A Faster Algorithm for the Half-Trek Criterion in Structural Causal Models

Yasmine Briefs
Max Planck Institute for Informatics and
Graduate School of Computer Science
Saarland Informatics Campus
Saarbrücken, Germany
ybriefs@mpi-inf.mpg.de

Markus Bläser
Saarland University
Saarland Informatics Campus
Saarbrücken, Germany
mblaeser@cs.uni-saarland.de

Abstract

Linear structural causal models (SCMs) are used to analyze relationships among random variables. In these models, directed edges encode direct causal effects, while bidirected edges capture latent confounding. The problem of generically identifying the hidden parameters from observed correlations in an efficient manner remains open in causal inference. The half-trek criterion by Foygel, Draisma, and Drton [2012] and its generalization to the edgewise criterion are important sufficient criteria for generic identification in linear SCMs, since the half-trek criterion can be efficiently decided by an algorithm with running time $\mathcal{O}(n^5)$. We here develop a new criterion that is equivalent to the edgewise criterion. Utilizing this criterion, we design a faster randomized algorithm for deciding the half-trek criterion with running time $\mathcal{O}(n^4)$. For sparse graphs, our algorithm even works in $\mathcal{O}(n^3)$. We show that the algorithm is useful in practice by providing an implementation that outperforms the implementation of HTC in the state-of-the-art R package SEMID.

1 Introduction

Understanding when causal effects can be discovered from observational data and can be distinguished from mere correlation is a central problem in statistics and machine learning. This question arises naturally in latent-variable modeling, where observed data are generated by interacting components that are only partially observed. While randomized controlled trials provide a direct route to causal identification by perturbing individual mechanisms, many systems of interest admit no such interventions, for instance due to ethical reasons such as the impossibility of experimentally inducing disease or harm.

The problem of determining whether such recovery is possible is known as the problem of *identification* in causal inference. Given observable parameters like covariances, can we recover the parameters of the model? From an algebraic perspective, it amounts to studying the injectivity—global or generic—of a rational map from the model parameters to the observable covariances.

A widely used formalism capturing these ideas is that of *structural causal models* (SCMs) [Pearl, 2009, Bareinboim et al., 2022]. In this paper, we focus on the identification problem for *linear* SCMs, also known as *linear structural equation models* (SEMs) [Bollen, 1989, Duncan, 1975]. These linear models are ubiquitous in ML as tractable approximations to nonlinear generative mechanisms. They are important because they provide the simplest mathematically tractable framework for reasoning about causality.

Concretely, a linear SCM assumes that each observed variable V_j is generated as $V_j = \sum_i \lambda_{i,j} V_i + \varepsilon_j$, where the coefficients $\lambda_{i,j}$ encode direct effects and the noise terms ε_j are jointly Gaussian with mean zero. Latent confounding is represented through non-diagonal entries of the covariance matrix

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

$\Omega = (\omega_{i,j})$ of the error terms. From an ML perspective, this model can be viewed as a linear latent-variable generator in which dependence is induced both through directed mixing (Λ) and correlated noise (Ω). A formal definition is given in Section 2.

1.1 Generic identification

An SCM is called globally identifiable if for *all* values of the observed covariances, the parameters $\lambda_{i,j}$ and $\omega_{i,j}$ are uniquely determined. Global identifiability can be decided in polynomial time [Drton et al., 2011], but it is often too strong, as illustrated by the example in Figure 1. It is easy to see that the observed covariances satisfy $\omega_{0,0}\lambda_{0,1} = \sigma_{0,1}$ and $\omega_{0,0}\lambda_{0,1}\lambda_{1,2} = \sigma_{0,2}$, where $\omega_{0,0}$ denotes the variance of the error term ε_0 of V_0, and then $\lambda_{1,2} = \frac{\omega_{0,0}\lambda_{0,1}\lambda_{1,2}}{\omega_{0,0}\lambda_{0,1}} = \frac{\sigma_{0,2}}{\sigma_{0,1}}$. (This follows from Wright's trek rule, see Section 2.) $\lambda_{1,2}$ is not globally identifiable, since $\lambda_{1,2}$ is not uniquely determined when $\sigma_{0,1} = 0$.

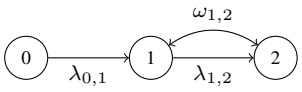

Figure 1: $\lambda_{1,2}$ is identifiable if $\sigma_{0,1} \neq 0$.

Generic identification relaxes the requirements of global identifiability by demanding identifiability only for almost all parameter values, excluding a measure-zero set of degenerate cases or, equivalently, a Zariski closed set. Since these non-generic configurations are not robust and cannot be distinguished statistically in the presence of noise, generic identification better reflects practical learnability of causal structure and effects. In the example of Figure 1, $\lambda_{1,2}$ is generically identifiable, since $\sigma_{0,1} = 0$ defines a set of measure zero or, equivalently, a Zariski closed set.

1.2 Algorithms for generic identification

The complexity of generic identification is wide open. We do not know any hardness results but on the other hand the only known complete algorithms are based on Gröbner bases [García-Puente et al., 2010] or the existential theory of the reals [Dörfler et al., 2024]. Hollering et al. [2026] recently presented a more fine-grained Gröbner basis approach.

As there is no known efficient algorithm for generic identification, researchers started to develop sufficient criteria that are however not complete in the sense that when they cannot identify a parameter, it may still be identifiable by other means. The earliest sufficient condition for identification is the instrumental variable criterion, introduced by Wright [1934] and further developed by Bowden and Turkington [1990]. Tian [2005] introduces a decomposition method that is useful for speeding up existing identification methods. Foygel et al. [2012] propose the half-trek criterion (HTC), which was a major breakthrough in the field. Subsequent improvements and extensions include criteria based on auxiliary variables [Chen et al., 2016, 2017]. Weihs et al. [2018], see also [Chen, 2016], generalize the half-trek criterion to an edgewise half-trek criterion which operates on subsets of the parents of a node, rather than on all parents at once. The half-trek criterion and its generalization to edgewise identification are still the current standard in practice and an efficient implementation is provided in the R package SEMID [Barber et al., 2026]. The running time of their algorithm is $\mathcal{O}(n^5)$. There are more general polynomial-time computable sufficient criteria for generic identifiability, the instrumental cutset criterion [Kumor et al., 2019] and the even more general auxiliary cutset criterion introduced by Kumor et al. [2020]. While both criteria have algorithms to check them in polynomial time, we are not aware of a detailed running time analysis or even a reference implementation. There are also the Tree-ID algorithms [van der Zander et al., 2022, Gupta and Bläser, 2024, Briefs and Bläser, 2025], which are complete randomized polynomial time algorithms which solve the generic identification problem in the special case of SCMs in which the directed edges form an outtree.

We give a provably faster randomized algorithm for the HTC with running time $\mathcal{O}(n^4)$ using methods from randomized linear algebra. For sparse SCMs, the running time is even $\mathcal{O}(n^3)$. We provide an efficient implementation which outperforms the state-of-the-art implementation of SEMID.

1.3 Randomized linear algebra

Randomized linear algebra studies algorithms that use randomness to solve large linear algebra problems efficiently. Randomized linear algebra has its roots in the probabilistic algorithms of algebraic complexity theory developed in the late 1970s and 1980s, where randomness was used to design efficient exact algorithms with bounded error probability for tasks such as matrix rank

or solving sparse linear systems. In the early 2000s, the growth of large data sets made classical deterministic algorithms often too expensive, leading to interest in randomized methods in these fields. This development culminated in influential work, such as the randomized SVD framework of Halko et al. [2011], as well as the regression and sampling results, see e.g. Mahoney [2011]. These works showed that randomized methods often outperform classical approaches, yet they come with provable guarantees. See Martinsson and Tropp [2020] for a modern survey.

1.4 Our results

As our first main contribution, we give a faster algorithm for the half-trek criterion. The running time of our algorithm is $\mathcal{O}(n^4)$, saving a factor of $\Theta(n)$ compared to the current best algorithm by Foygel et al. [2012]. Here, n is the number of nodes in the SCM. More precisely, our algorithm can be implemented to run in time $\mathcal{O}(n^3 + n^2 m_2)$ where m_2 denotes the number of directed edges. If the directed part of the given SCM is sparse, i.e., $m_2 \in \mathcal{O}(n)$, then it runs in $\mathcal{O}(n^3)$. Our algorithm is randomized, that is, it succeeds with high probability. The error bounds are provable, and the error can be made arbitrarily small. The error is one-sided, that is, whenever the algorithm identifies an edge, it is correct. (In complexity terms, our algorithm is an RP-algorithm.) We complement our theoretical algorithm by providing an efficient implementation that outperforms the state-of-the-art implementation in the R package SEMID by at least one order of magnitude in most of our test cases. Although our algorithm is randomized, it produced the same outputs as SEMID on all tests we ran during benchmarking.

Our algorithm can be seen as an example of applying principles from randomized linear algebra to the half-trek criterion. In the HTC, a max-flow computation is used to test the invertibility of certain matrices with rational functions as entries, which stem from the covariance matrix Σ. Explicitly expanding their determinants as rational functions can be expensive because their numerators and denominators can have exponentially many monomials. This is where methods of randomized linear algebra come into play. We first plug in random numbers for the parameters and then perform all computations modulo a random prime. By the famous Schwartz–Zippel lemma (Lemma 2), if the determinant's numerator is a nonzero polynomial, then with high probability, it will also be nonzero at the random evaluation point. Furthermore, the result will be even nonzero with high probability modulo the chosen prime [Schönhage, 1979]. However, this idea alone does not help directly. Computing a maximum flow using the push-relabel algorithm, see e.g. Cormen et al. [2022], which is state-of-the-art in practical applications, takes time $\mathcal{O}(n^3)$. Computing a determinant over finite fields also takes $\mathcal{O}(n^3)$. (We disregard both the recent fast max-flow algorithms by Chen et al. [2025], which would speed up the max-flow approach, as well as fast matrix multiplication, see e.g. Respondek [2025], which would speed up our algorithm, since neither of these are practical.) So how do we gain something here? In the algorithm by Foygel et al. [2012] and its generalization by Weihs et al. [2018], it is important that nodes are considered one after another in the right order, but this order is unknown. If we choose the wrong node then there will be no flow of the required value and this results in wasted computation time, as it is not possible to reuse parts of the maximum flow computation. We here replace these flow computations by linear algebra methods and gradually increase the rank of the matrices by adding rows until one of them is invertible. In this way, even if we consider the wrong node, it will lead to a rank increase and no work is wasted.

