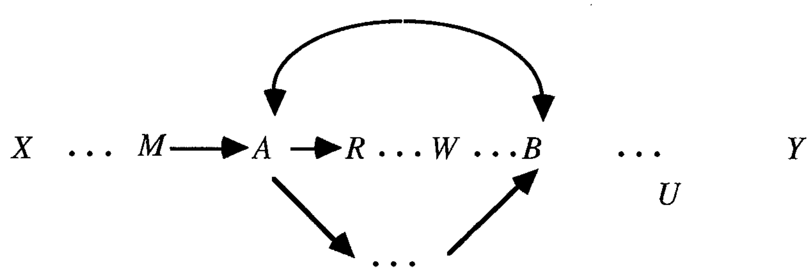

Figure 1

$X \neq A$ because U contains a subpath M *-> A -> R. By lemma 6.6.2 there is a vertex E between X and M inclusive such that the edge between E and B is substitutable for $U'(E,B)$ in U'. Let U'' be the concatenation of $U'(X,E)$, the edge between E and B, and $U'(B,Y)$. E is a collider on U'' if and only if it is a collider on U', and because $U'(X,A) = U(X,A)$, E is a collider on U'' if and only if it is a collider on U. The edge between E and B is into B; hence if B is a collider on U then the edge between E and B is substitutable for $U(E,B)$ in U, and U is not minimal. Suppose then that B is not a collider on U. Let W be the vertex on U adjacent to B and between B and X. If B is not a collider on U' then B is not a collider on U, the edge between E and B is substitutable for $U(E,B)$ in U, and U is not minimal. Suppose then that B is a collider on U'. Because B is a collider on U', but not on U, it follows that there is a directed edge B -> W on U. Hence some descendant of B on U is a collider on U, and has a descendant in $\mathbf{Z}$. It follows that B has a descendant in $\mathbf{Z}$. On U'', every vertex except B is a collider on U'' if and only if it is a collider on U. B is a collider on U'', and has a descendant in $\mathbf{Z}$. Hence U'' d-connects X and Y given $\mathbf{Z}$. But U'' is shorter than U, so U is not minimal. Q.E.D.

Lemma 5: In an inducing path graph G', if U is a path that d-connects X and Y given $\mathbf{Z}$, and there are vertices M and N on U that are adjacent in G' but not on U, and the edge between M and N is substitutable for $U(M,N)$ in U then U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$.

Proof. Suppose without loss of generality that M is between X and N. Let U' be concatenation of $U(X,M)$, the edge between M and N, and $U(N,Y)$. By the definition of substitutable, for every vertex V on U', V is a collider on U' if and only if it is a collider on U. Because U d-connects X and Y given $\mathbf{Z}$, U' d-connects X and Y given $\mathbf{Z}$. But U' is shorter than U, so U is not minimal. Q.E.D.

In a graph G, if U is an undirected path between X and Y that contains vertices A and B, A is between X and B, there is an edge between A and B that is not in U, and A is a collider on U if and only if A is not a collider on the concatenation of $U(X,A)$ and the edge between A and B, then the edge between A and B **disagrees with** $U(A,B)$ in U at A.

If U is an undirected path between X and Y, we will say that A is **on the** X **side** of B if and only if A is equal to X or A is between X and B on U. If E_1 and E_2 are single-headed edges (i.e. "->" and not "<->") that hide vertices on U, then E_1 and E_2 **point in the same direction** whenever the end of E_1 with an ">" is on the X side of the end of E_1 without an arrow if and only if the end of E_2 with an ">" is on the X side of the end of E_2 without an arrow. See figure 2.

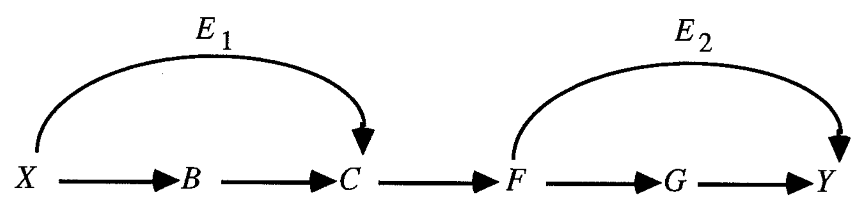

Figure 2: E_1 and E_2 point in the same direction

Lemma 6: In an inducing path graph G' over $\mathbf{O}$, if U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, and W is a hidden variable on U, then there is a unique subpath of U that is a semi-inducing path for W.

Proof. Suppose that W is a hidden variable on U and U is a minimal d-connecting path between X and Y given $\mathbf{Z}$. First we will show that some subpath of U is a semi-inducing path for W. Let R be the vertex on U adjacent to W and on the X side of W, and V be the vertex on U adjacent to W and on the Y side of W. Let E be the edge between R and V. See figure 3.

