TECHNICAL REPORT
R-185
October 1992

A simple algorithm to construct a consistent extension of a partially oriented graph

Dorit Dor
Computer Science Department
School of Mathematical Sciences
Tel-Aviv University
69978
Israel

Michael Tarsi *
Cognitive Systems Laboratory
Computer Science Department
University of California, Los Angeles, CA 90024

October 23, 1992

Abstract

A *Partially directed acyclic graph, (pdag)*, is a graph which contains both directed and undirected edges, with no directed cycle in its directed subgraph. An oriented extension of a pdag G is a fully directed acyclic graph (dag) on the same underlying set of edges, with the same orientation on the directed subgraph of G and the same set of *vee-structures*. A vee-structure is formed by two edges, directed toward a common head, while their tails are nonadjacent. A simple polynomial-time algorithm is presented, to solve the following problem: *Given a pdag, does it admit an oriented extension?* The problem was stated by Verma and Pearl, while studying the existence of causal explanation to a given set of observed independencies.

---

*Visiting professor. Permanent address: Computer Science Department, School of Mathematical Sciences, Tel-Aviv University, 69978, Israel. Supported in part by the Air Force Office, AFOSR 90 0136 and by Northrop MICRO, PO 11510.

1 Introduction

A *Partially directed acyclic graph, (pdag)*, is a graph which contains both directed and undirected edges, with no directed cycle in its directed subgraph. An oriented extension of a pdag G is a fully directed acyclic graph (dag) on the same underlying set of edges, with the same orientation on the directed subgraph of G and the same set of *vee-structures*. A vee-structure is formed by two edges, directed toward a common head, while their tails are nonadjacent. These definitions as well as some background and motivation are stated and explained in [1]. While studying the existence of causal explanation to a given set of observed independencies, Verma and Pearl [1] have faced the following combinatorial problem, to which we refer here as "*PDX*" (PDag eXtensibility): *Given a pdag, does it admit an oriented extension?*

In Section 2 of [1] the authors present an algorithm for *PDX*, which is conjectured, however not proven, to be polynomial. Another algorithm, given by Verma in [3], although it runs in linear time, is rather complicated and less intuitive. We present here a simple polynomial-time algorithm to solve the above problem.

2 The algorithm

Our algorithm selects first a vertex x to be the sink of the extension and recursively proceed to the subgraph obtained by the removal of the sink and all edges incident to it:

```
Algorithm extend(G: pdag);
  begin (extend)
    G' := G; A := G;

      while A is not empty do
      begin (iteration)

          Select a vertex x which satisfies the following properties
        in the subgraph A:
        a. x is a sink (no edge (x,y) in A is directed outward from x)
        b. For every vertex y, adjacent to x, with (x,y) undirected, y is
        adjacent to all the other vertices which are adjacent to x;

          If such x is not found, then the algorithm stops and returns
        a negative answer (G does not admit any extension);
        If x is found, let all the edges which are incident to x in A
        be directed toward x in G' (G' is meant to form the output);
        A := A - x (remove x and all the edges incident to x)
      end (iteration);
    return G' (an extension of the input pdag G)
  end (extend).
```

3 Validity and complexity

An extension of G, if it exists, is a dag and as such it contains a sink. To become a sink of the extension G' a vertex x must satisfy property a. (of the iteration phase above) in G. To avoid the creation of new vee-structures, while directing all edges toward x, it should also satisfy property b.. Hence a vertex which satisfies both properties a. and b. is indeed necessary for the existence of an extension. To justify our recursive method we should show first that the removal of a sink x from the extension G' provides an extension $G' - x$ of the pdag $G - x$, obtained when the same vertex is removed from the input pdag G: No directed cycle can be formed by the removal of x and hence $G' - x$ is still a dag. In the general case a new v-structure might be generated by the removal of edges if a single edge is deleted from a triangle, whose other two edges now form a v-structure. In the case on hand, however, all the edges incident to x are deleted, thus, the number of edges removed from any triangle is either 0 or 2. $G' - x$ is hence indeed an extension of $G - x$. To complete the proof we should notice that no existing extension is missed by selecting the specific vertex x to be a sink. If there exists an extension where x is not a sink then it will still remain an extension if the edges going out of x are reoriented toward x. All redirected edges are incident to x and since x is now a sink no directed cycle was formed. Also no new v-structures are created due to property b. of the selected vertex x.

For the complexity analysis note that there are $|V|$ iterations where every edge is searched at most twice (once for each endvertex). The time complexity is thus $O(|V||E|)$.

4 Some concluding remarks

A dag with no vee-structure is chordal (any orientation of a chordless cycle contains a vee-structure). Consequently, if G contains no vee-structure then its underlying graph should be chordal. A known characterization of chordal graphs states that all such graphs can be constructed, starting with an isolated vertex, by successive insertion of new vertices, each adjacent to a clique in the existing graph. When our algorithm is applied to a graph G with no oriented edges then property b. states that the neighbors of x form a clique. In this case G admits an extension if and only if it is chordal. Our algorithm is a natural generalization of a naive chordality test, based on the above characterization, to the case where v-structures are allowed, but no new ones should be formed. Chordality can be tested in linear time [2] and hence it takes linear time to test the existence of an extension where the input has no vee-structures. We believe that linear-time chordality algorithm can be modified to a general linear-time algorithm for *PDX*.

References

[1] J. Pearl & T. Verma, "An Algorithm for Deciding if a set of Observed Independencies a Causal Explanation," *Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence*, Stanford (1992, to appear).

[2] R.E. Tarjan & N. Yannakakis, "Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs ," *Siam J. Comput.*, 13 (1984), 566-579.

[3] T. Verma, "A linear-time algorithm for finding a consistent expansion of a partially oriented graph," *Technical Report R-180*, UCLA Cognitive Systems Laboratory, 1992.