Identification for Tree-Shaped Structural Causal Models in Polynomial Time

Aaryan Gupta[1], **Markus Bläser**[2]

[1]Indian Institute of Technology Bombay, Mumbai, India
[2]Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
aaryanguptaxyz@gmail.com, mblaeser@cs.uni-saarland.de

Abstract

Linear structural causal models (SCMs) are used to express and analyse the relationships between random variables. Direct causal effects are represented as directed edges and confounding factors as bidirected edges. Identifying the causal parameters from correlations between the nodes is an open problem in artificial intelligence. In this paper, we study SCMs whose directed component forms a tree. Van der Zander et al. give a PSPACE-algorithm for the identification problem in this case, which is a significant improvement over the general Gröbner basis approach, which has doubly-exponential time complexity in the number of structural parameters. However, they do not show that their algorithm is complete. In this work, we present a randomized polynomial-time algorithm, which solves the identification problem for tree-shaped SCMs. For every structural parameter, our algorithms decides whether it is generically identifiable, generically 2-identifiable, or generically unidentifiable. (No other cases can occur.) In the first two cases, it provides one or two fractional affine square root terms of polynomials (FASTPs) for the corresponding parameter, respectively. In particular, our algorithm is not only polynomial time, but also complete for for tree-shaped SCMs.

Introduction

Linear structural causal models (SCMs) model relationships between random variables (Bollen 1989; Duncan 1975). We are given random variables $V_0, V_1, \ldots, V_n$, which depend linearly on the other variables and an additional error term ϵ_i, that is, we can write $V_i = \sum_j \lambda_{j,i} V_j + \epsilon_i$. The error terms are normally distributed with zero mean and covariances between the terms given by some matrix $\Omega = (\omega_{i,j})$. We consider only *recursive models*, i.e., for all $j > i$, we have $\lambda_{j,i} = 0$. Such a model can be represented by a so-called mixed graph: The nodes $0, \ldots, n$ correspond to the variables $V_0, \ldots, V_n$. Directed edges represent a linear influence $\lambda_{j,i}$ of a parent node j on its child i. Bidirected edges represent an additional correlation $\omega_{i,j} \neq 0$ between the error terms.

Figure 1 shows a classical example of an SCM M_1. A change of V_0 by 1 implies a change of $\lambda_{0,1}$ of V_1 and a change of $\lambda_{01}\lambda_{12}$ of V_2. The covariance σ_{01} between V_0 and V_1 is thus λ_{01} and between V_0 and V_2, it is $\lambda_{01}\lambda_{12}$.

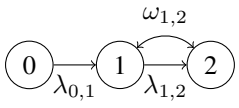

Figure 1: SCM M_1, a classical IV example

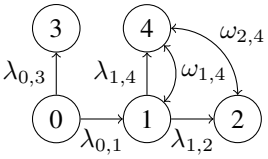

Figure 2: Tree-shaped SCM

More general, we write the coefficients of all directed edges as a matrix $\Lambda = (\lambda_{i,j})$ and the coefficients of all bidirected edges as $\Omega = (\omega_{i,j})$. Then the matrix $\Sigma = (\sigma_{i,j})$ of covariances between the variables V_i and V_j is given by

$$\Sigma = (I - \Lambda)^{-1}\Omega(I - \Lambda)^{-T}, \tag{1}$$

see e.g. Drton (2018). The reverse problem, the identification of causal effects, is a major open problem. Given the mixed graph and the matrix of covariances Σ, calculate the matrix Λ. (By (1), this also identifies Ω.) We here deal with what is called the *identification problem*: Is the number of solutions for a particular $\lambda_{i,j}$ finite and if yes, provide corresponding symbolic expressions for $\lambda_{i,j}$. If there is exactly one solution for $\lambda_{i,j}$, we call $\lambda_{i,j}$ *identifiable*, if there are k solutions, we call it k-identifiable, and if there is an infinite number of solutions, we call it unidentifiable.

There has been a considerable amount of research on identification in linear SCMs in economics, statistics, and artificial intelligence. Of particular interest here is the pioneering work by Pearl (2009) on the computational aspects of identification and subsequent works. Most approaches are based on *instrumental variables* (Bowden and Turkington 1990). In Figure 1, one can calculate $\lambda_{1,2} = \frac{\lambda_{0,1}\lambda_{1,2}}{\lambda_{0,1}} = \frac{\sigma_{0,2}}{\sigma_{0,1}}$. V_0 is called an instrumental variable (IV) in this case. Since IVs are not complete, that is, not all identifiable parameters can be identified with IVs, more complex generalizations of IVs have been developed, for instance conditional instrumental variables (Bowden and Turkington 1990; Pearl 2001; van der Zander, Textor, and Liśkiewicz 2015), instrumental sets (Brito and Pearl 2002a; Brito 2010; Brito and Pearl 2002b; van der Zander and Liśkiewicz 2016), half-treks (Foygel, Draisma, and Drton 2012), auxiliary instrumental variables (Chen, Pearl, and Bareinboim 2015), determinantal instrumental variables (Weihs et al. 2018), instrumental cutsets (Kumor, Chen, and Bareinboim 2019), or

auxiliary cutsets (Kumor, Cinelli, and Bareinboim 2020).

A major drawback of all criteria mentioned above is that they are not complete, that is, there are examples where they fail to identify a parameter that in principle is identifiable. An alternative approach is to solve the system of polynomial equations (1) using Gröbner bases (García-Puente, Spielvogel, and Sullivant 2010). By the properties of Gröbner bases, this approach is complete, that is, it identifies all parameters that are identifiable. However, Gröbner base algorithms typically have a doubly exponential running time and are often too slow to be used in practice once the SCMs get larger.

Our Results

Since it seems to be difficult to find criteria that are complete in general and can be decided in polynomial time, we restrict the underlying graph class. Van der Zander et al. (2022) consider tree-shaped SCMs $M = (V, D, B)$. Here, the graph (V, D) of directed edges forms a directed tree with root 0. Figure 2 shows an example, but also Figure 1 is one with the tree being a path. Van der Zander et al. (2022) propose an algorithm for tree-shaped SCMs called TreeID, which works by algebraically solving systems of equations. They prove that their algorithm is a PSPACE algorithm (with potentially exponential running time), a vast improvement over the general Gröbner basis approach. However, they do not show that their algorithm is complete.