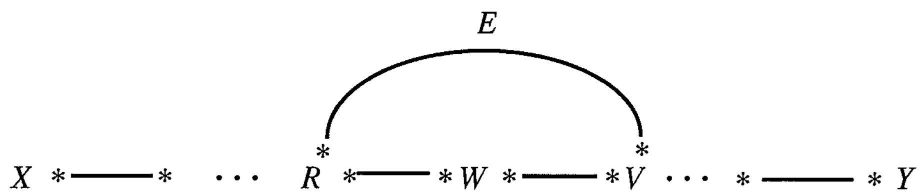

Figure 3

Because U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, by lemma 5 E disagrees with $U(R,V)$ in U at R or V. Note that if E disagrees with $U(R,V)$ in U at R, then $U(X,R)$ is into R and $U(R,Y)$ is out of R, and if E disagrees with $U(R,V)$ in U at V then $U(V,Y)$ is into V and $U(V,X)$ is out of V. There are three cases: either V <-> R in G', V -> R in G', or R -> V in G'. Let U' be the concatenation of $U(X,R)$, E and $U(V,Y)$.

Suppose first that V <-> R in G'. If U is a minimal d-connecting path between X and Y given $\mathbf{Z}$ then the edge between V and R disagrees with $U(V,R)$ on U at R, or V, or both. Hence on U, either V -> W, or R -> W, or both. Suppose first that R -> W and V -> W. It follows that W is a collider on U, and because U d-connects X and Y given $\mathbf{Z}$, W has a descendant in $\mathbf{Z}$. Hence V and R have descendants in $\mathbf{Z}$. Every vertex on U' except for V and R is a collider on U' if and only if it is a collider on U; and V and R are colliders on U and have descendants in $\mathbf{Z}$; hence U' d-connects X and Y given $\mathbf{Z}$ if U does. It follows that U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$, contrary to our assumption. Suppose then without loss of generality that R -> W and W *-> V. If W <-> V, then R has a descendant in $\mathbf{Z}$ because W is a collider on U; also V is a collider on U' if and only if it is a collider on U. Hence U' d-connects X and Y given $\mathbf{Z}$ if U does. It follows that U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$, contrary to our assumption. Suppose then that W -> V on U. Let T be the vertex adjacent to R on U, and on the X side of R. If T <- R on U then the edge R <-> V is substitutable for the subpath $U(R,V)$ in U, and U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$, contrary to our assumption. If T *-> R then by lemma 4, U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$, contrary to our assumption.

The cases V -> R and R -> V are essentially the same as each other, so without loss of generality we will consider only the case where R -> V. Suppose that R -> V in G'. First we will show that R -> V disagrees with $U(R,V)$ in U at R. Suppose that R -> V agrees with $U(R,V)$ in U at R. Because U is minimal, R -> V disagrees with $U(R,V)$ in U at V. It follows that V -> W in G'. Because G' is acyclic, R *-> W in G'. Hence W is a collider on U, and has a descendant in **Z**. It follows that although V is a collider on U' but not on U, it has a descendant in **Z**. Every vertex except V on U' is a collider on U' if and only if it is a collider on U. Hence U is not a minimal d-connecting path between X and Y given **Z**, which is a contradiction. It follows that R -> V disagrees with $U(R,V)$ in U at R. Hence R is a collider on U.

Let S be the vertex on $U(X,R)$ closest to R such that either there is no edge S -> V in G', or S is not a collider on U. It follows that for all vertices M between S and W on U that M -> V in G', and M is a collider on U. There are four cases: either S -> V, S is not adjacent to V, there is an edge S <- V in G', or an edge S <-> V in G' If S and V are adjacent, let U' be the concatenation of $U(X,S)$, the edge between S and V, and $U(V,Y)$.

Suppose first that there is an edge S -> V. By assumption then S is not a collider on U. If S -> V agrees with $U(S,V)$ in U at V, then it follows that the S -> V edge is substitutable for $U(S,V)$ in U, and U is not minimal. Suppose then that S -> V disagrees with $U(S,V)$ in U at V. It follows then that V is not a collider on U, but is a collider on U'. Hence the edge between V and W on U is V -> W. It follows that V has a descendant that is a collider on U, and hence V has a descendant in **Z**. It follows that U' d-connects X and Y given **Z**, so U is not a minimal d-connecting path between X and Y given **Z**.

Suppose next that S is not adjacent to V. Then $U(S,V)$ is a semi-inducing path for W.

Suppose next that S <-> V in G'. First we will show that either V is a collider on both U and U', or that V is a collider on U' and has a descendant in **Z**. If V is not a collider on U and U' then V is a collider on U' but not on U. Hence the edge between W and V on U is oriented as V -> W. It follows then that V has a descendant on U that is a collider on U, and hence V has a descendant in **Z**. It follows that either V is a collider on both U and U', or that V is a collider on U' and has a descendant in **Z**. If S is a collider on both U and U', then U' d-connects X and Y given **Z**, and U is not a minimal d-connecting path between X and Y given **Z**. Suppose then that S is a collider on U', but not on U. It follows that there exist vertices M and N on U such that U contains the subpath M *-> S -> N, where S is between N and X. There is an N-> V edge by the definition of S, so S is an ancestor of V. By lemma 4 then, U is not a minimal d-connecting path between X and Y given **Z**.