The work by Weihs et al. [2018] shows that the existence of certain flows implies the invertibility of certain matrices which can then be used for identification. The implication in Weihs et al. [2018] only goes in one direction, so one could hope that by checking the invertibility directly, one might get a stronger criterion. Unfortunately, this is not the case: Existence of the flows is actually equivalent to the invertibility, which we prove in Proposition 1. Thus, by checking invertibility directly, we still compute the HTC, but with a faster algorithm.

However, it turns out that the matrices used by Weihs et al. [2018] are not suited for our incremental rank increase method, since their definition might change during the identification process. As a further result, we prove a new structural theorem (Theorem 1) that allows us to replace these matrices by matrices which are suited for the incremental rank increase, yet, the resulting identification procedure is still equivalent to the (edgewise) HTC.

1.5 Organization of the paper

Section 2 introduces the necessary notation and defines SCMs. In Section 3, we show that the existence of a half-trek system with no sided intersection is equivalent to the invertibility of certain matrices. Parts of the proofs are moved to Appendix C, where we also prove further criteria. Our algorithm is presented in Section 4. We bound the error probability, estimate its running time, and prove its correctness. More details on the running time are given in Appendix A.2, and a detailed estimate of the error probability is given in Appendix B. In Section 5, we present the results of the benchmarking, with further details given in Appendix E.

2 Preliminaries

A *mixed graph* $M = (V, D, B)$ is a graph with a node set V and two types of edges, directed and bidirected. We can assume w.l.o.g. that the node set is $V = \{0, 1, \dots, n-1\}$. D is the set of directed edges. We denote directed edges by (i, j) or $i \to j$. B is the set of bidirected edges, which we denote by $\{i, j\}$ or $i \leftrightarrow j$. For a node v, $\mathrm{pa}(v) = \{u \mid u \to v \in D\}$ is the set of parents of v and $\mathrm{sib}(v) = \{u \mid u \leftrightarrow v \in B\}$ is the set of all nodes u connected by a bidirected edge to v. To each node i corresponds a random variable V_i. To each directed edge $i \to j$, we associate an indeterminate $\lambda_{i,j}$, and to each bidirected edge $u \leftrightarrow v$ an indeterminate $\omega_{u,v}$. Each random variable V_i is assumed to depend linearly on its parent variables together with an additional error term ε_i. More precisely, $V_i = \sum_{j \in \mathrm{pa}(i)} \lambda_{j,i} V_j + \varepsilon_i$. The error terms are assumed to be jointly normally distributed with zero mean. A bidirected edge $i \leftrightarrow j$ indicates that the corresponding error terms ε_i and ε_j may be correlated, with covariance parameter $\omega_{i,j}$. This specification defines a standard model, commonly referred to as a *linear structural equation model*; see, for example, Drton [2018], Pearl [2009]. We denote by Λ and Ω the matrices consisting of the parameters, $\Lambda = (\lambda_{i,j})$ and $\Omega = (\omega_{i,j})$, where the elements on the diagonal of Ω are the variances of the error terms.

Let $\sigma_{i,j}$ denote the observed covariance between the random variables V_i and V_j. Wright's trek rule expresses $\sigma_{i,j}$ in terms of the parameters $\lambda_{i,j}$ and $\omega_{i,j}$ of the underlying mixed graph. A *trek* from i to j consists of two directed walks $i_0 \to i_1 \to \cdots \to i_k = i$ and $j_0 \to j_1 \to \cdots \to j_\ell = j$ in (V, D) such that either $i_0 = j_0$ or there exists a bidirected edge $i_0 \leftrightarrow j_0 \in B$. The i-nodes are called the left part of the trek, the j-nodes the right part. Recall that nodes can repeat in walks and note that the two parts of a trek are allowed to intersect. We call a trek a *half-trek* if $k = 0$, that is, the left part of the trek only consists of the node i. For any node v, $\mathrm{tr}(v)$ denotes the set of all nodes reachable via a trek from v and $\mathrm{htr}(v)$ is the set of all nodes reachable via a half-trek from v.

For a trek π and parameter matrices Λ and Ω, $\pi(\Lambda, \Omega)$ denotes the trek monomial of the trek π, which is the product of all edge weights in the trek. In the case $i_0 = j_0$, we also multiply the monomial with ω_{i_0,i_0}. $\mathscr{T}(u, v)$ denotes the set of all treks from u to v and $\mathscr{H}(u, v)$ the set of all half-treks from u to v. If the directed graph (V, D) contains cycles, then both sets can be infinite. Wright's trek rule [Wright, 1934] now expresses the covariances $\sigma_{i,j}$ in terms of trek monomials:

$$\sigma_{i,j} = \sum_{\pi \in \mathscr{T}(i,j)} \pi(\Lambda, \Omega)$$

If the graph (V, D) is cyclic, then the sums are formal power series. If $\det(I - \Lambda) \neq 0$, then we can rewrite Wright's trek rule in matrix form:

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

where $\Sigma = (\sigma_{i,j})$ is the covariance matrix. A parameter $\lambda_{i,j}$ is *generically identifiable* if for almost all (Λ, Ω), all solutions (Λ', Ω') yielding the same Σ in (1) as (Λ, Ω) assign the same value to $\lambda_{i,j}$.

Let $S = \{s_1, \dots, s_k\}$ and $T = \{t_1, \dots, t_k\}$ be two sets of nodes. A set of treks $\pi_1, \dots, \pi_k$ is a *trek system* from S to T if s_i is the node at the left end of the trek π_i and t_i is the node at the right end, $1 \leq i \leq k$. A trek system is a *half-trek system* if each π_i is a half-trek. The system has *no sided intersection* if the left parts of the treks are pairwise node-disjoint and the right parts are pairwise node-disjoint, too. A node set $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ satisfies the *edgewise half-trek criterion* with respect to a node v and a set $W \subseteq \mathrm{pa}(v)$ [Weihs et al., 2018, Theorem 4.1] if

1. There exists a half-trek system from Y to W with no sided intersection, and

2. for every trek π from $y \in Y$ to v we have that either
   (a) π ends with an edge of the form $s \to v$ where either $s \in W$ or $s \to v$ is known to be generically identifiable, or
   (b) π begins with an edge of the form $y \leftarrow h$ where $h \to y$ is known to be generically identifiable.

If there is such a Y, then each $w \to v$ with $w \in W$ is generically identifiable. The original half-trek criterion by Foygel et al. [2012] is the special case with $W = \mathrm{pa}(v)$ (and a slightly less general condition 2). The existence of a half-trek system with no sided intersection can be efficiently checked using max-flow computations. Therefore, the half-trek criterion can be checked in polynomial time. For the edgewise HTC, we need to iterate over all subsets W, therefore it is only polynomial time in mixed graphs with bounded indegree.

3 An equivalent criterion for edgewise generic identifiability

Weihs et al. [2018, Theorem 4.1] prove a sufficient condition, called the edgewise criterion, for the generic identifiability of any set of incoming edges of a fixed node v. Their theorem generalizes the half-trek criterion [Foygel et al., 2012, Theorem 1], which is a sufficient condition for the generic identifiability of *all* incoming edges of a fixed node. Similar to the half-trek criterion, this more general edgewise criterion requires finding a set of nodes Y such that there exists a half-trek system from Y to a subset W of v's parents with no sided intersection. The proof of the criterion, however, only uses the existence of this half-trek system to infer that a specific matrix $A^v_{Y,W}$ is generically invertible, where the indices refer to the node v and the sets Y, W. (Generically invertible means invertible as a matrix with rational functions as entries.) This suggests that one may obtain a potentially stronger criterion by instead directly checking if the matrix $A^v_{Y,W}$ is generically invertible. However, we prove that the two criteria are equivalent, although their witnessing sets Y may differ: the subsequent Proposition 1 is equivalent to Theorem 4.1 in Weihs et al. [2018]. For ease of presentation, we drop the indices of $A^v_{Y,W}$ and simply call the matrix A in the following.

Proposition 1. *Let $M = (V, D, B)$ be a non-empty mixed graph and let $v \in V$. Let $W \subseteq \mathrm{pa}(v)$. Suppose there exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*

1. *The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in H_i} \sigma_{h,w_j}\lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \ldots, y_k\}$, $W = \{w_1, \ldots, w_k\}$ and the H_i contain all $h \in \mathrm{pa}(y_i)$ such that $h \to y_i$ is known to be generically identifiable, and*
2. *for every trek π from $y \in Y$ to v, we have that either*
   (a) π ends with an edge of the form $s \to v$ where either $s \in W$ or $s \to v$ is known to be generically identifiable, or
   (b) π begins with an edge of the form $y \leftarrow h$ where $h \to y$ is known to be generically identifiable.

Then for each $w \in W$, $w \to v$ is generically identifiable.

A proof of the proposition as well as a proof of the equivalence to Theorem 4.1 in Weihs et al. [2018] can be found in Appendix C. In the definition of the matrix A, $A_{i,j}$ depends on the set H_i. When more and more edges are identified during the course of the identification, the definition of $A_{i,j}$ changes. A key ingredient of our algorithm is a new criterion (see Theorem 1) that uses a modified definition of $A_{i,j}$ that does not change during the course of the identification. This way, we will be able to reuse work already done by the algorithm. While this new criterion is equivalent to edgewise HTC, it is crucial for speeding up the algorithm. We start with the following lemma:

Lemma 1. *Let $M = (V, D, B)$ be a non-empty mixed graph, and let $Y, W \subseteq V$ with $|Y| = |W|$. Then, the following two conditions are equivalent:*

- *There exists a half-trek system from Y to W with no sided intersection.*
- *The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in \mathrm{pa}(y_i)} \sigma_{h,w_j}\lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \ldots, y_k\}$ and $W = \{w_1, \ldots, w_k\}$.*

Proof. $\Rightarrow$: This follows from Lemma 4.3 in Weihs et al. [2018].