In this work, we provide an algorithm which solves the identification problem in tree-shaped SCMs in randomized polynomial time. Hence the problem can be solved in BPP. For each variable $\lambda_{i,j}$, we can decide in randomized polynomial time whether it is generically identifiable, 2-identifiable, or unidentifiable. No other cases can occur. In the first two cases, we present one or two rational expressions that might contain a square root (a so-called FASTP). Generically here means that the denominators are non-zero expressions but can become zero when plugging in concrete values. Generic identifiability is also called almost everywhere identifiability, see e.g. García-Puente, Spielvogel, and Sullivant (2010); Foygel, Draisma, and Drton (2012).

Every missing bidirected edge in M gives rise to a bilinear equation. Van der Zander et al. (2022) show that if we have a cycle of missing edges, then one can potentially identify or 2-identify the parameters of the directed edges entering the missing cycle. (Note that as the directed edges form a tree, for every node i in V, there is exactly one directed edge with head i. We will often relate the parameter $\lambda_{p,i}$ of the unique edge entering i to i itself and say that i is identifiable.) However, one can be unlucky, and the missing cycle does not identify any parameter. Therefore, van der Zander et al. (2022) need to enumerate all cycles of missing edges, which can be exponentially many.

As our first contribution, we present an algorithm which can find an identifying missing cycle in randomized polynomial time, avoiding the need to enumerate all cycles. This is crucial to achieve polynomial running time. Second, we relate the concept of missing cycles to the theory of resultants. The crucial point here is that all equations that arise from the missing edges have only two variables, which is essentially the theory of plane algebraic curves, which is well understood. This allows us to show the completeness of our algorithm for tree-shaped models. Thus, tree-shaped SCMs are a class of graphs with an identification algorithm that is complete and has polynomial running time.

Due to space limitations, some proofs are omitted, which is indicated by a "$\triangleright$" at the end of a lemma or theorem. Missing proofs can be found in the full version (Gupta and Bläser 2023). A "$\square$" at the end means that the proof has been given in the text before the lemma or theorem.

Preliminaries

We consider *mixed graphs* $M = (V, D, B)$ with $n + 1$ nodes $V = \{0, 1, \dots, n\}$. With each node we associate a random variable V_i. D is the set of directed edges. We denote directed edges either by (i, j) or $i \to j$. B is the set of bidirected edges, which we denote by $\{i, j\}$ or $i \leftrightarrow j$. We only consider the case when M is acyclic, that is, (V, D) is a directed acyclic graph. With each directed edge $i \to j$, we associate a variable $\lambda_{i,j}$ and with every bidirected edge $u \leftrightarrow v$ a variable $\omega_{u,v}$. Let Λ and Ω be the corresponding $(n + 1) \times (n + 1)$-matrices. Since M is acyclic, we can always assume that Λ is upper triangular. The matrix Ω is symmetric. Let $\sigma_{i,j}$ be the covariance between the random variables corresponding to nodes i and j and Σ be the corresponding matrix. It is well known that the $\sigma_{i,j}$ are polynomials in the parameters $\lambda_{i,j}$ and $\omega_{i,j}$, see (1). Since the $\sigma_{i,j}$ are polynomials in the $\lambda_{i,j}$ and $\omega_{i,j}$, they are not independent variables but might fulfill polynomial relations, which we will explore.

For a given mixed graph M, the *identification problem* asks to express the parameters $\lambda_{i,j}$ in terms of the $\sigma_{i,j}$. We consider the generic version of the problem, that is, the expressions that we provide for the $\lambda_{i,j}$ have only to be defined on an open set. A parameter $\lambda_{i,j}$ is generically identifiable if there is only one solution for $\lambda_{i,j}$, k-identifiable, if there are k different solutions, and unidentifiable, if there is an infinite number of solutions.

An important tool is Wright's trek rule (Wright 1921, 1934): A *trek* in M between two nodes i and j is a path (with edges from D and B) such that no two arrow heads collide, that is, it is of the form $i \leftarrow i_1 \leftarrow \cdots \leftarrow u \leftrightarrow v \to \cdots \to j_1 \to j$ or $i \leftarrow i_1 \leftarrow \cdots \leftarrow u \to \cdots \to j_1 \to j$. The part containing the i-nodes is the left part of the trek, the part containing the j-nodes is the right part of the trek. With each trek τ with associate a monomial $M(\tau)$, which is the product of the edge labels. In the second case, we also multiply the monomial with $\omega_{u,u}$. Wright's rule says that the covariances satisfy $\sigma_{i,j} = \sum_\tau M(\tau)$, where the sum is taken over all treks between i and j.

Results by Van der Zander et al.

For a mixed graph $M = (V, D, B)$, the *missing edge graph* is the undirected graph $M_{\text{miss}} = (V, \bar{B})$. (Here, $\bar{B}$ denotes the complement of B.) An edge e of M_{miss}, i.e., $e \notin B$, is called a *missing edge*. Foygel, Draisma, and Drton (2012) show that the missing edges are crucial for identification. More precisely, in the case of tree-shaped mixed graphs with $0 \in V$ being the root, van der Zander et al. (2022, Lemma

5) prove that a parameter $\lambda_{x,y}$ is generically identifiable (k-identifiable) iff the system of equations

$$\lambda_{p,i}\lambda_{q,j}\sigma_{p,q} - \lambda_{p,i}\sigma_{p,j} - \lambda_{q,j}\sigma_{i,q} + \sigma_{i,j} = 0 \quad (2)$$
$$\lambda_{p,i}\sigma_{0,p} - \sigma_{0,i} = 0 \quad (3)$$

has a unique solution for $\lambda_{x,y}$ (k solutions, respectively). Above, there is an equation for each missing edge $i \leftrightarrow j$ not involving the root and each missing edge $0 \leftrightarrow i$. In each equation, p denotes the unique parent of i and q the unique parent of j in the tree (V, D). We will use this convention frequently in the following. If there is a missing edge $0 \leftrightarrow i$, then $\lambda_{p,i}$ is generically identifiable as $\lambda_{p,i} = \sigma_{0,i}/\sigma_{0,p}$.