Suppose finally that V -> S in G'. $S \neq R$ because otherwise there is a directed cycle in G', and $S \neq W$ because S is on the X side of W. Let T be the vertex on U adjacent to S and between S and Y. There is no S -> T edge in G' else G' is cyclic. Because $S \neq W$, T is a collider on U, so there is an edge S <-> T on U. Let A be the vertex on U that is adjacent to T and not equal to S. If $A \neq W$, then A is a collider on U and an ancestor of V. Hence, there is a path between S and A on which each vertex except the endpoints is a collider, and each collider is an ancestor of S (because V is an ancestor of S.) By lemma 6.6.1 there is an S <-> A edge in G'. It is substitutable for $U(S,A)$ in U and hence U is not minimal. Suppose then that $A = W$ (in which case $T = R$). By lemma 6.6.1, there is an S <-* W edge in G'. If S <- W in G', then R <- W in G', so S <- W is substitutable for $U(S,W)$ in U, and U is not minimal. If S <-> W in G', then if R <-> W in G', S <-> W is substitutable for $U(S,W)$ in U, and U is not minimal. If not R <-> W in G', it follows that R <- W in G' because R is a collider on U. W is not in $\mathbf{Z}$ but has a descendant (R) that has a descendant in $\mathbf{Z}$. Hence the concatenation of $U(X,S)$, the edge between S and W, and $U(W,Y)$ d-connects X and Y given $\mathbf{Z}$, and U is not a minimal c-connecting path between X and Y given $\mathbf{Z}$.

Now we will show that there is at most one subpath of U that is a semi-inducing path for W. Suppose contrary to the hypothesis that U_i and U_j are distinct subpaths of U that are semi-inducing paths for W. Let E_{1i} be the vertex on U_i that is adjacent to W, and E_{2i} be the other endpoint of U_i. Similarly, let E_{1j} be the vertex on U_j that is adjacent to W, and E_{2j} the other endpoint of U_j. If $E_{1i} \neq E_{1j}$, then E_{1j} is on U_i, $E_{1j} \neq W$, and $E_{1j} \neq E_{2i}$ (because E_{2i} is not adjacent to W. Hence E_{1j} is a parent of E_{1i}. Similarly, E_{1i} is a parent ofE_{1j}. It follows that G' is cyclic, which is impossible. Suppose then that $E_{1i} = E_{1j}$. It follows that $E_{2i} \pi E_{2j}$. Hence either E_{2i} is on U_j, or E_{2j} is on U_i. Suppose without loss of generality that E_{2j} is on U_i. $E_{2j} \neq E_{1i}$ because $E_{2j} \neq E_{1j}$. $E_{2j} \neq W$ by definition of semi-inducing path for W. It follows that E_{2j} is a parent of E_{1i}, and hence of E_{1j}. But then U_j is not a semi-inducing path for W. Q.E.D.

Lemma 7: In an inducing path graph G', if U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, then there is no pair of distinct vertices A_i, A_j such that A_i is a hidden vertex on the semi-inducing path for A_j on U, and A_j is a hidden vertex on the semi-inducing path for A_i on U.

Proof. Let U_i be the semi-inducing path for A_i on U, and U_j be the semi-inducing path for A_j on U. Let E_{1i} be the endpoint of U_i that is adjacent to A_i, and E_{2i} be the other endpoint of U_i. Similarly, let E_{1j} be the endpoint of U_j be the endpoint of U_j that is adjacent to A_j, and E_{2j} be the other endpoint of U_j. By hypothesis, A_i is a hidden vertex on U_j. Hence A_i lies between E_{1j} and E_{2j} exclusive. Similarly, A_j lies between E_{1i} and E_{2i} exclusive. A_i is a collider on U_j and hence on U, because A_i is on U_j, and is not equal to either of the endpoints or A_j. Similarly, A_j is a collider on U_i, and hence on U.

E_{1i} is adjacent to A_i, and hence between E_{1j} and E_{2j} inclusive. Similarly, E_{1j} is between E_{1i} and E_{2i} inclusive.

We will now show that $E_{1i} \neq E_{1j}$. Suppose on the contrary that $E_{1i} = E_{1j}$. Then A_i and A_j are on opposite sides of E_{1i}, and A_j is not on U_i, contrary to our hypothesis.

