Transitive Closure Algorithms for Causal Directed Graphs

Michael J. Gilpin
Department of Mathematical Sciences
Michigan Technological University
Houghton MI. 49931

Robert O. Shelton
Information Systems Directorate
NASA JSC

Abstract

A causal directed graph (CDG) is a finite directed graph with *and-gates* and *or-nodes* in which nodes indicate true or false conditions and where arcs indicate causality. The set of all nodes implied true by a set of conditions (nodes declared true) is called the transitive closure of that set. Theorems 3-5 evaluate the number of distinct transitive closures for common CDG's. We present linear-space, linear-time algorithms for solving three transitive closure problems on CDG's: 1) determine if a particular node is implied by a set of conditions, 2) find the transitive closure of a set of conditions, and 3) determine the minimal set of initial conditions for a given transitive closure of an acyclic CDG. Implicit in Problem 3 is that every transitive closure of an acyclic CDG is generated by a unique minimal set of initial conditions. This is proved in Theorem 6.

1 Introduction

Basic terminology for causal directed graphs (CDG's) is introduced in Section 2. In Section 3 conditions and transitive closures of CDG's are discussed. The function $t(G)$, which counts the transitive closures of a CDG is introduced, and theorems are proven which evaluate $t(G)$

JCMCC 25 (1997), pp. 23-32

for elementary CDG's. In Section 4 we develop algorithms with linear complexity and linear space requirements for solving the following three problems.

Problem 1: Given a CDG, determine if a particular condition is implied by an initial set of conditions.

Problem 2: Given a CDG, determine all conditions implied by an initial set of conditions.

Problem 3: Given a transitive closure of an acyclic CDG, determine the minimal set of initial conditions which cause that transitive closure.

Causal graphs are of interest to NASA where nodes indicate "health" (functionality) of system components. A typical CDG used by NASA may have 20,000 nodes, so linear complexity and linear space requirements for CDG algorithms are desirable. Fault trees are discussed in [1] and [2]. Both of these references contain extensive bibliographies to the literature of applications of digraphs to fault-tree analysis. Algorithms for finding transitive closures are discussed in [3], [4], [5] and [6].

CDG's are also of interest in logic where truth values are assigned to predicates formed by conjunctions (and-statements) and disjunctions (or-statements) of a given finite set of propositions. Here it is convenient to use addition to denote "or," and to use multiplication to denote "and." In this setting one asks questions such as: what is the truth value of $(A + B) \times (B + C) \times (C + D)$ given A, B and D are true? See Cook's Theorem [7] for a related, but much harder problem.

The authors thank Dennis Lawler who suggested Problem 2. Thanks are also due Steven Schnurer and Don Kreher for discussions and guidance to the literature of fault-trees during the development of this paper.

2 Causal Graphs

We define a causal directed graph (CDG) to be a finite directed graph (perhaps with "and-gates") in which nodes indicate true or false conditions and where arcs indicate causality.

If G is a CDG and the directed edge (a, b) is in $e(G)$, then if a is assigned the value TRUE, then b must also be assigned the value TRUE. CDGs also involve *and-gates* which play the role of conjunction. If all nodes connected to arcs leading into an and-gate are marked TRUE, then all nodes connected to arcs leading away from that and-gate are marked TRUE.

We use circles to denote nodes of a CDG and solid rectangles to denote and-gates. Figure 1A shows an example of a CDG having an and-gate. In Figure 1A the and-gate has incoming edges from nodes b and c and an outgoing edge to node e. Thus if b and c are marked TRUE, then node e is marked TRUE.

A *set of conditions* of a CDG is an assignment of the value TRUE to a subset of nodes of the graph. thus we denote the set of conditions of a CDG by the set A of nodes that have been assigned the value TRUE. Once we are given a set of conditions of a CDG, other nodes in the graph are marked TRUE according to the Rules of Inference.

In our algorithms (Section 4) we treat and-gates as nodes of CDG's. Thus in our algorithms and-gates are treated as nodes which are true if and only if all their immediate ancestors are true.