$\Leftarrow$: For the proof, we model the graph M by a directed graph $M' = (V', D')$ with $V' = V \cup Y'$, where $Y' := \{y'_1, \dots, y'_k\}$ and the y'_i are fresh nodes, and

$$D' = D \cup \{y'_i \to y_i \mid i \in \{1, \dots, k\}\} \cup \{y'_i \to v \mid y_i \leftrightarrow v \in B\}.$$

Each edge in D' is labeled with an indeterminate: Edges $u \to v \in D$ are labeled with $\lambda_{u,v}$ and edges $y'_i \to v$ are labeled with $\omega_{y_i,v}$ (edges $y'_i \to y_i$ are a special case of this). For walks W in M', we denote by $W(\Lambda, \Omega)$ the product of the labels of all edges on the walk. By $\mathscr{W}(u, v)$, we denote the set of all walks from u to v in M'. Evidently, for all $i, j \in \{1, \dots, k\}$, the half-treks from y_i to w_j in M and the walks from y'_i to w_j in M' are in one-to-one correspondence, and given a half-trek π corresponding to a walk W, $\pi(\Lambda, \Omega) = W(\Lambda, \Omega)$. Moreover, two half-treks $\pi \in \mathscr{H}(y_i, w_j)$ and $\pi' \in \mathscr{H}(y_{i'}, w_{j'})$ have sided intersection if and only if their corresponding walks $W \in \mathscr{W}(y'_i, w_j)$ and $W' \in \mathscr{W}(y'_{i'}, w_{j'})$ intersect.

By the trek rule, it holds that

$$A_{i,j} = \sum_{\pi \in \mathscr{H}(y_i, w_j)} \pi(\Lambda, \Omega) = \sum_{W \in \mathscr{W}(y'_i, w_j)} W(\Lambda, \Omega).$$

The proof is by contraposition: Assuming that there is no half-trek system from Y to W with no sided intersection, we show that A is not generically invertible. From this assumption, it follows that there is no non-intersecting system of walks from Y' to W in M'. By Menger's theorem, see e.g. Göring [2000], this implies that there is a set $C \subseteq V'$ of size $|C| < k$ such that all walks from $y'_i \in Y'$ to $w_j \in W$ visit some node of $C = \{c_1, \dots, c_{|C|}\}$. Writing $\mathrm{Frac}(\mathbb{Z}[\Lambda \cup \Omega])$ for the field of fractions of $\mathbb{Z}[\Lambda \cup \Omega]$, we define $L \in \mathrm{Frac}(\mathbb{Z}[\Lambda \cup \Omega])^{k \times |C|}$ and $R \in \mathrm{Frac}(\mathbb{Z}[\Lambda \cup \Omega])^{|C| \times k}$ as

$$L_{i,j} = \sum_{W \in \mathscr{W}(y'_i, c_j), \text{the first node from } C \text{ in } W \text{ is } c_j} W(\Lambda, \Omega), \quad \text{and} \quad R_{i,j} = \sum_{W \in \mathscr{W}(c_i, w_j)} W(\Lambda, \Omega).$$

By splitting all walks from y'_i to w_j at the first node from C they visit, we get $A_{i,j} = \sum_{l=1}^{|C|} L_{i,l} R_{l,j}$, i.e., $A = L \cdot R$. It follows that

$$\mathrm{rank}(A) = \mathrm{rank}(L \cdot R) \leq \min(\mathrm{rank}(L), \mathrm{rank}(R)) \leq |C| < k,$$

so A is not generically invertible. □

Lemma 1 is a half-trek analogue of the trek separation theorems of Sullivant et al. [2010], Draisma et al. [2013]. The proof of the "⇐"-direction uses the same Menger-type factorization argument. Lemma 1 implies that the following theorem is also equivalent to Theorem 4.1 in Weihs et al. [2018] and can be used to check the edgewise half-trek criterion.

Theorem 1. *Let $M = (V, D, B)$ be a non-empty mixed graph and let $v \in V$. Let $W \subseteq \mathrm{pa}(v)$. Suppose there exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*

1. *The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in \mathrm{pa}(y_i)} \sigma_{h,w_j} \lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \dots, y_k\}$ and $W = \{w_1, \dots, w_k\}$, and*
2. *for every trek π from $y \in Y$ to v, we have that either*
   (a) *π ends with an edge of the form $s \to v$ where either $s \in W$ or $s \to v$ is known to be generically identifiable, or*
   (b) *π begins with an edge of the form $y \leftarrow h$ where $h \to y$ is known to be generically identifiable.*

Then for each $w \in W$, $w \to v$ is generically identifiable.

Proof. By Lemma 1, these conditions are equivalent to the ones of Weihs et al. [2018, Thm 4.1]. □

Remark 1. *While we subtract $\sigma_{h,w_j} \lambda_{h,y_i}$ for all $h \in \mathrm{pa}(y_i)$ in the definition of A, Weihs et al. [2018, Theorem 4.1] (see also Proposition 1 above) only subtract this for those $h \in \mathrm{pa}(y_i)$ such that $h \to y_i$ is known to be generically identifiable. Compared to Lemma 1, invertibility of their matrix $A^v_{Y,W}$*

only implies the existence of a trek system from Y to W with no sided intersection, which has to be restricted to a half-trek system from some Y' to W. We use this particular definition of $A^v_{Y,W}$ in Theorem 1, because it is constant—i.e., independent of which edges have already been identified.

In an ArXiv version of their paper, Weihs et al. [2017, Theorem 4.1] show a weaker version of the edgewise criterion. In the proof of this weaker theorem, they use yet another definition of the matrix $A^v_{Y,W}$ in which they subtract the mentioned term for all $h \in \mathrm{pa}(y_i) \cap \mathrm{htr}(v)$. Showing the equivalence of a criterion checking the invertibility of this matrix $A^v_{Y,W}$ actually requires the more restrictive condition (ii) of the weaker theorem. See Appendix C for a discussion of alternative sufficient criteria for edgewise generic identifiability.

4 A faster algorithm to check the half-trek criterion

In this section, we describe a randomized algorithm to check the half-trek criterion (Algorithm 1). The half-trek criterion [Foygel et al., 2012, Theorem 1] is the special case $W = \mathrm{pa}(v)$ of the edgewise criterion (Theorem 1). Thus, for each node, it either identifies all or none of its ingoing directed edges. We call the nodes for which we have already identified all of their ingoing directed edges *identified*, and all other nodes *unidentified*. Checking the half-trek criterion means repeatedly finding an unidentified v for which a Y exists that satisfies the conditions of Theorem 1 for $W = \mathrm{pa}(v)$. If we apply Theorem 1 with such a Y, all ingoing directed edges of v, and hence also v, become identifiable. Given $W = \mathrm{pa}(v)$, for a trek π from $y \in V$ to v, condition 2 of Theorem 1 is trivially satisfied by option (a) unless π is of the form $y \leftarrow \cdots \leftarrow v$ or $y \leftarrow \cdots \leftrightarrow v$. Such treks exist if and only if $y \in \mathrm{htr}(v)$, so condition 2 only has to be checked for $y \in \mathrm{htr}(v)$. For these y, condition 2 only becomes true for all treks π from y to v once y is identified. For our algorithm, this means that for each unidentified v, there is a set $I_v \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ of y that satisfy condition 2. Initially, I_v only contains the $y \notin \mathrm{htr}(v)$. During the identification, the $y \in \mathrm{htr}(v)$ are added to I_v once they are identified. This way, we have reduced the check for a given v to checking whether a $Y \subseteq I_v$ with $|Y| = |\mathrm{pa}(v)|$ exists such that the matrix $A^v_{Y,\mathrm{pa}(v)}$ given by $A^v_{y,w} = \sigma_{y,w} - \sum_{h \in \mathrm{pa}(y)} \sigma_{h,w}\lambda_{h,y}$ is generically invertible, i.e., the determinant of $A^v_{Y,\mathrm{pa}(v)}$ is generically nonzero. However, given that $\Sigma = (I - \Lambda)^{-T}\Omega(I - \Lambda)^{-1}$, the $\sigma_{u,v}$ are rational functions whose numerators and denominators can have exponentially many monomials, so it is not even feasible to compute the matrices $A^v_{Y,\mathrm{pa}(v)}$ explicitly.

4.1 Randomized linear algebra and error estimation

This is where randomization comes into play: Testing whether a determinant is nonzero is a special case of the so-called *polynomial identity testing problem (PIT)*, which has an efficient randomized algorithm. Instead of expanding the determinant as a polynomial, we evaluate it at a random integer point, which can be done in polynomial time due to its implicit representation as an arithmetic circuit. If the determinant is zero, then its value at the chosen random point is definitely zero, and if the determinant is nonzero, then the value at the chosen random point is also nonzero with a high probability. This is justified by the Schwartz–Zippel lemma:

Lemma 2 (Schwartz–Zippel Lemma, see e.g. Shpilka and Yehudayoff, 2010, Lemma 4.2)**.** *Let f be a nonzero polynomial of degree $\leq d$ in n variables. If we draw a vector $x \in S^n$ uniformly at random for some finite set S, then* $\Pr[f(x) = 0] \leq \frac{d}{|S|}$.

For all v, the determinant of $A^v_{Y,\mathrm{pa}(v)}$ is a polynomial in the $\lambda_{i,j}$ and $\omega_{i,j}$. When implemented naively, we have to pick new random values for the $\lambda_{i,j}$ and $\omega_{i,j}$ in each identity test of the determinants, and recompute the $\sigma_{i,j}$ in terms of these new random values. We can save this effort by using the same random values for all identity tests, at the cost of choosing them from a slightly larger set. A similar idea has already been used in Briefs and Bläser [2025].

Lemma 3 (Generalized Schwartz–Zippel, Briefs and Bläser, 2025, Lemma 3)**.** *Let $f_1, \ldots, f_\ell$ be nonzero polynomials of degree $\leq d_\lambda$, $1 \leq \lambda \leq \ell$, and in n variables each. If we draw $x \in S^n$ uniformly at random from a finite set S, then* $\Pr[f_\lambda(x) = 0$ *for at least one* $1 \leq \lambda \leq \ell] \leq \frac{d_1 + \cdots + d_\ell}{|S|}$.