Suppose then that $E_{1i} = E_{2j}$. Then all of the vertices between E_{1i} and E_{2i} are colliders and ancestors of E_{1i}, with the possible exception of A_i. However, A_i is a collider on U_i because it is a collider on U, and not an endpoint of U_i. Also A_i is an ancestor or E_{2j} because it is on U_j, is not an endpoint of U_j, and is not equal to A_j. Because $E_{2j} = E_{1i}$, A_i is an ancestor of E_{1i}. Hence every vertex between E_{1i} and E_{2i} is a collider and an ancestor of either E_{1i} or E_{2i}. But then there is an inducing path between E_{1i} and E_{2i}, so E_{1i} and E_{2i} are adjacent. It follows then that U_i is not a semi-inducing path, contrary to our hypothesis.

Similarly, if $E_{1j} = E_{2i}$, then U_j is not a semi-inducing path, contrary to our hypothesis.

Suppose then that $E_{1i} \neq E_{2j}$ and $E_{1j} \neq E_{2i}$. E_{1i} is on U_j, it is not an endpoint of U_j, and it is not equal to A_j (because A_j is between E_{1i} and E_{2i}). Hence E_{1i} is an ancestor of E_{1j}. Similarly E_{1j} is an ancestor of E_{1i}. But then G' is cyclic, contrary to our assumption.

It follows that it is not the case that both A_i is a hidden vertex on the semi-inducing path for A_j on U and that A_j is a hidden vertex on the semi-inducing path for A_i on U. Q.E.D.

Lemma 8: In an inducing path graph G', if U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, then there is no triple of distinct hidden vertices A_{i-1}, A_i, A_{i+1} on U such that A_{i-1} is a hidden vertex on the semi-inducing path for A_i on U, A_i is a hidden vertex on the semi-inducing path for A_{i+1} on U, and A_{i+1} is between A_i and A_{i-1} on U.
Proof. Let U_i be the semi-inducing path for A_i on U, and similarly for U_{r+1}. Because A_{i-1} is a hidden vertex on the semi-inducing path for A_i on U, every vertex between A_{i-1} and A_i is on the semi-inducing path for A_i on U. Hence A_{i+1} is on the semi-inducing path for A_i on U. Because A_{i+1} is between A_i and A_{i-1}, and neither A_i nor A_{i-1} is an endpoint of U_i,

A_{i+1} is not an endpoint of U_i; it follows that each of the vertices adjacent to A_{i+1} on U are also on U_i. Because A_{i+1} is a hidden vertex on U, and both of the vertices adjacent to A_{i+1} are also on the semi-inducing path for A_i on U, A_{i+1} is a hidden vertex on U_i. By hypothesis, A_i is a hidden vertex on U_{i+1}. But this contradicts lemma 7. Q.E.D.

Lemma 9: In an inducing path graph G', if U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, then there is no quadruple of distinct hidden vertices A_i, A_{r+1}, A_r, A_{i+1} in that order on U such that A_i is a hidden vertex on the semi-inducing path for A_{i+1} on U, and A_r is a hidden vertex on the semi-inducing path for A_{r+1} on U.

Proof. Let U_{i+1} be the semi-inducing path for A_{i+1} on U, and similarly for U_{r+1}. Suppose contrary to the hypothesis there is a quadruple of distinct hidden vertices A_i, A_{r+1}, A_r, A_{i+1} in that order on U such that A_i is a hidden vertex on the semi-inducing path for A_{i+1} on U, and A_r is a hidden vertex on the semi-inducing path for A_{r+1} on U. A_{r+1} is on U_{i+1} because it is between A_i and A_{i+1}, and A_i is on U_{i+1}. A_{r+1} is a hidden vertex on U, and both of the vertices adjacent to A_{r+1} on U are also on U_{i+1}, because A_{r+1} is not an endpoint of U_{i+1}. Hence A_{r+1} is a hidden vertex on U_{i+1}. Similarly, A_{i+1} is a hidden vertex on U_{r+1}. But this contradicts lemma 7. Q.E.D.

Lemma 10: In an inducing path graph G', if U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, then there is no sequence of length greater than 1 of distinct vertices $\langle A_1,A_2,\ldots,A_n,A_1\rangle$ such that for each pair of vertices A_i, A_{i+1} that are adjacent in the sequence, A_i is a hidden vertex on the semi-inducing path of A_{i+1} on U.

Proof. Suppose without loss of generality that A_1 is to the right of A_n on U. Let r be the highest index such that A_r is to the right of A_1 and A_{r+1} is to the left of A_1, if such a pair exists; otherwise let $r = 1$. We will now show that A_{r+1} is to the left of A_n. Suppose first that $r = 1$. We will show that all vertices not equal to A_1 are to the left of A_1. Suppose some vertex is to the right of A_1. Let A_i be the vertex with the highest index to the right of A_1. By hypothesis, there is a vertex with index $n > i$ such that A_n is to the left of A_1; hence there is a vertex A_{i+1} in the sequence. A_{i+1} is to the left of A_1 by definition. But then $r \neq 1$. It follows that if $r = 1$, every vertex except A_1 is to the left of A_1, so A_2 is to the left of A_1. $A_2 \neq A_n$ by lemma 7. By lemma 8 then A_2 is not between A_1 and A_n, so it is to the left of A_n. If $r \neq 1$, then A_{r+1} is not between A_1 and A_n by lemma 9. Hence A_{r+1} is to the left of A_n.