Polynomial Identity Testing

We consider multivariate polynomials in indeterminates $x_1, \ldots, x_n$ over some field. An *arithmetic circuit* is an acyclic graph. The nodes with indegree 0 (input gates) are labelled with variables or constants from a given field. The inner nodes are either labelled with $+$ (addition gate) or $*$ (multiplication gate). There is one unique node with outdegree 0, the output gate. An arithmetic circuit computes a polynomial at the output gate in the natural way, see Saptharishi et al. (2021) for more background. Such arithmetic circuits occur in our setting for instance as iterated matrix products or determinants with polynomials as entries.

Given a polynomial computed by an arithmetic circuit, we want to know whether it is identically zero. This problem is known as *polynomial identity testing (PIT)*. Note that we cannot check this efficiently by simply computing the polynomial at the output gate explicitly, since it can have an exponential number of monomials. The Schwartz-Zippel lemma provides an efficient randomized solution for PIT.

Lemma 1 (Schwartz 1980; Zippel 1979)**.** *Let $p(x_1, ..., x_n)$ be a non-zero polynomial of total degree $\leq d$ over a field K. Let $S \subseteq K$ be a finite set and let $a_1, ..., a_n \in S$ be selected uniformly at random. Then* $\Pr(p(a_1, a_2, ..., a_n) \neq 0)) \geq 1 - \frac{d}{|S|}$.

The error probability introduced by the Schwartz-Zippel lemma can be easily controlled. Assume that the final algorithm calls the Schwartz-Zippel lemma T times. Then we choose a set S of size dT/ϵ. By the union bound, the overall error probability is $\leq T \cdot \frac{d}{|S|} = \epsilon$.

Besides polynomials, we will also consider *fractional affine square-root terms of polynomials (FASTP)*, as defined by van der Zander et al. (2022). A FASTP is an expression of the form $\frac{p+q\sqrt{s}}{r+t\sqrt{s}}$, where p, q, r, t, s are polynomials given by arithmetic circuits. Van der Zander et al. (2022, Lemma 2) prove that we can test whether a FASTP is the root of a quadratic polynomial with polynomials as coefficients in randomized polynomial time by reduction to Lemma 1.

The Rank of Edges

Let $M_{\text{miss}} = (V, \bar{B})$ denote the graph of missing bidirected edges. Let $i \leftrightarrow j \in \bar{B}$ and consider the corresponding equation (2). We write the coefficients in this equations as a 2×2-matrix $\left(\begin{smallmatrix}\sigma_{p,q} & \sigma_{i,q}\\ \sigma_{p,j} & \sigma_{i,j}\end{smallmatrix}\right)$.

Definition 2. The *rank* of a missing bidirected edge is the rank of the corresponding 2×2-matrix.

Recall that the $\sigma_{x,y}$ are polynomials in the $\lambda_{x,y}$ and $\omega_{x,y}$. Above, by rank, we mean the rank over the corresponding rational function field.

It turns out that rank-1 and rank-2 edges provide different kind of equations. (We can ignore potential rank-0 edges, since they only give trivial equations.) We will show in the subsequent sections that rank-1 edges uniquely identify one of the two parameters, but need not tell anything about the other. Rank-2 edges do not identify any of the parameters directly but can be used to transfer known values for one of the parameters to the other parameter and vice versa.

Rank-1 Edges

If a missing edge $i \leftrightarrow j$ has rank 1, then the left-hand side of the corresponding equation (2) factorizes into two linear terms.

Lemma 3. $\det\left(\begin{smallmatrix}\sigma_{p,q} & \sigma_{i,q}\\ \sigma_{p,j} & \sigma_{i,j}\end{smallmatrix}\right) = 0$ *if and only if* $\lambda_{p,i}\lambda_{q,j}\sigma_{p,q} - \lambda_{p,i}\sigma_{p,j} - \lambda_{q,j}\sigma_{i,q} + \sigma_{i,j}$ *factorizes.* ▷

Rank-2 Edges

If a missing edge has rank 2, then $\lambda_{q,j}\sigma_{p,q} - \sigma_{p,j} \neq 0$ as a polynomial. If it were identically 0, then by plugging this into (2), we get that $\lambda_{q,j}\sigma_{i,q} - \sigma_{i,j} = 0$, too. This is however a contradiction, since it yields a linear dependence between the two columns of $\left(\begin{smallmatrix}\sigma_{p,q} & \sigma_{i,q}\\ \sigma_{p,j} & \sigma_{i,j}\end{smallmatrix}\right)$. Thus, we can write $\lambda_{p,i} = \frac{\lambda_{q,j}\sigma_{i,q} - \sigma_{i,j}}{\lambda_{q,j}\sigma_{p,q} - \sigma_{p,j}}$ in the rank-2 case. The same is true when we exchange the roles of $\lambda_{p,i}$ and $\lambda_{q,j}$. This shows the following:

Lemma 4. *Let $i \leftrightarrow j$ be a missing edge of rank two. Then $\lambda_{p,i}$ is generically identifiable (2-identifiable, unidentifiable) if and only if $\lambda_{q,j}$ is generically identifiable (2-identifiable, unidentifiable, respectively).* □

Remark 5. If $\lambda_{q,j}$ is determined by a rational expression, so is $\lambda_{p,i}$. If $\lambda_{q,j}$ is determined by a FASTP, so is $\lambda_{p,i}$. Moreover, the square root term is the same for $\lambda_{p,i}$ and $\lambda_{q,j}$. ▷

Computing the Rank of an Edge

How do we check whether a matrix of the form $\left(\begin{smallmatrix}\sigma_{p,q} & \sigma_{i,q}\\ \sigma_{p,j} & \sigma_{i,j}\end{smallmatrix}\right)$ has rank 1 or 2? We can compute the entries of this matrix by computing $(I - \Lambda)^{-1}\Omega(1 - \Lambda)^{-T}$ and then compute the determinant, which will be a polynomial in the $\lambda_{s,t}$ and $\omega_{s,t}$. This can be done using a polynomial-sized arithmetic circuit. However, we cannot compute this polynomial explicitly, since intermediate results might have an exponential number of monomials.