Although our tests now operate on integers rather than on polynomials with exponentially many monomials, these integers can still become very large. Their bit size is still polynomial, but they would

Algorithm 1 Faster HTC

Input: A mixed graph $M = (V, D, B)$
Output: All directed edges identifiable by the half-trek criterion

Choose a random prime P and instantiate all $\omega_{u,v}$ and $\lambda_{u,v}$ with random values in $\{0, \dots, P-1\}$.
Compute $\Sigma := (I - \Lambda)^{-T}\Omega(I - \Lambda)^{-1}$ using these values.
For each $v \in V$, compute $\mathrm{htr}(v)$ by DFS.
For $v \in V$, $y \in V \setminus (\{v\} \cup \mathrm{sib}(v))$ and $w \in \mathrm{pa}(v)$, compute $A^v_{y,w} := \sigma_{y,w} - \sum_{h \in \mathrm{pa}(y)} \sigma_{h,w}\lambda_{h,y}$.
Initialize $I_v := \emptyset$ for all $v \in V$, and initialize $Q := \emptyset$.
for $y, v \in V$ such that $y \notin \{v\} \cup \mathrm{sib}(v)$, $\mathrm{rank}(A^v_{I_v,\mathrm{pa}(v)}) < |\mathrm{pa}(v)|$ and $y \notin \mathrm{htr}(v)$ **do**
    $I_v := I_v \cup \{y\}$, add row $A^v_{y,\mathrm{pa}(v)}$ to $A^v_{I_v,\mathrm{pa}(v)}$ and bring it into row echelon form again.
for $v \in V$ such that $\mathrm{rank}(A^v_{I_v,\mathrm{pa}(v)}) = |\mathrm{pa}(v)|$ **do** ▷ This includes nodes v with $\mathrm{pa}(v) = \emptyset$
    Q.push(v)
while $Q \neq \emptyset$ **do**
    $y := Q$.pop()
    Output all ingoing edges $s \to y$ of y as identified
    for $v \in V$ such that $y \notin \{v\} \cup \mathrm{sib}(v) \cup I_v$ and $\mathrm{rank}(A^v_{I_v,\mathrm{pa}(v)}) < |\mathrm{pa}(v)|$ **do**
        $I_v := I_v \cup \{y\}$, add row $A^v_{y,\mathrm{pa}(v)}$ to $A^v_{I_v,\mathrm{pa}(v)}$ and bring it into row echelon form again.
        if $\mathrm{rank}(A^v_{I_v,\mathrm{pa}(v)}) = |\mathrm{pa}(v)|$ **then**
            Q.push(v)

not fit into the registers of a Word-RAM (for the theoretical analysis, see e.g. Cormen et al. [2022] for a formal definition) and we would have to use big integer libraries (in the actual implementation). This would nullify the runtime gain. To be able to perform all computations on fixed-sized integers, we choose a random prime P at the start of our algorithm and perform all computations modulo P. The following lemma shows that if the determinant is nonzero at the chosen random point, then it is also nonzero modulo P with high probability. A proof can be found in Schönhage [1979], but we also give one in Appendix D for completeness.

Lemma 4. *Let z be a nonzero integer. If we choose a prime P uniformly at random from a set N of primes each larger than $B > 1$, then* $\Pr(P \text{ divides } z) \leq \log_2(|z|)/(|N| \cdot \log_2(B))$.

Theorem 2. *Algorithm 1 can be implemented using numbers with $\mathcal{O}(\log n)$ bits, i.e., on a Word-RAM, with an error probability tending to 0 as n goes to infinity.*

The proof is in Appendix D. While Theorem 2 analyzes the error probability from a theoretical perspective (on a Word-RAM), in Appendix B, we derive explicit error bounds for concrete values of n and using built-in integers, as these are important for the practical application of our algorithm. For $n = 1000$, for instance, the error probability is smaller than $5.68 \cdot 10^{-8}$. These estimates are worst-case estimates; we encountered no errors when comparing against SEMID during benchmarking.

4.2 Correctness and running time

Now, the remaining problem we have to solve is how to pick v and $Y \subseteq I_v$. For a fixed v, to check if a suitable Y exists, we consider the matrix $A^v_{I_v,\mathrm{pa}(v)}$. A set $Y \subseteq I_v$ such that $A^v_{Y,\mathrm{pa}(v)}$ is generically invertible exists if and only if the matrix $A^v_{I_v,\mathrm{pa}(v)}$ has full rank generically. Throughout, "full rank" means full column rank. As explained earlier, during the identification, I_v grows, so the rank of $A^v_{I_v,\mathrm{pa}(v)}$ increases gradually. Instead of recomputing the rank every time we add a $y \in V$ to I_v, i.e., append a row to $A^v_{I_v,\mathrm{pa}(v)}$, we maintain $A^v_{I_v,\mathrm{pa}(v)}$ incrementally in row echelon form by Gaussian elimination. The comparisons performed during Gaussian elimination are also instances of polynomial identity testing.

Our algorithm explores the HTC-identifiable nodes in a BFS-like fashion. To this end, it maintains a queue of identified nodes. Once it detects that some $A^v_{I_v,\mathrm{pa}(v)}$ has full rank, it pushes v to the queue. Whenever it pops some y from the queue, it adds y to all the I_v for which $y \in \mathrm{htr}(v) \setminus (\{v\} \cup \mathrm{sib}(v))$. If any of the $A^v_{I_v,\mathrm{pa}(v)}$ attains full rank after adding y to I_v, it adds v to the queue. This way, it discovers all nodes that are identifiable by the half-trek criterion. Algorithm 1 summarizes this algorithm.

Theorem 3. *Algorithm 1 correctly computes the half-trek criterion with high probability.*

Algorithm 2 Extension to the edgewise HTC

8: For all v, W such that $A^v_{I_{v,W},W}$ has full rank, add the pair (v, W) to Q.
9: **while** Q is nonempty **do**
10: $(v, W) = Q.\texttt{pop()}$
11: If all $w \to v$ for $w \in W$ are already identified, continue.
12: Mark $w \to v$ for $w \in W$ as identified.
13: $y := v$. Iterate over $v' \in V$, check for which subsets $W' \subseteq \mathrm{pa}(v')$ y became eligible now, and integrate y's row into the respective row echelon forms. Add (v', W') to Q if the rank became full.
14: For each $W' \subseteq \mathrm{pa}(v)$ determine all y that became eligible by identifying the $w \to v$ for $w \in W$ and integrate the rows into the respective row echelon forms. Add (v, W') to Q if the rank became full.

Theorem 4. *Algorithm 1 can be implemented to run in $\mathcal{O}(n^3 + n^2 m_2)$ where $n = |V|$, $m_1 = |B|$, and $m_2 = |D|$. If the directed part of the given mixed graph is sparse, i.e., $m_2 \in \mathcal{O}(n)$, then it runs in $\mathcal{O}(n^3)$.*

The proof of Theorem 3 is in Appendix D. Theorem 4 is proven in Appendix A.2, which provides a detailed running time analysis.

4.3 Comparison to SEMID

Let us explain in more detail how we achieve the savings compared to SEMID. Figure 2 shows a small SCM. The nodes of the SCM can be identified as follows using Theorem 1: Node 0 is identified immediately (it has no ingoing edges). Node 1 is identified by $Y = \{0\}$ immediately ($0 \notin \mathrm{htr}(1)$). Node 2 is identified by $Y = \{0\}$ or $Y = \{1\}$ immediately ($0, 1 \notin \mathrm{htr}(2)$) as well as node 3 ($0, 1 \notin \mathrm{htr}(3)$). Node 4 is more complicated: Nodes 0 and 1 are eligible immediately ($0, 1 \notin \mathrm{htr}(4)$), but the rank is only 1 with these two rows because all half-trek systems intersect. $2 \in \mathrm{sib}(4)$, so it is not eligible at all. Node 3 is only eligible after it is identified because $3 \in \mathrm{htr}(4)$. With this third row, the rank will be full because there is a half-trek system with no sided intersection.

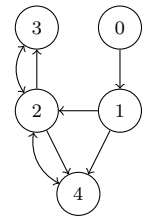

Figure 2: Example.

In SEMID, it can happen that the algorithm tries to identify 4 before 3. Then, the whole max-flow computation is wasted and has to be redone once 3 is identified. In our algorithm, even if we try to identify 4 first, we will add a row to the corresponding matrix and the result of the work gets stored and can be used later. Therefore, the order in which we try to identify 3 and 4 does not matter.

4.4 Edgewise HTC

SEMID additionally implements an extension of the HTC algorithm to the edgewise HTC. This is achieved by iterating over all nonempty subsets $W \subseteq \mathrm{pa}(v)$, instead of only considering the subset $W = \mathrm{pa}(v)$ that contains all parents. SEMID also has a control parameter ℓ, which restricts iteration to subsets W whose size or co-size is at most ℓ. (W has co-size i if its size is $|\mathrm{pa}(v)| - i$.) In this way, runtime can be balanced against identification power.

Our algorithm can be extended to the edgewise HTC in a similar way: Instead of one matrix per node, we now maintain a matrix in row echelon form per node and allowed subset of parents. Further, instead of maintaining the set of eligible y per node (stored in the set I_v), we now maintain the set of eligible y per node and considered subset of parents (now stored in a set $I_{v,W}$). According to Theorem 1, a node y is eligible for a node v and a subset of v's parents W if all treks from y to v either end in edges $w \to v$ that are already identified or for which $w \in W$, or begin in identified edges $y \leftarrow h$. This condition can be checked by the following case distinction: If y's parent $h \in \mathrm{htr}(v)$, then y is only eligible once $h \to y$ is identified. If v's parent $w \in \mathrm{htr}(y)$, then y is only eligible if $w \in W$ or once $w \to v$ is identified. Further, we consider all pairs of y's parents h and v's parents w. If there is a trek between h and w, then y is only eligible if $w \in W$, or once $h \to y$ or $w \to v$ is identified. Similarly to how we maintain the I_v in Algorithm 1, we can incrementally maintain the $I_{v,W}$ by re-checking this condition whenever a new edge is identified. In Algorithm 1, lines 1 to 4 remain unchanged for the edgewise case. Analogously to lines 5 to 7, we initialize the $I_{v,W}$

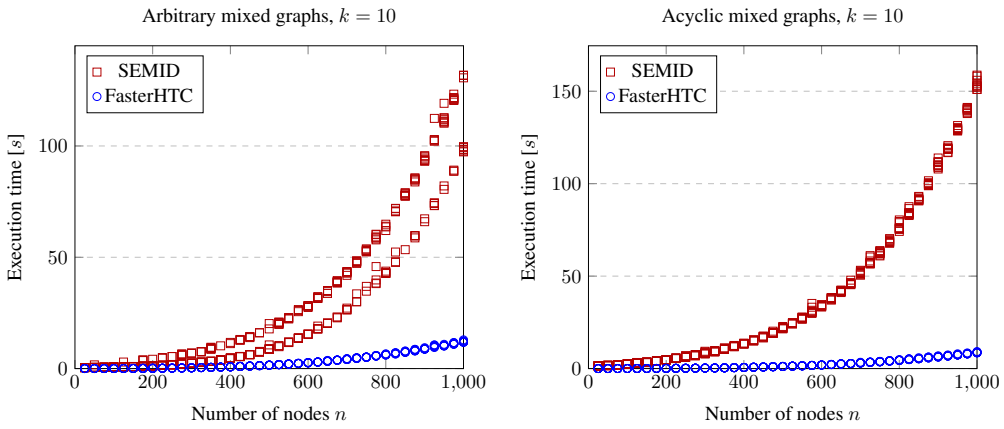

Figure 3: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 10$. The left plot shows the results for the arbitrary mixed graphs. In some of these graphs, all directed edges can be identified by the half-trek criterion, whereas in the rest of these graphs, none of the directed edges can be identified by the half-trek criterion. While the execution time of our program is less sensitive to whether the half-trek criterion succeeds, SEMID takes significantly longer when the half-trek criterion succeeds, because work needs to be redone. This can be observed by the presence of two separate curves for SEMID. The upper one corresponds to graphs for which the half-trek criterion succeeds, the lower one corresponds to graphs for which the half-trek criterion fails. The right plot shows the results for the acyclic mixed graphs. The half-trek criterion succeeds on all of these graphs.

according to the previously described condition and add the rows for the y that are initially in $I_{v,W}$. The pseudocode for the remaining algorithm can be found in Algorithm 2.