We will now show that some vertex A_{s+1} whose index is greater than $r+1$ is to the right of A_r. A_{r+2} is not equal to A_n by lemma 8. A_{r+2} is not between A_{r+1} and A_r by lemma 8. A_{r+2}

is not between A_r and A_{r+1} by lemma 8. There are two cases. If A_{r+2} is to the right of A_r, then we are done. Suppose then that A_{r+2} is to the left of A_{r+1}. It follows that there is some vertex with index greater than r+2 (e.g. A_n) on the other side of A_{r+1}. Let A_s and A_{s+1} be on opposite sides of A_{r+1} (where A_s is to the left of A_{r+1}.) A_{s+1} is not between A_{r+1} and A_r by lemma 9. So A_{s+1} is to the right of A_r.

We will now show that some vertex A_t whose index is greater than s+1 is to the left of A_s. A_{s+2} is not between A_s and A_{s+1} by lemma 8. If A_{s+2} is to the left of A_s then we are done. Suppose then A_{s+2} is to the right of A_{s+1}. Then there some vertex with index greater than s+2 on the other side of A_{s+1} (e.g. A_n.) Let A_t and A_{t+1} be on opposite sides of A_{s+1}, with A_t to the right of A_{s+1}. A_{t+1} is not between A_s and A_{s+1} by lemma 9. It follows that A_{t+1} is to the left of A_s and hence to the left of A_1. But then A_t and A_{t+1} are on opposite sides of A_1, and t is greater than r. But if $r = 1$ there is no pair of vertices on opposite sides of A_1, and if $r \neq 1$, then r is the highest index such that A_r and A_{r+1} are on opposite sides of A_1. Hence there is a contradiction. Q.E.D.

We will now recursively define the order of a semi-inducing path for a hidden variable on a minimal d-connecting path. If U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, and W is a hidden variable on U such that the semi-inducing path for W on U contains no hidden variables other than W, then W is a 0-order hidden variable on U. If U is a minimal d-connecting path between X and Y given $\mathbf{Z}$, and W is a hidden variable on U such that the maximum order of any other hidden variable on the semi-inducing path for W on U is n-1, then W is an n^{th}-order hidden variable on U.

Lemma 10 guarantees that this recursive definition is sound, because it guarantees that if U is a minimal d-connecting path between X and Y given $\mathbf{Z}$ that contains hidden variables, then there is a 0 order hidden variable on U and also that the definition of the order of any hidden variable W on U is not defined in terms of the order of W.

From the definition of partially oriented inducing path graph, it follows that if π is a partially oriented inducing path graph for a directed acyclic graph G over $\mathbf{O}$, and G' is the inducing path graph for G over $\mathbf{O}$, then if a path U is a semi-inducing path for W in π, then it is a semi-inducing path for W in G'. Let us call the **n^{th} stage of the POIPG Algorithm** the stage that occurs at the n^{th} iteration of the repeat loop in step C) of the POIPG Algorithm. Let us call the POIPG produced by the POIPG Algorithm with inputs G', $\mathbf{O}$, and **Sepset** the **POIPG Algorithm partially oriented inducing path graph** for G' over **O** using **Sepset**. (Including as inputs to inducing path algorithm both G and G' (the inducing path graph for G over **O**) is somewhat redundant. Hence we will also speak of the POIPG Algorithm partially oriented inducing path graph for G over **O** using **Sepset**.)

Lemma 11: If G' is an inducing path graph, U is a minimal d-connecting path between X and Y given **Z** in G', $U(A,B)$ is a semi-inducing path for R in G', π is the POIPG Algorithm partially oriented inducing path graph for G' over **O** using **Sepset**, π_n is the partially oriented inducing path graph for G' constructed at the n^{th} stage of the POIPG Algorithm, $U_n'(A,B)$ is the path corresponding to $U(A,B)$ in π_n, $U'(A,B)$ is the path corresponding to $U(A,B)$ in π, and if there exists an n such that every vertex on $U_n'(A,B)$ except for R is oriented as a collider, then $U'(A,B)$ is a semi-inducing path between A and B in π.

Proof. The proof is by induction.

Base Case: Let $<A,M,N>$ be the first three vertices on $U_n'(A,B)$ in π. M is a collider on $U'_n'(A,B)$ in π_n by hypothesis. $M \rightarrow B$ in G' because $U(A,B)$ is a semi-inducing path for R on U. Hence M is adjacent to B in π_n. A, M, and B are not in a triangle in π_n because A and B are not adjacent in G'. Hence the edge from M to B is oriented as $M \rightarrow B$ in π by the POIPG Algorithm.