Luckily, this is an instantiation of the polynomial identity testing problem, and by Lemma 1, it is sufficient to replace the indeterminates by random numbers chosen from a large enough set. The resulting determinant can be computed in polynomial time, since the numbers are all small. We get:

Lemma 6. *Given a mixed graph $M = (V, B, D)$ and a missing edge $i \leftrightarrow j$, there is a randomized polynomial time algorithm that determines the rank of $i \leftrightarrow j$.* □

Composition and Elimination

Van der Zander et al. (2022) show that each missing edge not involving the root 0 gives a bilinear equation in $\lambda_{p,i}$ and $\lambda_{q,j}$, see (2). In the following three sections, we study this problem in a more abstract setting: We are given a set of indeterminates X and a set of bilinear equations such that each equation contains exactly two indeterminates. For each pair of variables, there is at most one such equation. Therefore, we can view the problem as an undirected graph G with node set X and there is an edge connecting two nodes iff there is an equation involving these two variables. We will assume that G is connected, since if not, we can treat each connected component separately.

Let

$$axy - bx + cy - d = 0, \tag{4}$$
$$Ayz - By + Cz - D = 0 \tag{5}$$

be two such bilinear equations, sharing the indeterminate y. If we want to eliminate y, we can write (assuming all denominators are nonzero)

$$y = \frac{bx + d}{ax + c}, \quad z = \frac{By + D}{Ay + C}$$

and obtain by plugging the first equation into the second

$$z = \frac{(Bb + Da)x + (Bd + Dc)}{(Ab + Ca)x + (Ad + Cc)}. \tag{6}$$

The new coefficients are given by the entries of the matrix product $\left(\begin{smallmatrix} B & D \\ A & C \end{smallmatrix}\right)\left(\begin{smallmatrix} b & d \\ a & c \end{smallmatrix}\right) = \left(\begin{smallmatrix} Bb+Da & Bd+Dc \\ Ab+Ca & Ad+Cc \end{smallmatrix}\right)$. This is the well-known composition of Möbius transforms, see e.g. Krantz (1999). The definition of a Möbius transform requires that the 2×2-matrix is invertible, that is, we have a rank-2 edge.

We can get a meaningful interpretation even in the case of rank-1 edges if we use the language of resultants (Gelfand, Kapranov, and Zelevinsky 1994; Cox, Little, and O'Shea 1997), a standard tool for solving polynomial equations. Write both equations as polynomial equations in the variable y with coefficients in $R[x, z]$:

$$f(y) = (ax + c)y - (bx + d) = 0,$$
$$g(y) = (Az - B)y + (Cz - D) = 0.$$

The resultant w.r.t. y of the two polynomials is

$$\begin{aligned}\mathrm{Res}_y(f, g) &= \det \begin{pmatrix} -(bx + d) & ax + c \\ Cz - D & Az - B \end{pmatrix} \\ &= -(Ab + Ca)xz + (Bb + Da)x \\ &\quad - (Ad + Cc)z + Bd + Dc.\end{aligned}$$

The fact that $\mathrm{Res}_y(f, g) = 0$ can be rewritten as

$$(Ab + Ca)xz + (Ad + Cc)z = (Bb + Da)x + Bd + Dc,$$

which is equivalent to (6).

When eliminating the variable y in (4) and (5) we implicitly assume a "direction" in the process, the starting variable is x and we want to express z in terms of x. We can also do the opposite and express x in terms of z. In this case all matrices are replaced by their adjoints.

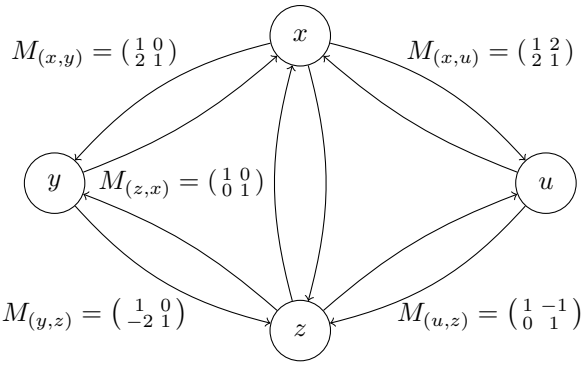

Figure 3: An example of the graph $\vec{G}$. For simplicity, we show only one weight for each pair of edges between two nodes. The other edge has the corresponding adjoint matrix as weight.

This suggests the following modelling as an edge labeled directed graph $\vec{G}$. The node set is again the set of variables X. Assume there is an equation involving x and y like in (4). We add a directed edge (x, y) with weight $\left(\begin{smallmatrix} b & d \\ a & c \end{smallmatrix}\right)$ and the reverse edge (y, x) with weight $\left(\begin{smallmatrix} c & -d \\ -a & b \end{smallmatrix}\right)$. Call the resulting set of directed edges E. For an edge $e \in E$, let M_e denote the corresponding weight. The corresponding polynomial is called f_e. Note that the polynomial is the same irrespective of the direction of the edges, but the weights are different, they are the adjoint matrices of each other. Figure 3 shows an example of such a graph $\vec{G}$.

Lemma 7. *Let $x_1, \ldots, x_t$ be a simple directed path in $\vec{G}$. Then the coefficients of*

$$\mathrm{Res}_{x_{t-1}}(\ldots \mathrm{Res}_{x_3}(\mathrm{Res}_{x_2}(f_{(x_1,x_2)}, f_{(x_2,x_3)}), f_{(x_3,x_4)}), \ldots, f_{(x_{t-1},x_t)}).$$

are given by $M_{(x_{t-1},x_t)} \cdots M_{(x_2,x_3)} \cdot M_{(x_1,x_2)}$. ▷

Identification Using Cycles

We can identify a variable, if we close the path in Lemma 7 by adding an edge (x_t, x_1), obtaining a (simple) cycle. This corresponds to an equation

$$x_1 = \frac{bx_1 + d}{ax_1 + c} \qquad \text{or} \qquad ax_1^2 + (c - b)x_1 - d = 0$$

where the coefficients are given by the matrix product $M_{(x_t,x_1)} \cdot M_{(x_{t-1},x_t)} \cdots M_{(x_2,x_3)} \cdot M_{(x_1,x_2)}$.