3 Transitive Closures

Terminology: Throughout G is a *directed* graph. We use the term *subgraph* to mean a vertex-induced subgraph. We say G is the *disjoint union* of the subgraphs $G_1, G_2, \ldots, G_n$ in case $v(G_1), v(G_2), \ldots, v(G_n)$ are pairwise disjoint sets and $v(G) = \bigcup_{i=1}^{n} v(G_i)$. For example, the graph in Figure 1B is the disjoint union of the subgraphs $\{a, b, c, d\}$ and $\{e, f\}$. The *transitive closure* of a set of conditions A is the set of all nodes of G implied TRUE by those conditions. We denote the transitive closure of a set of conditions A as $T(A)$. Note that $T(\emptyset) = \emptyset$, $T(G) = G$ and $T(T(A)) = T(A)$ for any $A \subset v(G)$. For any directed graph G we define $t(G)$ to be the number of different transitive closures of G. A set of nodes A of G is called a *minimal causal set* for G in case no proper subset of A has the same transitive closure as A.

Let a and b be nodes of G. We say a *influences* b and we write $a \rightarrow_i b$ in case there exists $A \subset v(G)$ with $b \in T(A)$ and $b \notin T(A - \{a\})$. For example, in Figure 1A, $b \rightarrow_i e$ since $e \in T(\{b, c\})$ but $e \notin T(\{c\})$. We say a directed graph has an *influential cycle* in case for some integer $n > 1$ there exist nodes $a_1, a_2, \ldots, a_n$ with $a_k \rightarrow_i a_{k+1}$ for $1 \leq k < n$, and $a_1 = a_n$. If G has no influential cycles, we say G is *influentially acyclic*. Note that $a \rightarrow_i b$ implies there exists a conventional directed path (possibly through and-gates) from a to b. The converse is not

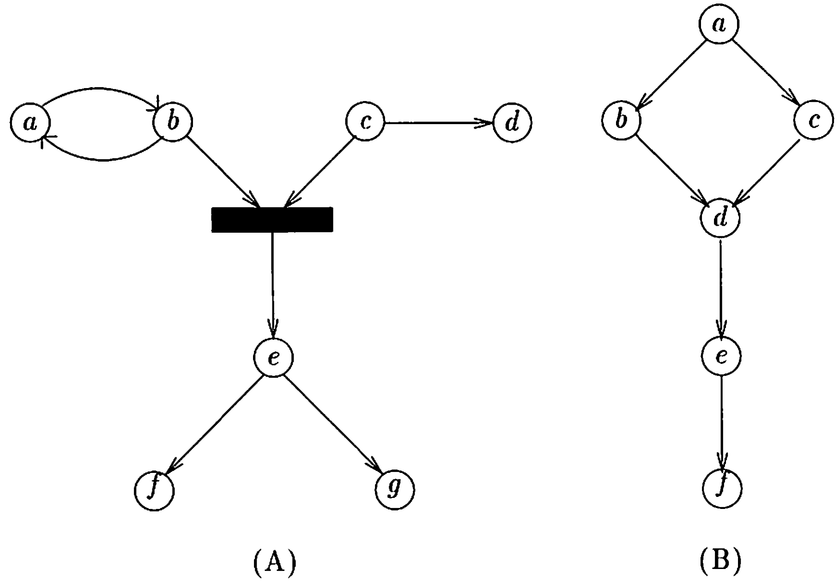

Figure 1

necessarily the case, i.e. there can be a conventional path from a to b, with $a \not\to_i b$, thus a graph containing conventional cycles can be influentially acyclic.

The disjoint subgraphs $G_1, G_2, \ldots, G_n$ of G are called *mutually independent* in case $i \neq j$ implies no node in $v(G_i)$ influences any node in $v(G_j)$. For example, the subgraphs $\{a, b\}$ and $\{c, d\}$ in Figure 1A are mutually independent. Let A and B be subgraphs of G. Then A is said to be *indifferent to B* in case no node in B influences a node in A.