Induction Case: Let S be the m^{th} vertex on $U_n'(A,B)$. Let the induction hypothesis be that for each vertex C between A and S, the edge between C and B has been oriented as $C \rightarrow B$. A is not adjacent to B, so it follows that the concatenation of $U_n'(A,S)$ and the edge between S and B is a semi-inducing path for S. Hence the POIPG algorithm orients the edge between S and B as $S \rightarrow B$.

It follows that $U'(A,B)$ is a semi-inducing path in π. Q.E.D.

In a POIPG π, let us say that B is a **definite non-collider** on an undirected path U if and only if either B is an endpoint of U or U contains a subpath $<A,B,C>$ and either A <- B *-* C, A *-* B -> C, or A *-$\underline{* B *}$-* C in π. In a POIPG π, if $X \neq Y$, and X and Y are not in **Z**, then an undirected path U between X and Y **definitely d-connects** X and Y given **Z** if and only if every collider on U has a descendant in **Z**, and every definite non-collider on U is not in **Z**. X and Y are **definitely d-connected given Z** in π if and only if some path definitely d-connects them. **X** and **Y** are **definitely d-connected given Z** if and only if some X in **X** and some Y in **Y** are definitely d-connected given **Z**.

Lemma 12: If G' is an inducing path graph over **O**, π is the POIPG Algorithm partially oriented inducing path graph for G' over **O** using **Sepset**, U is a minimal d-connecting path between X and Y given **Z** in G', $U(A,B)$ is a semi-inducing path on U for R in G', $U'(A,B)$ is the path corresponding to $U(A,B)$ in π, then $U'(A,B)$ is a semi-inducing path for R in π.

Proof. The proof is by induction on the order of the semi-inducing path. Let M be the semi-inducing path from a variable W on U, M_n' be the corresponding path in π_m, and M' the corresponding path in π.

Base Case: Suppose that W is a hidden variable on U, and M is zero order. Because M contains no hidden variables other than W, in π every vertex on M except W and the endpoints is oriented as a collider on M by the POIPG Algorithm. By lemma 11, M' is a semi-inducing path in π. Hence the POIPG Algorithm orients W as a collider or a definite non-collider on U'.

Induction Case: Suppose that W is a hidden variable on U, M is n^{th} order, and all of the hidden variables on M_n' except W of order n-1 or less have been oriented as colliders by the POIPG algorithm. It follows that all of the colliders on M_n' except for W are oriented. By lemma 11, M' is a semi-inducing path for W in π. Hence W is oriented as a collider or a definite non-collider on U' in π by the POIPG Algorithm. Because each of the vertices on U except for the endpoints and possibly R is a collider on U in G', each of the vertices on U' except for the endpoint and possibly R is a collider on U' in π. Hence by lemma 11, U' is a semi-inducing path for R in π. Q.E.D.

Lemma 13: If G' is an inducing path graph, π is the POIPG Algorithm partially oriented inducing path graph for G' over **O** using **Sepset**, and U is a minimal d-connecting path between X and Y given **Z** in G', then the corresponding path U' in π definitely d-connects X and Y given **Z**.

Proof. First we will show that every vertex on U' is a collider or a definite non-collider. If there are no hidden variables on U', this follows from step C) of the POIPG Algorithm. If there are hidden variables on U', then by lemma 12, for each hidden vertex C on U', the path in π corresponding to the semi-inducing path for C on U is a semi-inducing path for C

in π. Hence C can be oriented as a collider or a definite non-collider by the POIPG Algorithm. Hence every vertex on U' is either a collider or a definite non-collider.

Suppose now that some collider C on U has a descendant in $\mathbf{Z}$ in G'. Let V be the shortest path in G' from C to a variable in $\mathbf{Z}$. Suppose that there is a hidden vertex W on V, and the edge hiding W is between A and B, where A is before B on V. We will now prove by induction that for every vertex S between C and A inclusive on V, there is an edge S <-> B in G' by induction on the number of vertices preceding A on V.

Base Case: In G', the edge between A and B is not A -> B else V is not the shortest path from C to a member of $\mathbf{Z}$. The edge is not B -> A because G' is acyclic. Hence the edge is A <-> B.

Induction Case: Let T be the n^{th} vertex before A on V, and let the induction hypothesis be that there is a T <-> B edge in G'. Let S be the predecessor of T on V. There is an edge S -> T in G' because that edge is on V. Hence there is a path S -> T <-> B in G', and B is a descendant of T in G'. This is an inducing path between S and B in G', so there is an edge between S and B in G'. The edge is not S -> B, else V is not the shortest path from C to a member of $\mathbf{Z}$. The edge is not B -> S because G' is acyclic. It follows that the edge is S <-> B.