Lemma 8. *Let $\vec{G}$ be as above and let $x_1, \ldots, x_t, x_1$ be a directed cycle in $\vec{G}$. Let $\left(\begin{smallmatrix} b & d \\ a & c \end{smallmatrix}\right)$ be the product $M_{(x_t,x_1)} \cdot M_{(x_{t-1},x_t)} \cdots M_{(x_2,x_3)} \cdot M_{(x_1,x_2)}$.*

1. *If $a \neq 0$, then x_1 has at most two solutions (depending on the discriminant).*
2. *If $a = 0$ but $c - b \neq 0$, then x_1 has exactly one solution.*
3. *If $a = c - b = 0$ but $d \neq 0$, then there is no solution.*
4. *If $a = c - b = d = 0$, then x_1 has infinitely many solutions.* ▷

(In our application, case 3 cannot happen, since we assume that the covariances are in accordance with the model.)

Remark 9. $a = c - b = d = 0$ in item 4 of Lemma 8 is equivalent to the statement that $\left(\begin{smallmatrix} b & d \\ a & c \end{smallmatrix}\right)$ is a multiple of the identity matrix.

Definition 10. We call a cycle in Lemma 8 that satisfies the first or second condition *identifying*.

Example 11. The cycle $x \to z \to y \to x$ in Figure 3 is not identifying, since $M_{(z,x)}M_{(y,z)}M_{(x,y)} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$. On the other hand, the cycle $x \to u \to z \to x$ is identifying, since $M_{(z,x)}M_{(u,z)}M_{(x,u)} = \left(\begin{smallmatrix} -1 & 1 \\ 2 & 1 \end{smallmatrix}\right)$

Now assume that the system of polynomial equations of G has a finite number of solutions. Then there is a subset of size n of these equations, such that this system has a finite number of solutions, where $n := |X|$ is the number of variables. Graph-theoretically, a graph on n nodes with n edges consists of connected components such that each connected component has the same number of nodes and edges. A connected component with c nodes and c edges is an undirected tree with one additional edge which creates a cycle. The system of the equations which belong to the edges of the cycle has a finite number of solutions, too. (This follows easily, since every variable not in the cycle appears in only one equation.) It is well-known (Cox, Little, and O'Shea 1997) that such a system of polynomial equations has a finite number of solutions if the iterated resultant of the polynomials (as in Lemma 7) is nonzero. By the correspondence between resultants and matrix products (Lemma 7) we get:

Lemma 12. *If the system of equations given by G has finitely many solutions, then $\vec{G}$ has an identifying cycle.* □

We also have a converse in the case when all edges in G are rank-2 edges.

Lemma 13. *Let all edges in G be rank-2 edges. If there is an identifying cycle in $\vec{G}$, then the system of equations given by G has a finite number of solutions.* ▷

Remark 14. Note that from the proof of Lemma 13 it even follows that the number of total solutions is the number of solutions for x_1.

Finding Identifying Cycles

A *directed walk* in a directed graph $\vec{G} = (X, E)$ is a sequence of nodes $x_0, \ldots, x_t$ (potentially with repetitions) such that (x_i, x_{i+1}) is an edge in $\vec{G}$ for all $0 \leq i < t$. t is the length of the walk. A walk with no repetitions is called a *path*. A walk is *closed* if $x_0 = x_t$.

Now assume that each edge e in $\vec{G}$ has a weight w_e. Let W be the weighted adjacency matrix, that is, the entry in position (i, j) is $w_{(v_i,v_j)}$ if $(v_i, v_j) \in E$ and 0 otherwise. (We assume w.l.o.g. that $X = \{v_1, \ldots, v_n\}$.) For a walk p consisting of nodes $x_1, \ldots, x_t$, the weight p is defined as $w(p) = w_{(x_1,x_2)} \cdots w_{(x_{t-1},x_t)}$, that is, the product of the edge weights. The weights can come from any ring. In the light of the previous section, we will choose the ring $\mathbb{R}[\sigma_{i,j}]^{2\times 2}$, the ring of 2×2-matrices with polynomials in the $\sigma_{i,j}$ as entries, which are itself polynomials in the $\lambda_{i,j}$ and $\omega_{i,j}$. Note that this ring is not commutative, so the order in which we multiply the weights matters.

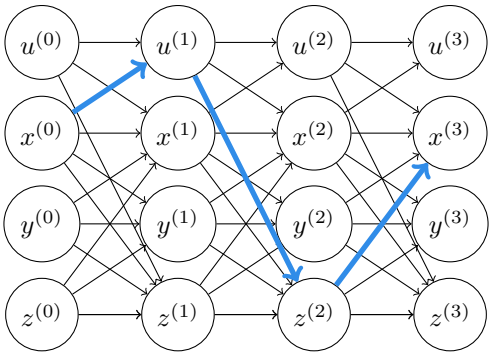

Figure 4: The layered graph $\vec{G}^{(3)}$ corresponding to $\vec{G}$ from Figure 3. Weights are not drawn for simplicity. The bold blue path in this graph corresponds to the identifying cycle $x \to u \to z \to x$.

Fact 15. The entry of W^t in position (i, j) is the sum of the weights of all directed walks of length t from v_i to v_j. □

For a proof, see Stanley (2018). If the edge weights for instance are all 1, then the entry of the matrix power W^t in position (i, j) just counts all walks of length t from v_i to v_j. However, the weights of different paths might not simply add up as in this simple case, but the weights of the all walks from v_i to v_j may add up to 0. So the entry of W^t in position (i, j) is zero, but there are walks from v_i to v_j. Therefore, we extend the weights to make every walk unique. Given the directed graph $\vec{G} = (X, E)$, we construct a layered version $\vec{G}^{(t)} = (X^{(t)}, E^{(t)})$ of it. For every node $v_i \in V$, there are $t + 1$ copies $v_i^{(0)}, \ldots, v_i^{(t)}$. And for each edge $(v_i, v_j) \in E$, there are t copies $(v_i^{(k)}, v_j^{(k+1)})$, $0 \leq k < t$. We call all nodes with superscript k the kth layer of $\vec{G}^{(t)}$. Edges only go from one layer to the next. One can think of the layers as being time steps. In $\vec{G}^{(t)}$, every walk is necessarily a path, since we cannot take an edge twice due to the layered structure. Figure 4 shows the layered graph corresponding to the graph in Figure 3.