Let $v(G)$ be the disjoint union of non-void subsets A and B. We say A *strongly implies B* and we write $A \Rightarrow_s B$ in case A is indifferent to B and whenever *any* node in A is TRUE then *every* node in B is TRUE. For example, in Figure 1B the subgraph $\{a, b, c, d\}$ strongly implies $\{e, f\}$. We say the subgraph A *weakly implies B* and we write $A \Rightarrow_w B$ in case A is indifferent to B, $B \subset T(A)$, and for any $A' \subset A$ and $B' \subset B$, $A \not\subset T(A')$ implies $T(A' \cup B') \cap B = T(B')$.

We state two lemmas which allow us to compute $t(G)$ for some elementary CDGs.

Lemma 1 *Let G be the disjoint union of the subgraphs A and B.*

i) *If $A \Rightarrow_s B$, then $t(G) = t(A) + t(B) - 1$, and*

ii) *if $A \Rightarrow_w B$, then $t(G) = (t(A) - 1)t(B) + 1$.*

Proof: Suppose $A \Rightarrow_s B$. There are $t(B)$ transitive closures of G in which no node of A is marked TRUE. There are $t(A) - 1$ transitive closures of G in which at least one node in A is marked TRUE. Summing the states of these mutually exclusive cases yields $t(G) = t(A) + t(B) - 1$ which proves i). Now, suppose $A \Rightarrow_w B$ with $v(G)$ the disjoint union of A and B. In the one case where every node of A is TRUE, then every node of B must be TRUE. The subgraph A has $t(A) - 1$ other transitive closures, each of which can correspond to any of the $t(B)$ transitive closures of B. Summing we obtain $t(G) = 1 + (t(A) - 1)t(B)$. □

Lemma 2 *Let G be the disjoint union of the mutually independent subgraphs $G_1, G_2, \ldots, G_n$. Then $t(G) = \prod_{i=1}^{n} t(G_i)$.*

Proof: Since the subgraphs $G_1, G_2, \ldots, G_n$ are mutually independent, any transitive closure of G is the union of transitive closures of the subgraphs $G_1, G_2, \ldots, G_n$. Thus, transitive closures of G occur in $\prod_{i=1}^{n} t(G_i)$ ways. □

As an immediate consequence of Lemmas 1 and 2 we have:

Theorem 3 *Let G be the disjoint union of A and B where A is the disjoint union of the mutually independent sets $A_1, A_2, \ldots, A_n$ and where B is the disjoint union of the mutually independent sets $B_1, B_2, \ldots, B_m$.*

i) *If $A \Rightarrow_s B$, then $t(G) = \prod_{i=1}^{n} t(A_i) + \prod_{j=1}^{m} t(A_j) - 1$, and*

ii) *if $A \Rightarrow_w B$, then $t(G) = 1 + (\prod_{i=1}^{n} t(A_i) - 1) \times (\prod_{j=1}^{m} t(B_j)$*

We define two terms before stating our next result. By a *directed interval of length n* we mean a directed graph with nodes $v_1, v_2, \ldots, v_n$ and with edges (v_i, v_{i+1}) for $1 \leq i < n$. A *lattice of r rows and c columns* is a rectangular grid of r rows and c columns of nodes $\nu_{i,j}$ in the plane with edges of the form

$$(\nu_{i,j}, \nu_{i+1,j}) \text{ and } (\nu_{i,j}, \nu_{i,j+1})$$

with all arcs leading upwards or from left-to-right.

Theorem 4 *Let G be a CDG:*

i) *if G is a cycle, then* $t(G) = 2$,

ii) *if G is a directed interval of length n, then* $t(G) = n + 1$,

iii) *if G is a lattice of r rows and c columns, then* $t(G) = \binom{r+c}{r}$, *and*

iv) *if G is a full, regular, k-ary tree of depth n, then* $t(G) = \phi_k^n(2)$, *where* $\phi_k(x) = 1 + x^k$.

Proof: i) is trivial, and ii) is a special case of iii).

To prove iii) let G be a directed lattice with r rows and c columns. Then any transitive closure T of G is completely determined by the count of marked nodes in the rows of G. For T a transitive closure, let x_i denote the number of marked nodes in the i^{th} row of G. Then $0 \leq x_1 \leq x_2 \leq \ldots \leq x_r \leq c$. For $0 \leq j \leq c$, let b_j denote the number of terms of $\{x_i\}_{i=1}^r$ equal to j. Then $0 \leq b_j$ for $0 \leq j \leq c$, and $\{b_j\}_{j=0}^c$ completely determines $\{x_i\}_{i=1}^r$, so completely determines T. But $\sum_{j=0}^c b_j = r$, so $t(G) = \binom{r+c}{r}$.