Let R be the vertex between X and C that is adjacent to C on U, and let S be the vertex between Y and C that is adjacent to C on U. There are three cases: either B is on $U(X,R)$, or B is on $U(S,Y)$, or B is on neither.

Suppose first that B is on $U(X,R)$. It follows that B is not on $U(S,Y)$ because U is acyclic. Let U_2 be the concatenation of $U(Y,C)$ and the C <-> B edge; U_2 is acyclic because B is not on $U(Y,C)$ and $U(Y,C)$ is acyclic. Because C is a collider on U, there is an S *-> C edge on U, and hence on U_2. By lemma 6.6.2 there is a vertex F between Y and S inclusive such that the edge between F and B is substitutable in U_2 for $U_2(F,B)$, and the edge between F and B is into B. Let U' be the path that is the concatenation of $U_2(Y,F)$, the edge between F and B and $U(B,X)$. U' is acyclic because $U_2(Y,F)$, the edge between F and B and $U(B,X)$ are each acyclic, $U_2(Y,F)$ and $U(B,X)$ do not intersect, $U_2(Y,F)$ intersects the edge between F and B only at F, and the edge between F and B intersects $U(B,X)$ only at B. Because the edge between F and B is substitutable in U_2 for $U_2(F,B)$, F is a collider on U' if and only if it is a collider on U_2. It follows that F is a collider on U' if and only if it is a collider on U. If B is a collider on U and on U' or on neither U nor U', then U' d-connects X and Y given $\mathbf{Z}$, so U is not minimal. Suppose then that B is a collider on U' but not on U. Let H be the vertex adjacent to B on U and between B and C. Because B is a collider on U but not on U', it follows then B is not in $\mathbf{Z}$, but that there is an edge B *-> H on U, so B has a descendant who is a collider on U, and hence has a descendant in $\mathbf{Z}$. Hence U' d-connects X and Y given $\mathbf{Z}$. U' is shorter than U because it replaces the subpath $U(F,B)$ with a single edge between F and B. Hence U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$.

The case where B is on $U(S,Y)$ is analogous to the previous case.

Suppose then that B is not on $U(S,Y)$ or $U(X,R)$. Let U_1 be the concatenation of $U(X,C)$ and the C <-> B edge. U_1 is acyclic because $U(X,C)$ is acyclic and B is not on $U(X,R)$. Because C is a collider on U, there is an R *-> C edge on U, and hence on U_1. By lemma 6.6.2 there is a vertex E between X and R inclusive such that the edge between E and B is substitutable in U_1 for $U_1(E,B)$, and the edge between E and B is into B. Let U_1' be the path that is the concatenation of $U_1(X,E)$ and the edge between E and B. Because the edge between E and B is substitutable in U_1 for $U_1(E,B)$, E is a collider on U_1' if and only if it is a collider on U_1. It follows that E is a collider on U_1' if and only if it is a collider on U.

Similarly, let U_2 be the concatenation of $U(Y,C)$ and the C <-> B edge. U_2 is acyclic because $U(Y,C)$ is acyclic and B is not on $U(Y,S)$. Because C is a collider on U, there is an S *-> C edge on U, and hence on U_2. By lemma 6.6.2 there is a vertex F between Y and S inclusive such that the edge between F and B is substitutable in U_2 for $U_2(F,B)$, and the edge between F and B is into B. Let U_2' be the path that is the concatenation of $U_2(Y,F)$ and the edge between F and B. Because the edge between F and B is substitutable in U_2 for $U_2(F,B)$, F is a collider on U_2' if and only if it is a collider on U_2. It follows that F is a collider on U_2' if and only if it is a collider on U.

Let U' be the concatenation of $U(X,E)$, the edge between E and B, the edge between B and F, and $U(F,Y)$. $U(X,E)$ does not intersect the edge between E and B except at E, it does not intersect the edge between B and F, and it does not intersect $U(F,Y)$. The edge between E and B intersects the edge between B and F only at B, and it does not intersect $U(F,Y)$. The edge between B and F intersects $U(F,Y)$ only at F. Hence U' is acyclic. It is no longer than U, because it replaces $U(E,C)$ with the single edge between E and B, and $U(C,F)$ with the single edge between B and F. With the possible exception of B, every vertex on U' is a collider on U' if and only if it is a collider on U. B is a collider on U', and B has a descendant in $\mathbf{Z}$. Moreover, the shortest path from B to a member of $\mathbf{Z}$ is shorter than the shortest path from C to a member of $\mathbf{Z}$, because B is after C on V. Hence U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$.

In any of these cases it follows from the assumption that there is a hidden vertex on V that U is not a minimal d-connecting path between X and Y given $\mathbf{Z}$; hence there is no hidden vertex on V. Let V' be the path corresponding to V in π. C is a collider on U' in π, and because there is no hidden variable on V in G', there is no hidden variable on V' in π, and V' is a directed path in π.