Lemma 5. *Algorithm 2 can be implemented to run in $\mathcal{O}(2^d(n^3 + n^2 m_2))$, where d is the maximum indegree in the graph. This is $\mathcal{O}(2^d n^4)$ for dense and $\mathcal{O}(2^d n^3)$ for sparse graphs.*

Lemma 6. *Algorithm 2 can be implemented using numbers with $\mathcal{O}(\log n)$ bits with an error probability tending to 0 as n goes to infinity. More precisely, we only need to pass d times more identity tests successfully compared to Algorithm 1.*

At first glance, one might expect a factor of 2^d here. However, our algorithm has a one-sided error. When it identifies, it is always right. Therefore, we only have to pass the tests when an incoming edge is identified for the first time. We can get precise estimates for Algorithm 2, similar to those for Algorithm 1. For maximum indegree $d \leq 10$, the error probability is smaller than $1.07 \cdot 10^{-8}$, $1.40 \cdot 10^{-6}$ and $1.73 \cdot 10^{-4}$ for $n = 1\,000$, $10\,000$ and $100\,000$, respectively. (These bounds are even better than the one for Algorithm 1 given in Appendix B, because we assume a bound on d.) Without any assumption on d, it is smaller than $1.07 \cdot 10^{-4}$ for $n = 1\,000$.

5 Benchmarks

To show the efficiency of our algorithm, we implemented it in C++[1] and benchmarked it against the implementation of the half-trek criterion in the R package SEMID [Barber et al., 2026] (htcID). Inspired by Foygel et al. [2012], we generated random mixed graphs and acyclic mixed graphs as follows. For each $k \in \{1, \ldots, 10\}$ and each n between 25 and 1 000 (in steps of 25), we generated 10 graphs with n nodes for which we selected each possible edge with a probability of $\frac{k \cdot n}{3 \cdot n \cdot (n-1)/2}$ for mixed graphs and $\frac{k \cdot n}{2 \cdot n \cdot (n-1)/2}$ for acyclic mixed graphs, resulting in graphs in which the expected total number of edges is $k \cdot n$ and the expected degree of each node is $2k$. We executed both SEMID and our program on all generated mixed graphs. The sets of identified directed edges by both programs were equal for all graphs. All experiments were carried out on a Debian Linux server equipped with an AMD EPYC 7702 64-core processor running at 3.35GHz and an overall memory of 2TB. Figure 3 shows plots of the execution times for $k = 10$. Plots for $k \in \{1, \ldots, 9\}$ can be found in Appendix E.

[1] Our implementation is available on GitHub [Briefs and Bläser, 2026].

Acknowledgments and Disclosure of Funding

The authors thank the anonymous reviewers for their helpful comments. This work was inspired by discussions that MB had while visiting the Simons Institute for the Theory of Computing as a participant in the "Complexity and Linear Algebra" program.

References

Rina Foygel Barber, Mathias Drton, Miriam Kranzlmüller, Luca Weihs, and Nils Sturma. *SEMID: Identifiability of Linear Structural Equation Models*, 2026. URL https://cran.r-project.org/package=SEMID. R package version 0.5.1.

Elias Bareinboim, Juan D. Correa, Duligur Ibeling, and Thomas Icard. *On Pearl's Hierarchy and the Foundations of Causal Inference*, pages 507–556. Association for Computing Machinery, New York, NY, USA, 2022. URL https://doi.org/10.1145/3501714.3501743.

Kenneth A. Bollen. *Structural equations with latent variables*. Wiley, 1989. URL https://doi.org/10.1002/9781118619179.

Roger J. Bowden and Darrell A. Turkington. *Instrumental Variables*. Cambridge University Press, 1990. URL https://doi.org/10.1017/CCOL0521262410.

Yasmine Briefs and Markus Bläser. Faster generic identification in tree-shaped structural causal models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 89365–89393. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/80f48ffa8022773973a4a5cec7cce19c-Paper-Conference.pdf.

Yasmine Briefs and Markus Bläser. C++ implementation FasterHTC. GitHub, 2026. URL https://github.com/yasminebriefs/FasterHTC/.

Bryant Chen. Identification and overidentification of linear structural equation models. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1579–1587, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/49c9adb18e44be0711a94e827042f630-Abstract.html.

Bryant Chen, Judea Pearl, and Elias Bareinboim. Incorporating knowledge into structural equation models using auxiliary variables. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 3577–3583. AAAI Press, 2016. URL https://dl.acm.org/doi/10.5555/3061053.3061120.

Bryant Chen, Daniel Kumor, and Elias Bareinboim. Identification and model testing in linear structural equation models using auxiliary variables. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 757–766. JMLR.org, 2017. URL https://dl.acm.org/doi/10.5555/3305381.3305460.

Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. *Journal of the ACM*, 72 (3), 2025. URL https://doi.org/10.1145/3728631.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 4th edition, 2022. ISBN 9780262046305.

Julian Dörfler, Benito van der Zander, Markus Bläser, and Maciej Liskiewicz. On the complexity of identification in linear structural causal models. In Amir Globerson, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/b588d9b67932b459ea66ff6e2804c6b3-Abstract-Conference.html.

Jan Draisma, Seth Sullivant, and Kelli Talaska. Positivity for Gaussian graphical models. *Advances in Applied Mathematics*, 50(5):661–674, 2013.

Mathias Drton. Algebraic problems in structural equation modeling. In *The 50th anniversary of Gröbner bases*, pages 35–86. Mathematical Society of Japan, 2018. URL https://doi.org/10.2969/aspm/07710035.

Mathias Drton, Rina Foygel, and Seth Sullivant. Global identifiability of linear structural equation models. *The Annals of Statistics*, 39(2):865–886, 2011. URL https://doi.org/10.1214/10-AOS859.

Otis Dudley Duncan. *Introduction to structural equation models*. Academic Press, 1975. URL https://doi.org/10.1016/C2009-0-03148-1.

Rina Foygel, Jan Draisma, and Mathias Drton. Half-trek criterion for generic identifiability of linear structural equation models. *The Annals of Statistics*, 40(3):1682–1713, 2012. URL https://doi.org/10.1214/12-AOS1012.

Luis D. García-Puente, Sarah Spielvogel, and Seth Sullivant. Identifying causal effects with computer algebra. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, UAI'10, page 193–200, Arlington, Virginia, USA, 2010. AUAI Press. URL https://dl.acm.org/doi/10.5555/3023549.3023572.

Frank Göring. Short proof of Menger's theorem. *Discret. Math.*, 219(1-3):295–296, 2000. URL https://doi.org/10.1016/S0012-365X(00)00088-1.

Torbjörn Granlund and the GMP development team. GNU MP: The GNU multiple precision arithmetic library, 2023. URL https://gmplib.org. Version 6.3.0.

Aaryan Gupta and Markus Bläser. Identification for tree-shaped structural causal models in polynomial time. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 20404–20411. AAAI Press, 2024. URL https://doi.org/10.1609/aaai.v38i18.30023.

Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53 (2):217–288, 2011. URL https://doi.org/10.1137/090771806.

Benjamin Hollering, Pratik Misra, and Nils Sturma. Efficient symbolic computations for identifying causal effects, 2026. URL https://arxiv.org/abs/2604.20516.

Daniel Kumor, Bryant Chen, and Elias Bareinboim. Efficient identification in linear structural causal models with instrumental cutsets. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 12477–12486, 2019. URL https://openreview.net/forum?id=r1GHqSrx8S.

Daniel Kumor, Carlos Cinelli, and Elias Bareinboim. Efficient identification in linear structural causal models with auxiliary cutsets. In *Proc. Int. Conference on Machine Learning (ICML)*, pages 5501–5510, 2020. URL https://proceedings.mlr.press/v119/kumor20a.html.

Michael W. Mahoney. Randomized algorithms for matrices and data. *Found. Trends Mach. Learn.*, 3 (2):123–224, February 2011. ISSN 1935-8237. URL https://doi.org/10.1561/2200000035.

Per-Gunnar Martinsson and Joel A. Tropp. Randomized numerical linear algebra: Foundations and algorithms. *Acta Numerica*, 29:403–572, 2020. URL https://doi.org/10.1017/S0962492920000021.

OpenSSL Project. OpenSSL: The open source toolkit for SSL/TLS, 2026. URL https://www.openssl.org. Version 3.6.1.

Judea Pearl. *Causality*. Cambridge University Press, 2009. ISBN 9780521895606.

Jerzy S. Respondek. *Fast Matrix Multiplication with Applications*, volume 166 of *Studies in Big Data*. Springer Nature Switzerland, 2025. URL https://doi.org/10.1007/978-3-031-76930-6.

Arnold Schönhage. On the power of random access machines. In Hermann A. Maurer, editor, *Automata, Languages and Programming, 6th Colloquium, Graz, Austria, July 16-20, 1979, Proceedings*, volume 71 of *Lecture Notes in Computer Science*, pages 520–529. Springer, 1979. URL https://doi.org/10.1007/3-540-09510-1_42.

Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends® in Theoretical Computer Science*, 5(3–4):207–388, 2010. URL https://doi.org/10.1561/0400000039.

Seth Sullivant, Kelli Talaska, and Jan Draisma. Trek separation for Gaussian graphical models. *The Annals of Statistics*, 38(3):1665–1685, 2010.