Observation 16. There is a one to one correspondence between walks of length t from v_i to v_j in $\vec{G}$ and paths from $v_i^{(0)}$ to $v_j^{(t)}$ in $\vec{G}^{(t)}$. □

If (v_i, v_j) has weight $w_{(v_i,v_j)}$ in $\vec{G}$, then all edges $(v_i^{(k)}, v_j^{(k+1)})$, $0 \leq k < t$, have weight $w_{(v_i,v_j)}$ in $\vec{G}^{(t)}$. For a walk p in $\vec{G}$ of length t, let q be the corresponding path in $\vec{G}^{(t)}$. We have $w(p) = w(q)$, so the cancellation problem is still there. We resolve this problem by extending the weight function of $\vec{G}^{(t)}$. Let $x_{i,j}^{(k)}$, $1 \leq i, j \leq n$, $1 \leq k \leq t$, be new indeterminates. The new weight of the edge $(v_i^{(k)}, v_j^{(k+1)})$ is $\hat{w}(v_i^{(k)}, v_j^{(k+1)}) = x_{i,j}^{(k+1)} w(v_i, v_j)$ So if the original weights are from some ring R, the new weights are now from the polynomial ring $\hat{R} := R[x_{i,j}^{(k)}]$.

Observation 17. Let p be a walk in $\vec{G}$ that visits the nodes $v_{i_0}, \dots, v_{i_t}$ in $\vec{G}$ in this order. Let q be the corresponding path in $\vec{G}^{(t)}$. Then $\hat{w}(q) = \prod_{k=1}^{t} x^{(k)}_{i_{k-1},i_k} w(p)$. In this way, each path q with $w(p) \neq 0$ gets a unique weight (in $\hat{R}$), since the monomial $\prod_{k=1}^{t} x^{(k)}_{i_{k-1},i_k}$ exactly describes the edges taken in q. □

Example 18. If we identify the nodes u, x, y, z with v_1, v_2, v_3, v_4 in Figure 4, then the bold blue path has weight $x^{(1)}_{2,1}x^{(2)}_{1,4}x^{(3)}_{4,2} \cdot \left(\begin{smallmatrix} -1 & 1 \\ 2 & 1 \end{smallmatrix}\right) \cdot \left(\begin{smallmatrix} -1 & 1 \\ 2 & 1 \end{smallmatrix}\right)$ is the original weight in $\vec{G}$ and $x^{(1)}_{2,1}x^{(2)}_{1,4}x^{(3)}_{4,2}$ is the unique label of the path.

Let $\hat{W}$ be the weighted adjacency matrix of $\vec{G}^{(t)}$ using the weights $\hat{w}$. By Fact 15 and Observation 17 we have:

Lemma 19. *The entry of $\hat{W}^t$ corresponding to the nodes $v_i^{(0)}$ and $v_j^{(t)}$ is the sum of all $\prod_{k=1}^{t} x^{(k)}_{i_{k-1},i_k} w(p)$ over all walks p of length t from v_i to v_j in $\vec{G}$ (interpreted as a path in $\vec{G}^{(t)}$ using Observation 16).* □

The matrix $\hat{W}$ has size $(t+1)n$. However, due to the layered structure of $G^{(t)}$, it has a special structure: It has a block structure with $\hat{W}_1, \dots, \hat{W}_t$ being the only nonzero $n \times n$ blocks above the main diagonal. Each block $\hat{W}_k$ contains the weights of the edges between layer $k-1$ and k. It is easy to check that $\hat{W}^t$ has only non-zero block, which is in the top right and contains the matrix $\hat{W}_1 \dots \hat{W}_t$. Therefore, instead of raising $\hat{W}$ to the tth power, we can simply compute the iterated product $\hat{W}_1 \dots \hat{W}_t$, which is more efficient.

Polynomial Identity Testing

Checking whether some entry in the iterated product $\hat{W}_1 \dots \hat{W}_t$ is nonzero is an instance of PIT. Note that in Lemma 7, the weights are multiplied from the left. This is not a problem, since we can simply transpose all 2×2-matrices and use the fact that $(AB)^T = B^T A^T$.

Lemma 20. *There is a randomized polynomial time algorithm which given a node v_i checks whether v_i lies on an identifying closed walk of length at most t.* ▷

Selfreducibility

Note that while we now have an algorithm to test whether there is an identifying closed walk, this algorithm does not construct the identifying closed walk. Furthermore, an identifying closed walk is not necessarily a simple cycle. However, our interpretation in terms of resultants in Lemma 7 is only valid for simple cycles.

The first problem can be solved using a technique called selfreducibility. We remove an arbitrary edge from the graph and check whether there is still an identifying closed walk. If yes, we can safely remove the edge. If not, we know that the edge is in every remaining identifying closed walk and we keep it. The edges that remain will form an identifying closed walk.

The second problem will be solved by the following lemma:

Algorithm 1: Finding identifying cycles

Input: A graph $\vec{G} = (X, E)$
Output: An identifying cycle

```
for t = 1, ..., n do
    Build the corresponding graph G^(t) and matrix Ŵ.
    Replace all indeterminates by random values according to the Schwartz-Zippel lemma.
    Compute Ŵ_1 ··· Ŵ_t.
    Check whether there is an identifying closed walk using Lemma 20 for i = 1, ..., n.
    if there is such a walk then
        Compute such a walk p using selfreducibility.
        return p
return no solution
```

Lemma 21. *The shortest identifying closed walk in G is always a cycle.* ▷

An Algorithm for Finding Identifying Cycles

Algorithm 1 summarizes the results of this section.

Lemma 22. *Algorithm 1 finds an identifying cycle if there is one. It runs in randomized polynomial time.* ▷

Identification Using Rank-1 Edges

We consider a missing edge $i \leftrightarrow j$ of rank 1. Again, p is the parent of i and q of j in the directed graph (V, D). We will show that a rank-1 edge always uniquely identifies one of the nodes (and need not provide any information about the other node, since when we plug in the unique value in the equation, the equation will be identically 0).

Definition 23. Let U, W be two sets of nodes. A pair (S, T) of sets of nodes *trek-separates* U and W if every trek from any $i \in U$ to any $j \in W$ intersects S on its left side or T on its right side.