Hence U definitely d-connects X and Y given $\mathbf{Z}$ in π. Q.E.D.

Theorem 4: If G is a directed acyclic graph, and π is the POIPG Algorithm partially oriented inducing path graph for G over $\mathbf{O}$ using **Sepset**, for any set of vertices $\mathbf{Z} \cup \{A,B\}$ included in $\mathbf{O}$, A and B are d-connected given $\mathbf{Z}$ in G if and only if they are definitely d-connected given $\mathbf{Z}$ in π.

Proof. Because π is a partially oriented inducing path graph for G over $\mathbf{O}$, it follows trivially that A and B are d-connected given $\mathbf{Z}$ in G if they are definitely d-connected given $\mathbf{Z}$ in π. Lemma 13 entails that A and B are d-connected given $\mathbf{Z}$ in G only if they are definitely d-connected given $\mathbf{Z}$ in π. Q.E.D.

Corollary 1: If G is a directed acyclic graph, and π is the FCI partially oriented inducing path graph for G over $\mathbf{O}$, for any set of vertices $\mathbf{Z} \cup \{A,B\}$ included in $\mathbf{O}$, A and B are d-connected given $\mathbf{Z}$ in G if and only if they are definitely d-connected given $\mathbf{Z}$ in π.

Proof. The output of the FCI algorithm is a partially oriented inducing path graph for G over $\mathbf{O}$ so A and B are d-connected given $\mathbf{Z}$ in G if they are definitely d-connected given $\mathbf{Z}$ in π. The POIPG Algorithm partially oriented inducing path graph for G over $\mathbf{O}$ orients a subset of the edges that the FCI Algorithm orients, so it follow from lemma 13 that A and B are d-connected given $\mathbf{Z}$ in G only if they are definitely d-connected given $\mathbf{Z}$ in π.

Theorem 5: The Faithful Entailment Equivalence Algorithm is correct.

Proof. The Faithful Entailment Equivalence Algorithm is correct if when G_1' and G_2' are inducing path graphs over $\mathbf{O}$ of directed acyclic graphs G_1 and G_2, then G_1' and G_2' have the same d-connection relations if and only if the POIPG partially oriented inducing path graphs for G_1' and G_2' using **Sepset** are the same.

Suppose first that G_1' and G_2' have the same d-connection relations. It follows that they have the same adjacencies. Hence step A) of the algorithm yields the same results for π_1 and π_2. Furthermore, **Sepset**[A,B] was constructed so that if A and B are not adjacent in G_1', A and B are d-separated by **Sepset**[A,B] in G_1'. Hence for any two non-adjacent vertices A and B in G_2', **Sepset**[A,B] d-separates A and B in G_2' because **Sepset**[A,B] d-separates A and B in G_1'.

After step A), the rest of the algorithm depends only upon **Sepset**, which is the same for both calls to the algorithm. It follows that G_1' and G_2' have the same POIPG Algorithm partially oriented inducing path graphs over **O** using **Sepset**.

By lemma 13, if G_1' and G_2' have the same POIPG Algorithm partially oriented inducing path graphs over O using **Sepset** they have the same d-connection relations. Q.E.D.

Theorem 6: If the inputs to the Faithful Entailment Equivalence Algorithm are directed acyclic graphs G_1, G_2, and subset of vertices **O**, and the graph with the largest number of vertices has n vertices, and **O** has m vertices, then the Faithful Entailment Equivalence Algorithm is $O(m^8 + n^5)$.

Proof. This follows from the fact that the two dominating steps in the Faithful Entailment Equivalence Algorithm are the Inducing Path Graph Algorithm which is $O(n^5)$, and the POIPG Algorithm, which is $O(m^8)$.

References

Aho, A., Hopcroft, J., Ullman, J. (1974). The Design and Analysis of Computer Algorithms. Addison-Wesley, Massachusetts.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems. Morgan and Kaufman, San Mateo.

Spirtes P., Glymour C., Scheines, R. (1990). Causality from probability. Conference Proceedings: Advanced Computing for the Social Sciences, Williamsburgh, VA.

Spirtes, P., Glymour, C., and Scheines, R. (1991). An algorithm for fast recovery of sparse causal graphs. Social Science Computer Review, vol. 9, 1991, pp. 62-72.

Spirtes, P. (1992). Building causal graphs from statistical data in the presence of latent variables", forthcoming in Proceedings of the IX International Congress on Logic, Methodology, and the Philosophy of Science, B. Skyrms, ed., Uppsala, Sweden, 1991.

Verma, T. and Pearl, J. (1990). Equivalence and synthesis of causal models. Proc. Sixth Conference on Uncertainty in AI. Association for Uncertainty in AI, Inc., Mountain View, CA, 220-227.

Verma, T. and Pearl, J. (1991). Equivalence and synthesis of causal models. Technical Report R-150, Cognitive Systems Laboratory, University of California, Los Angeles.