To prove iv) we let T_n denote the full k-nary tree of depth n. Then T_0 consists of only a root node, and $t(T_0) = 2$. For $n > 0$, Theorem 3 i) implies the recurrence $t(T_n) = 1 + (t(T_{n-1}))^k$. Thus, $t(T_n) = (\phi_k)^n(2)$ where the 0^{th} power of ϕ_k is the identity function. □

Theorem 5 *Let G be a CDG having no and-gates. Then* $t(G) = t(G')$ *where* G' *is the reversal graph* $v(G') = v(G)$ *and* $(a,b) \in v(G')$ *iff* $(b,a) \in v(G)$.

Proof: Let G be a CDG and let V be any transitive closure of G.

Let $x \in V$ and let $y \in \hat{V} = G - V$. Then there is no path from x to y, so there is no path in the reversal graph G' from y to x. So, $\hat{V}$ is a transitive closure of G'. Likewise, if W is a transitive closure of G', then $\hat{W} = G - W$ is a transitive closure of G. This establishes a bijection between transitive closures of G and transitive closures of G', which proves the theorem. □

We now apply our results to compute $t(G)$ for the graphs in Figure 1. Let G be the graph in Figure 1A and let $A_1 = \{a, b\}, A_2 = \{c, d\}$

and $B = \{e, f, g\}$. Then A_1 is a 2-cycle, A_2 is a directed interval of length 2, and B is a binary tree of depth 1. By Theorem 4, $t(A_1) = 2$, $t(A_2) = 3$ and $t(B) = 1 + 2^2 = 5$. Note $A_1 \cup A_2 \Rightarrow_w B$. So, by Theorem 3, $t(G) = (2 \times 3 - 1) \times 5 + 1 = 26$.

Now let G be the graph in Figure 1B and let $A = \{a, b, c, d\}$ and $B = \{e, f\}$. By Theorem 4, $t(A) = \binom{4}{2}$ and $t(B) = 3$. Since $A \Rightarrow_s B$ it follows from Lemma 1ii that $t(G) = 6 + 3 - 1 = 8$.

4 Transitive Closure Algorithms

The algorithms presented here solve Problems 1-3 by breadth-first searches with modifications that account for and-gates. Each algorithm is linear, both in complexity and space requirements, in the number of arcs of the graph. Our algorithms employ an abstract data structure which we call *dual stacks*. The ADT dual stacks has the following operations:

- Initialize: Defines *New_stack* and *Old_stack* and sets stack pointers to initial values
- Push_new_stack, Push_old_stack
- Pop_new_stack, Pop_old_stack
- Clear_new_stack, Clear_old_stack
- Swap_stacks
- New_stack_size, Old_stack_size

In writing code for CDGs it is convenient to view and-gates as nodes of the graph. And-gates are assigned an In_degree equal to the number of their immediate ancestors, whereas nodes are assigned an In_degree of 1. When a node (or and-gate) is marked TRUE then the In_degree of each of that node's children is decremented. So, after assigning the initial set of conditions, nodes are marked TRUE whenever their In_degree is less than 1.

Pseudo-code for obtaining the transitive closure of a set of conditions (Problem 2) is as follows:

I. Read in CDG. This process results in the following:

1. An array Exit_list that contains the out-arcs from each node.
2. Arrays In_degree and Out_degree.
3. Designation of And_gates.
4. Initialized attribute arrays:
   Visited, And_gates and Initial_node.
5. Nodes of initial condition set pushed onto New_stack.