Jin Tian. Identifying direct causal effects in linear models. In Manuela M. Veloso and Subbarao Kambhampati, editors, *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pages 346–353. AAAI Press / The MIT Press, 2005. URL http://www.aaai.org/Library/AAAI/2005/aaai05-055.php.

Benito van der Zander, Marcel Wienöbst, Markus Bläser, and Maciej Liskiewicz. Identification in tree-shaped linear structural causal models. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, volume 151 of *Proceedings of Machine Learning Research*, pages 6770–6792. PMLR, 2022. URL https://proceedings.mlr.press/v151/van-der-zander22a.html.

Luca Weihs, Bill Robinson, Emilie Dufresne, Jennifer Kenkel, Kaie Kubjas, Reginald McGee II, Nhan Nguyen, Elina Robeva, and Mathias Drton. Determinantal generalizations of instrumental variables, 2017. URL https://arxiv.org/abs/1702.03884.

Luca Weihs, Bill Robinson, Emilie Dufresne, Jennifer Kenkel, Kaie Kubjas, Reginald McGee II, Nhan Nguyen, Elina Robeva, and Mathias Drton. Determinantal generalizations of instrumental variables. *Journal of Causal Inference*, 6(1):20170009, 2018. URL https://doi.org/10.1515/jci-2017-0009.

Sewall Wright. The method of path coefficients. *The Annals of Mathematical Statistics*, 5(3):161–215, 1934. URL https://doi.org/10.1214/aoms/1177732676.

A Further details on the algorithm

A.1 Decomposition of mixed graphs

Given the mixed graph $M = (V, D, B)$, instead of checking the half-trek criterion directly for M, the algorithm implemented in the R package SEMID [Barber et al., 2026] decomposes M into its so-called Tian components [Tian, 2005] and checks the half-trek criterion for all components separately. Besides the subproblems potentially being smaller, and identification therefore potentially being faster, the Tian decomposition has the benefit of actually strengthening the half-trek criterion. While any edge that is HTC-identifiable in M is also HTC-identifiable in its Tian component, the converse is not true. As the example in Figure 9 in Foygel et al. [2012] shows, it is possible that there are edges that are only HTC-identifiable in their Tian components, but not in M. Our implementation, like SEMID, first performs a Tian decomposition and then treats each component separately.

The Tian decomposition $M_1 = (V_1, D_1, B_1), \ldots, M_k = (V_k, D_k, B_k)$ of a mixed graph $M = (V, D, B)$ is defined as follows: Let $C_1, \ldots, C_k \subseteq V$ be the pairwise disjoint sets in which two nodes are in the same C_i if and only if they are in the same strongly connected component w.r.t. D or the strongly connected components they belong to are connected via bidirected edges. Then

$$\begin{aligned} V_i &= C_i \cup \{s \in \mathrm{pa}(v) \mid v \in C_i\}, \\ D_i &= \{(u \to v) \in D \mid v \in C_i\}, \text{ and} \\ B_i &= \{(u \leftrightarrow v) \in B \mid u, v \in C_i\}. \end{aligned}$$

This means that each directed and bidirected edge is in exactly one Tian component, whereas nodes can occur in multiple Tian components. To be comparable to SEMID, we also compute the Tian decomposition and run our algorithm on each Tian component separately. Due to the fact that each node can be in multiple components, the asymptotic running time of certain steps of our algorithm increases by the Tian decomposition, but the asymptotic running time of the whole algorithm stays the same. In practice, however, we expect that the Tian decomposition makes our algorithm faster.

A.2 The implemented algorithm and its running time

In this section, we discuss the implementation of Algorithm 1 and the running time of its individual steps. In the following, n is the number of nodes, m_1 is the number of bidirected edges and m_2 is the number of directed edges. The following computations assume that all arithmetic operations, including computing inverses modulo P, take constant time.

While in the Tian decomposition, each edge is part of exactly one Tian component, single nodes can be in multiple components. In each Tian component, we distinguish basic and non-basic nodes, where the non-basic nodes are the ones that were only added because they are the parents of basic nodes in the component. While we have to consider both basic and non-basic nodes for the computation of Σ, we can treat them as special cases in the remaining steps: Non-basic nodes do not have any ingoing edges, so they are not reached by any other nodes by half-treks, nor do we need to compute A^v for them. The total number of nodes is bounded by $n + m_2$ because every node occurs as a basic node once, and can occur as a non-basic node only as many times as it has outgoing directed edges. For our algorithm, the worst case occurs when there are $(n + m_2)/n$ components each having n nodes.

- The first step is to compute the Tian decomposition. This involves computing strongly connected components w.r.t. the directed edges and using a union find data structure to merge the different strongly connected components that are connected by bidirected edges. This step is implemented in $\mathcal{O}(n^2 + nm_1 + nm_2)$. This could of course be done faster, but this is not the step that dominates the running time.
- We identify each Tian component separately. First, we generate random values for the Λ and Ω and compute Σ in terms of Λ and Ω by $\Sigma = (I - \Lambda)^{-T}\Omega(I - \Lambda)^{-1}$ (lines 1 and 2). This takes $\mathcal{O}(n^3)$ for a single component, or $\mathcal{O}((n + m_2)/n \cdot n^3) = \mathcal{O}(n^3 + n^2 m_2)$ for all components. This is actually the only step of our algorithm that may become asymptotically slower when using the Tian decomposition.
- The next step is to compute, for each node, all nodes reachable by half-treks. To achieve this, for all nodes v, we run a DFS that only considers directed edges (line 3). We start this DFS at v and at all siblings of v. This takes $\mathcal{O}(m_1 + n(n + m_2)) = \mathcal{O}(n^2 + m_1 + nm_2)$ for a single component. It also takes $\mathcal{O}((n + m_2)/n \cdot n^2 + m_1 + nm_2) = \mathcal{O}(n^2 + m_1 + nm_2)$ for all components because every edge occurs in exactly one component.
- It is enough to compute the rows $A^v_{y,\mathrm{pa}(v)}$ of the matrices A^v (line 4) once they are needed, i.e., once y is added to I_v (lines 7 and 14). The total length of all rows for a single y is at most m_2, so overall, there are at most nm_2 matrix entries that will be computed. Computing a single matrix entry $A^v_{y,w}$ for $w \in \mathrm{pa}(v)$ requires summing up $1 + |\mathrm{pa}(y)|$ terms, so computing all matrix entries requires summing up at most $\sum_y m_2 \cdot (1 + |\mathrm{pa}(y)|) = m_2(n + m_2)$ terms, which takes $\mathcal{O}(nm_2 + m_2^2)$ regardless of the Tian decomposition.
- We maintain each matrix $A^v_{I_v,\mathrm{pa}(v)}$ in row echelon form, discarding empty rows immediately. To maintain the row echelon form with new rows $A^v_{y,\mathrm{pa}(v)}$ added one by one (lines 7 and 14), we use Gaussian elimination: When adding a row, we iterate over the columns from left to right. If the entry is nonzero and another row already has a leading 1 in this column, we make the entry 0 using this other row. If the entry is nonzero and no other row has a leading 1 in this column, then we make the entry 1 by multiplying this row by the inverse of its leading entry. In the second case, we can stop iterating. To make leading entries zero, we visit each entry at most once per column of the matrix it is in, so as there are at most nm_2 entries and at most n columns per matrix, this takes $\mathcal{O}(n^2 m_2)$ regardless of the Tian decomposition. Potentially multiplying a row by an inverse to make its leading entry 1 happens at most once per row (and hence entry) and will take $\mathcal{O}(nm_2)$ regardless of the Tian decomposition.

- The next step is a preprocessing step in which we iterate over all pairs of y and v where $y \notin \{v\} \cup \mathrm{sib}(v)$ and check if $y \notin \mathrm{htr}(v)$ (line 6). If so, we can add y's row to v's matrix immediately (line 7), otherwise, we have to wait until we have identified y. Nodes y without ingoing edges are a special case of this. The running time of this step is negligible. Potentially added rows in this step are accounted for in the previous two items.
- In the last step, we maintain a queue that initially (lines 8 and 9) contains all nodes that were already identified during the preprocessing step (their matrix contains as many rows as it has columns). We iteratively visit the nodes y in the queue (lines 10 and 11), add their ingoing edges to the output (line 12) and add the respective rows to the v for which we could not add the row before because $y \in \mathrm{htr}(v)$ (lines 13 and 14). If some of the $A^v_{I_v,\mathrm{pa}(v)}$ get full rank by that, we add these v to the queue (lines 15 and 16). The running time of computing and adding the rows is already accounted for in the previous items, so the running time of this last step is negligible.

This means that the total running time of our algorithm is $\mathcal{O}(n^3 + n^2 m_2) \subseteq \mathcal{O}(n^4)$ regardless of the Tian decomposition. If the directed part of the graph is sparse, i.e., $m_2 \in \mathcal{O}(n)$, then the running time is $\mathcal{O}(n^3)$.

B Error estimation

Due to the use of polynomial identity testing, our algorithm is only correct with a very high probability. Only two steps of our algorithm involve polynomial identity testing: computing Σ in terms of Λ and Ω and checking if the matrices $A^v_{I_v,\mathrm{pa}(v)}$ have full rank. Computing Σ in terms of Λ and Ω fails when $\det(I - \Lambda) = 0$, which can easily be detected. A failure of the check if the matrices $A^v_{I_v,\mathrm{pa}(v)}$ have full rank can lead to a one-sided error, which means that we might fail to identify directed edges which are identifiable by the half-trek criterion. In the following, we estimate the probability with which our algorithm fails. All of the estimations are upper bounds.

Before starting the algorithm, we choose a random prime P of 59 bits, i.e., $2^{58} < P < 2^{59}$. By the prime number theorem, there are at least $6.9 \cdot 10^{15}$ such primes. The random prime is generated using OpenSSL [OpenSSL Project, 2026]. Alternatively, the GNU Multiple Precision Arithmetic Library (GMP) [Granlund and the GMP development team, 2023] can be used to generate the random prime. According to the OpenSSL specification, with a probability of 2^{-128}, it can happen that the generated number P is non-prime. Our motivation to choose a prime of 59 bits is that we perform all arithmetic operations on built-in 64-bit integers, which makes our algorithm very efficient in practice. Using larger primes requires using a big integer package.