Theorem 24 (see Drton 2018, Theorem 11.1)**.** *The submatrix with rows corresponding to U and columns corresponding to W has (generic) rank $\leq r$ if there is a pair (S, T) with $|S| + |T| \leq r$ such that (S, T) trek-separates (U, W).*

In our case $U = \{i, p\}$ and $W = \{j, q\}$ (p and q can be potentially the same). We are interested when $\left(\begin{smallmatrix} \sigma_{p,q} & \sigma_{i,q} \\ \sigma_{p,j} & \sigma_{i,j} \end{smallmatrix}\right)$ has rank 1. In this case either S or T consists of one node and the other set is empty. The situation is symmetric, so we assume that $|T| = 1$. T cannot be $\{j\}$, since then the trek from i to q (consisting only of directed edges) is not separated. Let $0, p_1, \dots, p_t, p, i$ be the directed path from the root 0 to i. The trek consisting of the left side being any suffix of this path starting in node $x \in \{0, p_1, \dots, p_t, p, i\}$ and the bidirected edge $x \leftrightarrow j$ is not trek-separated by (S, T). Therefore, all bidirected edges $x \leftrightarrow j$ are missing! See Figure 5 for an illustration.

We will now prove that all these missing edges are useless for identifying any of the λ-parameter in the directed path.

- As $0 \leftrightarrow j$ is missing, we can identify $\lambda_{q,j} = \frac{\sigma_{0,j}}{\sigma_{0,q}}$.

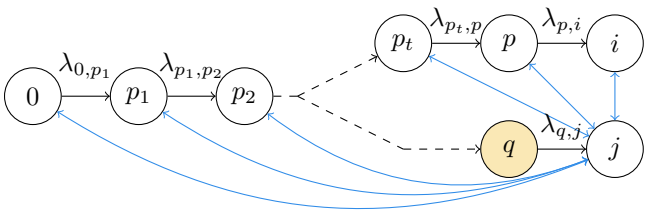

Figure 5: If the missing edge $i \leftrightarrow j$ has rank 1, then all the other blue edges have to be missing by trek-separability. The best choice for T is $T = \{q\}$ (drawn yellow).

- Since $p_1 \leftrightarrow j$ is missing, we get by (2)

$$\lambda_{0,p_1}\lambda_{q,j}\sigma_{0,q} - \lambda_{0,p_1}\sigma_{0,j} - \lambda_{q,j}\sigma_{p_1,q} + \sigma_{p_1,j} = 0.$$

  Plugging in $\lambda_{q,j} = \frac{\sigma_{0,j}}{\sigma_{0,q}}$, we get $\sigma_{0,j}\sigma_{p_1,q} - \sigma_{0,q}\sigma_{p_1,j} = 0$. Thus this edges does not yield any information about λ_{0,p_1} (and we already identified $\lambda_{q,j}$). Furthermore, the equation means that the missing edge $p_1 \leftrightarrow j$ is rank-1, too, and we get that $\lambda_{q,j} = \frac{\sigma_{0,j}}{\sigma_{0,q}} = \frac{\sigma_{p_1,j}}{\sigma_{p_1,q}}$.
- We can go on inductively until we reach the node p and get $\lambda_{q,j} = \frac{\sigma_{0,j}}{\sigma_{0,q}} = \cdots = \frac{\sigma_{p,j}}{\sigma_{p,q}}$.
- Finally, since the edge $i \leftrightarrow j$ is missing,

$$\lambda_{p,i}\lambda_{q,j}\sigma_{p,q} - \lambda_{p,i}\sigma_{p,j} - \lambda_{q,j}\sigma_{i,q} + \sigma_{i,j} = 0.$$

  Plugging in $\lambda_{q,j} = \frac{\sigma_{p,j}}{\sigma_{p,q}}$, we get $\sigma_{p,j}\sigma_{i,q} - \sigma_{p,q}\sigma_{i,j} = 0$ and the equation does not depend on $\lambda_{p,i}$.

There is the special case that $p = q$. Then q lies on any directed path to i. However, this does not matter in the definition of trek-separation, since the set $T = \{q\}$ can only intersect the right side of a trek. Finally i could be a predecessor of j (or vice versa). This is treated similarly.

Lemma 25. *For every missing rank-1 edge $i \leftrightarrow j$, the edges $0 \leftrightarrow i$ or $0 \leftrightarrow j$ (or both) are missing. The corresponding expressions satisfy the equation of the edge $i \leftrightarrow j$.* □

Corollary 26. *Let C be a connected component of the missing edge graph. If C contains a rank-1 edge, then C also contains the root.* □

The Final Algorithm

Algorithm 2 solves the identification problem. For each $\lambda_{p,i}$ is decides in randomized polynomial time whether $\lambda_{p,i}$ is generically identifiable, 2-identifiable, or unidentifiable. (Other cases cannot occur.)

Theorem 27. *Algorithm 2 is correct.*

Proof. Consider a variable $\lambda_{p,i}$. First assume that i is contained in a rank-1 missing edge $i \leftrightarrow j$. If i is connected to the root in M_{miss}, then i is identified by (3). If i is not connected to the root, then j is. The solution for j satisfies the equation by Lemma 25. Therefore, it will not be useful for identifying $\lambda_{p,i}$. Thus we can safely remove all rank-1 edges in step 3. Assume that $\lambda_{p,i}$ is not identified by a rank-1 edge.

Algorithm 2: Identification

Input: A tree-shaped mixed graph $M = (V, D, B)$
Output: For each $\lambda_{p,i}$, we decide whether it is generically identifiable, 2-identifiable, or unidentifiable. In the first two cases, we output FASTPs for $\lambda_{p,i}$.

1: Find all rank-1 edges in the missing edge graph $(V, \bar{B})$ (using Lemma 6).
2: For each missing rank-1 edge $i \leftrightarrow j$, check which of the parameters $\lambda_{p,i}$ or $\lambda_{q,j}$ we can identify (using Lemma 25). Mark the node i or j, respectively, and compute a corresponding rational expression.
3: Remove all rank-1 edges from the missing edge graph. Call the resulting graph H. Let $C_1, \ldots, C_t$ be the connected components of H.
4: **for** each connected component C_i **do**
5: **if** C_i contains a marked node **then**
6: Propagate the result to all unidentified nodes in C_i and produce corresponding rational expressions.
7: **else**
8: Find an identifying cycle in C_i (by Algorithm 1).
9: If no such cycle is found, report that all nodes of C_i are unidentifiable.
10: If the cycle produces one solution, then progagate it to all the nodes of C_i and compute corresponding rational expressions.
11: If the cycle produces two solutions, then propagate it to all the nodes of C_i and compute corresponding FASTPs.
12: Plug the FASTPs into the equations of C_i and use PIT to check whether all equations are satisfied. If yes, keep the solution, otherwise, drop it.