II. Call a recursive subroutine (ALL_CAUSED), pseudo-code as follows:

```
ALL_CAUSED (void)
Swap_stacks
Clear_new_stack
while Old_stack_size > 0
  BEGIN
    NODE = Pop_old_stack
    for each CHILD of NODE
      BEGIN
        if not Initial_node(CHILD) and not Visited(CHILD)
          BEGIN
            if And_gate(CHILD)
              BEGIN
                decrement In_degree(CHILD)
                if In_degree(CHILD) equals 0
                  mark CHILD as Visited
                  push CHILD on New_stack
              end if And_gate
            else
              mark CHILD as Visited
              push CHILD on New_stack
            end if not Visited
      end for each CHILD
  end while
```

```
 if New_stack_size > 0
    call All_CAUSED
 else return
```

```
III. Print transitive closure (Initial and Visited nodes)
```

Algorithms for determining if a target node is in the transitive closure of an initial set for CDG's (Problem 1) and for determining the minimal causal set for acyclic CDG's (Problem 3) are obtained with minor changes. C language source code and QuickBASIC source code for Problem 1 and example data are available by anonymous ftp from Michigan Technological University on the host math.mtu.edu (141.219.151.128)in the directory /pub/cdg.

We conclude by showing that minimal causal sets for influentially acyclic CDG's are unique.

Theorem 6 *If G is an influentially acyclic CDG, then each transitive closure of G is generated by a unique minimal causal set.*

Proof: Let V be any transitive closure in G and suppose that A and B are distinct minimal causal sets for V. Let $\hat{A}$ be $A - B$ and $\hat{B}$ be $B - A$. Because A and B are assumed to be minimal and distinct, neither $\hat{A}$ nor $\hat{B}$ is empty. Let a_0 be any node in $\hat{A}$. Since $a_0 \in V = T(B)$, it follows that $a_0 \in T(\beta)$ for a non-void collection of subsets β of B. Among all such subsets of B, let β_0 be minimal. Observe that at least one node $b_0 \in \beta_0$ must belong to $\hat{B}$, for otherwise a_0 would be in the transitive closure of a set of other nodes in A, and thus a_0 would not be part of the minimal causal set A. Since $a_0 \in T(\beta_0)$ and since β_0 is minimal with respect to this property, there must exist $b_0 \in \beta_0$ with $a_0 \notin T(\beta_0 - \{b_0\})$. Thus $b_0 \rightarrow_i a_0$. Likewise, there exists $a_1 \in \hat{A}$ with $a_1 \rightarrow_i b_0$.

Repetition of the argument produces a sequence of nodes a_0, b_0, a_1, $b_1, \ldots$ in G where each term in the sequence is influentially implied by the next. Since G is finite there must exist two terms of the sequence that are the same. Thus, G contains an influential cycle. This contradiction completes the proof. □

Note that Theorem 6 also holds if acyclic CDG's are defined in the usual sense, but with the stipulation that paths may include edges passing through and-gates. With either interpretation of acyclic graphs, the proof of the theorem also shows that for acyclic CDG's, the minimal causal set is the set of *orphans* of the transitive closure where orphans are defined to be nodes with no ancestors within the transitive closure.

References

[1] D. L. Iverson and F. A. Patterson-Hine, Object-Oriented Fault Tree Models Applied to System Diagnosis, *Proceedings of the SPIE Applications of Artificial Intelligence VII*, Orlando, FL, (April 19, 1990).

[2] D. L. Iverson, Automatic Translation of Digraph to Fault-Tree Models, *Proceedings of the 38-th Annual Reliability and Maintainability Symposium*, Las Vegas, Nevada, (January 21-22, 1992), ©IEEE 1992.

[3] S. Warshall, A Theorem on Boolean Matrices, *J. Assoc Computing Machinery*, **9** (1963), 11-12.

[4] H. S. Warren, A Modification of Warshall's Algorithm for the Transitive Closure of Binary Relations, *Comm. ACM*, **18** (1975), 218-220.

[5] I. J. Sacks, Digraph Matrix Analysis, *IEEE Transactions on Reliability*, **R-34** (1985), 437-446.

[6] R. W. Stevenson, J. G. Miller, and M. Austin, "Failure Environment Analysis Tool (FEAT) Development Status", *American Institute of Aeronautics and Astronautics*, ©1991 by R. W. Stevenson, et al.

[7] S. A. Cook, The Complexity of Theorem Proving Procedures, *Proceedings of the 3rd Annual ACM Symp. on Theory of Computing*, New York, (1982), 231-233.