All polynomials we consider are in the variables

$$\mathcal{V} = \{\lambda_{i,j} \mid (i \to j) \in D\} \cup \{\omega_{u,v} \mid u < v \wedge (u \leftrightarrow v) \in B\} \cup \{\omega_{u,u} \mid u \in V\}.$$

The matrices $\Lambda, \Omega \in \mathbb{Z}[\mathcal{V}]^{n \times n}$ are defined as

$$\Lambda_{i,j} = \begin{cases} \lambda_{i,j} & \text{if } (i \to j) \in D, \text{ and} \\ 0 & \text{otherwise, and as} \end{cases} \qquad \Omega_{u,v} = \begin{cases} \omega_{u,v} & \text{if } u < v \text{ and } u \leftrightarrow v \in B, \\ \omega_{v,u} & \text{if } u > v \text{ and } u \leftrightarrow v \in B, \\ \omega_{u,u} & \text{if } u = v, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

As the first step of our algorithm, we randomly choose an assignment $a : \mathcal{V} \to \mathbb{Z}_P$. We define $\mathrm{Frac}(\mathbb{Z}[\mathcal{V}]) = \{f/g \mid f, g \in \mathbb{Z}[\mathcal{V}] \wedge g \neq 0\}$. For a matrix $A \in \mathrm{Frac}(\mathbb{Z}[\mathcal{V}])^{r \times s}$, we denote by $A(a) \in \mathbb{Z}_P^{r \times s}$ the matrix obtained by evaluating each entry of A under a, provided all denominators evaluate to nonzero elements of $\mathbb{Z}_P$.

We compute $\Sigma \in \mathrm{Frac}(\mathbb{Z}[\mathcal{V}])^{n \times n}$ as

$$\Sigma = (I - \Lambda)^{-T} \Omega (I - \Lambda)^{-1} = \frac{1}{\det(I - \Lambda)^2} \operatorname{adj}(I - \Lambda)^T \Omega \operatorname{adj}(I - \Lambda).$$

The determinant $\det(I - \Lambda)$ is a nonzero polynomial of degree at most n and l_1 norm at most $n!$. Checking if the determinant $\det(I - \Lambda(a))$ is nonzero, too, corresponds to one identity test of degree at most n and l_1 norm at most $n!$. Each entry of $\operatorname{adj}(I - \Lambda)$ is the determinant of an

$(n-1) \times (n-1)$ submatrix of $(I - \Lambda)$. Hence, each entry of $\operatorname{adj}(I - \Lambda)$ is a polynomial of degree at most $(n-1)$ and l_1 norm at most $(n-1)!$. It follows that the $\Sigma_{u,v}$ are rational functions whose numerator is a polynomial of degree at most $(n-1) + 1 + (n-1) = 2n-1$ and l_1 norm at most $n \cdot (n-1)! \cdot (n \cdot 1 \cdot (n-1)!) = n!^2$ and whose denominator is $\det(I - \Lambda)^2$.

Given $v \in V$, each entry $A^v_{y,w}$ of $A^v_{I_v,\operatorname{pa}(v)}$ is a rational function whose numerator is a polynomial of degree at most $(2n-1)+1 = 2n$ and l_1 norm at most $n!^2 + (n-1)(n!^2 \cdot 1) = n(n!^2)$. All entries of the A^v can be written with the common denominator $\det(I - \Lambda)^2$. We have already verified that $\det(I - \Lambda(a))$ is nonzero. We add the rows to A^v one by one. As long as $A^v_{I_v,\operatorname{pa}(v)}$ does not have full rank, we detect this without any error. Assume $A^v_{I_v,\operatorname{pa}(v)}$ has full rank. We detect that $A^v_{I_v,\operatorname{pa}(v)}$ has full rank if and only if $A^v_{I_v,\operatorname{pa}(v)}(a) \in \mathbb{Z}_P^{|I_v| \times |\operatorname{pa}(v)|}$ has full rank. Here, it is worth noting that we use Gaussian elimination to check if $A^v_{I_v,\operatorname{pa}(v)}(a)$ has full rank. Gaussian elimination succeeds if and only if the rank of the matrix $A^v_{I_v,\operatorname{pa}(v)}(a)$ is full. To obtain simpler estimations, instead of estimating all identity tests that are performed during Gaussian elimination, we estimate an identity test whose success is sufficient for Gaussian elimination to succeed. $A^v_{I_v,\operatorname{pa}(v)}$ has full rank if and only if there exists a subset $Y \subseteq I_v$ such that $|Y| = |\operatorname{pa}(v)|$ and $\det(A^v_{Y,\operatorname{pa}(v)})$ is nonzero. If $\det(A^v_{Y,\operatorname{pa}(v)}(a)) \neq 0$, then the rank of $A^v_{I_v,\operatorname{pa}(v)}(a)$ is full and Gaussian elimination succeeds. For each $v \in V$, there is at most one I_v during the algorithm such that $A^v_{I_v,\operatorname{pa}(v)}$ has full rank, but does not have full rank without the most recently added row. Given that $|Y| = |\operatorname{pa}(v)| \leq n$, checking if the respective determinants are nonzero corresponds to at most n identity tests of degree at most $n \cdot 2n = 2n^2$ and l_1 norm at most $n!(n(n!^2))^n = n! \cdot n^n \cdot n!^{2n}$ each.

Let f be the product of all nonzero polynomials for which we perform identity tests. By the above, $\deg f \leq n + 2n^3$, and $\|f\|_1 \leq L$, where $L = n! \cdot (n! \cdot n^n \cdot n!^{2n})^n = n^{n^2} \cdot n!^{2n^2+n+1}$ is an upper bound on the product of their ℓ_1 norms, so $\log_2 \|f\|_1 \leq n^2 \log_2 n + (2n^2 + n + 1) \cdot n \log_2 n$. All identity tests succeed if $f(a) \not\equiv 0 \pmod{P}$. Note that a is chosen uniformly from $\{0, \ldots, P-1\}$ after P has been chosen, so we cannot apply Lemma 4 to the value $f(a)$, which depends on P. Instead, we fix a nonzero coefficient c of f, which does not depend on the random choices and satisfies $1 \leq |c| \leq \|f\|_1$.

The first source of error is that the chosen prime divides c. By Lemma 4, this happens with probability at most $\log_2 \|f\|_1 / (|N| \cdot \log_2(2^{58})) \leq \big(n^2 \log_2 n + (2n^2 + n + 1) \cdot n \log_2 n\big)/(6.9 \cdot 10^{15} \cdot 58)$.

The second source of error is that $P \nmid c$, but the chosen evaluation point is a root of $f \bmod P$. In this case, $f \bmod P$ is a nonzero polynomial over $\mathbb{Z}_P$ of degree at most $\deg f$, and a is uniform on $\mathbb{Z}_P^{\mathcal{V}}$, so by the Schwartz–Zippel lemma (Lemma 2), this happens with probability at most $\deg f / P \leq (n + 2n^3)/2^{58}$.

By the union bound, the overall error probability is bounded by the sum of the probability that we choose a non-prime, the probability that we choose a prime dividing c, and the probability that we choose a bad evaluation point. For $n = 1\,000$, this sum is smaller than $5.68 \cdot 10^{-8}$, and for $n = 10\,000$, it is smaller than $7.34 \cdot 10^{-5}$. The concrete numbers come from the fact that we perform all computations on built-in 64-bit integers. This is feasible up to $n \approx 100\,000$, for which the error probability is bounded by 0.091. By running the algorithm multiple times with new random primes and new random assignments, the error probability drops exponentially. For larger n, it is necessary to choose from a larger set of primes, which requires using a big integer package.

C Alternative sufficient criteria for edgewise generic identifiability

As mentioned in Remark 1, we use a definition for the matrix $A^v_{Y,W}$ we check for invertibility that is different from the definition used by Weihs et al. [2018, Theorem 4.1]. As Lemma 7 and Proposition 1 show, it is also possible to show an equivalent criterion based on the invertibility of the matrix used by Weihs et al. [2018]. However, as the example in Figure 4 shows, it is not possible to show a lemma as general as Lemma 1, stating that for any Y, the existence of a half-trek system with no sided intersection is equivalent to the invertibility of $A^v_{Y,W}$. Instead, we show that a Y that satisfies the conditions of the theorem and admits a half-trek system to W with no sided intersection exists if and only if a possibly different Y that satisfies the conditions of the theorem and for which the matrix $A^v_{Y,W}$ is invertible exists. By Lemma 4.3 in Weihs et al. [2018], the forward implication holds even

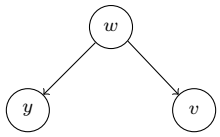

Figure 4: This example shows that under the conditions of Weihs et al. [2018, Theorem 4.1], the matrix $A^v_{Y,W}$ can be generically invertible even if no half-trek system from Y to W with no sided intersection exists. Let $Y = \{y\}$ and $W = \{w\}$ and suppose that no edges have been identified so far. The chosen Y satisfies the condition about treks from y to v and the matrix given by $A^v_{y,w} = \sigma_{y,w} = \lambda_{w,y}\omega_{w,w}$ is generically invertible, but no half-trek system from Y to W exists.

for the same Y, but for the converse implication, we only get that there is a trek system from Y to W with no sided intersection. Given that Y satisfies the conditions of the theorem, however, we can shorten the trek system to a half-trek system from a possibly different Y that satisfies the conditions.

Lemma 7. *Let $M = (V, D, B)$ be a non-empty mixed graph and let $v \in V$. Let $W \subseteq \mathrm{pa}(v)$. Then, the following are equivalent:*

- *There exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*
  1. *There exists a half-trek system from Y to W with no sided intersection,*
  2. *for every trek π from $y \in Y$ to v we have that either*
     (a) *π ends with an edge of the form $s \to v$ where either $s \in W$ or $s \to v$ is known to be generically identifiable, or*
     (b) *π begins with an edge of the form $y \leftarrow h$ where $h \to y$ is known to be generically identifiable.*
- *There exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*
  1. *The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in H_i} \sigma_{h,w_j}\lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \ldots, y_k\}$, $W = \{w_1, \ldots, w_k\}$ and $H_i = \{h \in \mathrm{pa}(y_i) \mid h \to y_i$ is known to be generically identifiable$\}$,*
  2. *for every trek π from $y \in Y$ to v we have that either*
     (a) *π ends with an edge of the form $s \to v$ where either $s \in W$ or $s \to v$ is known to be generically identifiable, or*
     (b) *π begins with an edge of the form $y \leftarrow h$ where $h \to y$ is known to be generically identifiable.*

Proof. $\Rightarrow$: As mentioned before, by Lemma 4.3 in Weihs et al. [2018], the forward implication holds even for the same Y.