If the rank-2 component of $\lambda_{p,i}$ contains a marked node, which is identifiable, then each node in this component is identifiable by Lemma 4. And by the subsequent Remark 5, we also get a rational expression.

If there is no marked node in the component, then the nodes in the component are k-identifiable iff there is an identifying cycle by Lemmas 12 and 13. If such a cycle is found, it can have either one or two solutions by Lemma 8. These one or two solutions can be propagated using Lemma 4 and Remark 5. Note that we get in total at most two solutions for the component, given by FASTPs (see Remark 14). If there is only one solution, then we are done. If there are two solutions, we plug them into all equations of the component and see whether they both satisfy all equations or just one of them. In the second case, $\lambda_{p,i}$ is identifiable. In the first case, it is 2-identifiable, but not identifiable. (Note that we do not have to check equations of other components, since they do not contain any variables of the present component). We can check whether two FASTPs satisfy a bilinear equation using the test by van der Zander et al. (2022, Lemma 2), since both FASTPs have the same square root term by Remark 5.

If there is no identifying cycle, then the nodes are unidentifiable by Lemma 12. Thus the algorithm is correct. □

The running time of Algorithm 2 is clearly polynomial.

Acknowledgments

We thank Frank-Olaf Schreyer for helpful discussions on resultants.

References

Bollen, K. A. 1989. *Structural equations with latent variables*. Wiley.

Bowden, R. J.; and Turkington, D. A. 1990. *Instrumental Variables*. Cambridge University Press.

Brito, C. 2010. Instrumental sets. In Dechter, R.; Geffner, H.; and Halpern, J. Y., eds., *Heuristics, Probability and Causality. A Tribute to Judea Pearl*, chapter 17, 295–308. College Publications.

Brito, C.; and Pearl, J. 2002a. Generalized Instrumental Variables. In *Proc. Conference on Uncertainty in Artificial Intelligence (UAI)*, 85–93.

Brito, C.; and Pearl, J. 2002b. A graphical criterion for the identification of causal effects in linear models. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, 533–538.

Chen, B.; Pearl, J.; and Bareinboim, E. 2015. Incorporating knowledge into structural equation models using auxiliary variables. In *Proc. Int. Joint Conference on Artificial Intelligence (IJCAI)*, 3577–3583.

Cox, D.; Little, J.; and O'Shea, D. 1997. *Ideals, Varieties, and Algorithms*. Springer.

Drton, M. 2018. Algebraic problems in structural equation modeling. In *The 50th anniversary of Gröbner bases*, 35–86. Mathematical Society of Japan.

Duncan, O. D. 1975. *Introduction to structural equation models*. Academic Press.

Foygel, R.; Draisma, J.; and Drton, M. 2012. Half-trek criterion for generic identifiability of linear structural equation models. *The Annals of Statistics*, 40(3): 1682–1713.

García-Puente, L. D.; Spielvogel, S.; and Sullivant, S. 2010. Identifying Causal Effects with Computer Algebra. In *Proc. Conference on Uncertainty in Artificial Intelligence (UAI)*, 193–200.

Gelfand, I. M.; Kapranov, M.; and Zelevinsky, A. 1994. *Discriminants, resultants, and multidimensional determinants*. Birkhäuser.

Gupta, A.; and Bläser, M. 2023. Identification for Tree-shaped Structural Causal Models in Polynomial Time. *CoRR*, abs/2311.14058.

Krantz, S. G. 1999. *Handbook of Complex Variables*, chapter Linear Fractional Transformations. Birkhäuser.

Kumor, D.; Chen, B.; and Bareinboim, E. 2019. Efficient Identification in Linear Structural Causal Models with Instrumental Cutsets. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 12477–12486.

Kumor, D.; Cinelli, C.; and Bareinboim, E. 2020. Efficient identification in linear structural causal models with auxiliary cutsets. In *Proc. Int. Conference on Machine Learning (ICML)*, 5501–5510.

Pearl, J. 2001. Parameter identification: A new perspective. Technical Report R-276, UCLA.

Pearl, J. 2009. *Causality*. Cambridge University Press.

Saptharishi, R.; et al. 2021. A selection of lower bounds in arithmetic circuit complexity. Version 2021-07-27.

Schwartz, J. T. 1980. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4): 701–717.

Stanley, R. P. 2018. *Algebraic Combinatorics*. Springer.

van der Zander, B.; and Liśkiewicz, M. 2016. On Searching for Generalized Instrumental Variables. In *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, 1214–1222.

van der Zander, B.; Textor, J.; and Liśkiewicz, M. 2015. Efficiently Finding Conditional Instruments for Causal Inference. In *Proc. Int. Joint Conference on Artificial Intelligence (IJCAI)*, 3243–3249.

van der Zander, B.; Wienöbst, M.; Bläser, M.; and Liskiewicz, M. 2022. Identification in Tree-shaped Linear Structural Causal Models. In Camps-Valls, G.; Ruiz, F. J. R.; and Valera, I., eds., *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, volume 151 of *Proceedings of Machine Learning Research*, 6770–6792. PMLR.

Weihs, L.; Robinson, B.; Dufresne, E.; Kenkel, J.; McGee II, K. K. R.; Reginald, M. I.; Nguyen, N.; Robeva, E.; and Drton, M. 2018. Determinantal generalizations of instrumental variables. *Journal of Causal Inference*, 6(1).

Wright, S. 1921. Correlation and causation. *J. Agricultural Research*, 20: 557–585.

Wright, S. 1934. The method of path coefficients. *The Annals of Mathematical Statistics*, 5(3): 161–215.

Zippel, R. 1979. Probabilistic algorithms for sparse polynomials. *Symbolic and algebraic computation*, 216–226.