$\Leftarrow$: For $i, j \in \{1, \ldots, k\}$, let $\mathscr{T}_{i,j}$ be the set of all treks from y_i to w_j that do not start in an edge $y_i \leftarrow h$ for $h \in H_i$. Evidently, $A_{i,j} = \sum_{\pi \in \mathscr{T}_{i,j}} \pi(\Lambda, \Omega)$. In contrast to the proof of Lemma 1, not all π in the sum are necessarily half-treks, so we need a slightly more complicated definition to model the treks in the $\mathscr{T}_{i,j}$ by walks in a directed graph $M' = (V', D')$. For this proof, V' contains a left and a right copy of each node in V. For all bidirected edges $(u \leftrightarrow v)$ in B, D' contains a directed edge from the left copy of u to the right copy of v, labeled with $\omega_{u,v}$. For all $v \in V$, D' contains a directed edge from the left copy of v to the right copy of v, labeled with $\omega_{v,v}$. For all directed edges $(u \to v) \in D$, D' contains a directed edge from the left copy of v to the left copy of u and from the right copy of u to the right copy of v, both labeled with $\lambda_{u,v}$. Additionally, V' contains fresh nodes $\{y'_1, \ldots, y'_k\}$ that each have edges to the same nodes as the left copy of the respective y_i, skipping edges introduced due to edges $h \to y_i$ with $h \in H_i$. We define $W(\Lambda, \Omega)$ for walks W in M', and $\mathscr{W}(u, v)$ for $u, v \in M'$ in the same way as they were defined in the proof of Lemma 1. Now, like in the proof of Lemma 1, for $i, j \in \{1, \ldots, k\}$, the treks in $\mathscr{T}_{i,j}$ and the walks in M' between y'_i and the right copy of w_j are in one-to-one correspondence, and given a trek π corresponding to a walk W, $\pi(\Lambda, \Omega) = W(\Lambda, \Omega)$. Moreover, two treks $\pi \in \mathscr{T}_{i,j}$ and $\pi' \in \mathscr{T}_{i',j'}$ have sided intersection if and only if their corresponding walks W and W' intersect.

Using this directed graph M', analogously to the proof of Lemma 1, we conclude that there is a trek system using only treks from the $\mathscr{T}_{i,j}$ from Y to W with no sided intersection. We shorten

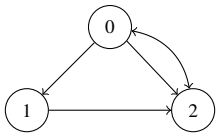

Figure 5: In this example, $0 \to 1$ is identifiable by both Theorem 4.1 in Weihs et al. [2018] and Theorem 4.1 in Weihs et al. [2017]. If we now pick $v = 2$ and $W = \{1\}$, then for $Y = \{1\}$, all conditions of Theorem 4.1 in Weihs et al. [2018] are satisfied, but condition (ii) of Theorem 4.1 in Weihs et al. [2017] is not satisfied because $0 \in \mathrm{tr}(1) \cap \mathrm{pa}(2)$ is not contained in W and $0 \to 2$ is also not known to be generically identifiable. There is also no other Y that satisfies the conditions of Theorem 4.1 in Weihs et al. [2017].

this trek system to a half-trek system from Y^* to W as follows: If the trek between y_i and w_j is already a half-trek, we do not change it. If it is of the form $y_i \leftarrow \cdots \leftarrow y_i^* \leftrightarrow \cdots \to w_j$ or $y_i \leftarrow \cdots \leftarrow y_i^* \to \cdots \to w_j$, then we shorten it to start at y_i^*. All that is left to show is that Y^* satisfies condition 2. None of the y_i^* is contained in $\{v\} \cup \mathrm{sib}(v)$ because otherwise, there would be a trek between y_i and v neither starting in $y_i \leftarrow h$ for $h \in H_i$ nor ending in $s \to v$. Let π be any trek between y_i^* and v. By extending π to start from y_i, we obtain a trek from y_i to v that does not start in $y_i \leftarrow h$ for $h \in H_i$, so from the condition, since it cannot satisfy option (b), it follows that this trek, and hence π, must satisfy option (a). □

This implies that the following proposition is equivalent to Theorem 4.1 in Weihs et al. [2018].

Proposition 1. *Let $M = (V, D, B)$ be a non-empty mixed graph and let $v \in V$. Let $W \subseteq \mathrm{pa}(v)$. Suppose there exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*

1. *The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in H_i} \sigma_{h,w_j} \lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \ldots, y_k\}$, $W = \{w_1, \ldots, w_k\}$ and the H_i contain all $h \in \mathrm{pa}(y_i)$ such that $h \to y_i$ is known to be generically identifiable, and*
2. *for every trek π from $y \in Y$ to v, we have that either*
   (a) *π ends with an edge of the form $s \to v$ where either $s \in W$ or $s \to v$ is known to be generically identifiable, or*
   (b) *π begins with an edge of the form $y \leftarrow h$ where $h \to y$ is known to be generically identifiable.*

Then for each $w \in W$, $w \to v$ is generically identifiable.

Proof. By Lemma 7, these conditions are equivalent to the ones of Weihs et al. [2018, Thm 4.1]. □

In an ArXiv version of their paper, Weihs et al. [2017, Theorem 4.1] prove a weaker version of the edgewise criterion in which condition 2, stating that each trek π from $y \in Y$ to v must either end in $s \to v$ where $s \in W$ or $s \to v$ is known to be generically identifiable, or must start in $y \leftarrow h$ where $h \to y$ is known to be generically identifiable, is replaced by the following two stricter conditions:

(ii) $\mathrm{tr}(y) \cap \mathrm{pa}(v) \subseteq W \cup \{s \mid s \to v \text{ known to be generically identifiable}\}$ for all $y \in Y$, and

(iii) for every $y \in Y$, if $h \in \mathrm{htr}(v) \cap \mathrm{pa}(y)$, then $h \to y$ is known to be generically identifiable.

It is easy to verify that for any Y, the conditions of Theorem 4.1 in Weihs et al. [2017] imply the conditions of Theorem 4.1 in Weihs et al. [2018]. As Figure 5 shows, the converse is not true: Condition (iii) follows from the conditions of Theorem 4.1 in Weihs et al. [2018], but condition (ii) does not. However, in case $W = \mathrm{pa}(v)$, condition (ii) is obviously satisfied and both theorems are equivalent to the half-trek criterion [Foygel et al., 2012, Theorem 1].

As mentioned in Remark 1, Weihs et al. [2017, Theorem 4.1] use yet another definition for the matrix $A^v_{Y,W}$ we check for invertibility. Again, for this third possibility to define the matrix, we cannot show a lemma as general as Lemma 1, stating that the invertibility of the matrix is equivalent to the existence of a half-trek system with no sided intersection. Indeed, the example in Figure 4 also serves as a counterexample for this equivalence. $Y = \{y\}$ also satisfies conditions (ii) and (iii) of Theorem 4.1 in Weihs et al. [2017], and the matrix is also given by $A^v_{y,w} = \lambda_{w,y}\omega_{w,w}$ according to this third definition. This is generically invertible, yet no half-trek system from Y to W exists.

Lemma 8. *Let $M = (V, D, B)$ be a non-empty mixed graph and let $v \in V$. Let $W \subseteq \mathrm{pa}(v)$. Then, the following are equivalent:*

- *There exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*
  - *(i) There exists a half-trek system from Y to W with no sided intersection,*
  - *(ii) $\mathrm{tr}(y) \cap \mathrm{pa}(v) \subseteq W \cup \{s \mid s \to v$ known to be generically identifiable$\}$ for all $y \in Y$, and*
  - *(iii) for every $y \in Y$, if $h \in \mathrm{htr}(v) \cap \mathrm{pa}(y)$, then $h \to y$ is known to be generically identifiable.*
- *There exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*
  - *(i) The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in H_i} \sigma_{h,w_j}\lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \ldots, y_k\}$, $W = \{w_1, \ldots, w_k\}$ and $H_i = \mathrm{htr}(v) \cap \mathrm{pa}(y_i)$,*
  - *(ii) $\mathrm{tr}(y) \cap \mathrm{pa}(v) \subseteq W \cup \{s \mid s \to v$ known to be generically identifiable$\}$ for all $y \in Y$, and*
  - *(iii) for every $y \in Y$, if $h \in \mathrm{htr}(v) \cap \mathrm{pa}(y)$, then $h \to y$ is known to be generically identifiable.*

Proof. $\Rightarrow$: Again, by Lemma 4.3 in Weihs et al. [2018], the forward implication holds even for the same Y.

$\Leftarrow$: Analogously to the proof of Lemma 7, for $i, j \in \{1, \ldots, k\}$, we define $\mathscr{T}_{i,j}$ to be the set of all treks from y_i to w_j that do not start in an edge $y_i \leftarrow h$ for $h \in H_i$. Again, it holds that $A_{i,j} = \sum_{\pi \in \mathscr{T}_{i,j}} \pi(\Lambda, \Omega)$. We use the same construction as in the proof of Lemma 7 to construct the graph $M' = (V', D')$, this time according to our new definition of H_i. Analogously to the proofs of Lemmas 1 and 7, we conclude that there is a trek system using only treks from the $\mathscr{T}_{i,j}$ from Y to W with no sided intersection. We now argue that if we shorten this trek system to a half-trek system from Y^* to W like in the proof of Lemma 7, Y^* satisfies conditions (ii) and (iii).

For condition (ii), let $y_i^* \in Y^*$ and $u \in \mathrm{tr}(y_i^*) \cap \mathrm{pa}(v)$. From $u \in \mathrm{tr}(y_i^*)$, it follows that $u \in \mathrm{tr}(y_i)$, so this follows from condition (ii) for Y. For condition (iii), the case $y_i^* = y_i$ is trivial, so consider the case $y_i^* \neq y_i$. We claim that in this case, $\mathrm{htr}(v) \cap \mathrm{pa}(y_i^*) = \emptyset$. Otherwise, $y_i^* \in \mathrm{htr}(v)$, which implies that there is a half-trek from v to y_i that ends in $h \to y_i$ for $h \notin H_i$, but $h \in \mathrm{htr}(v) \cap \mathrm{pa}(y_i) = H_i$, contradiction. □

This shows that the following proposition is equivalent to Theorem 4.1 in Weihs et al. [2017].

Proposition 2. *Let $M = (V, D, B)$ be a non-empty mixed graph and let $v \in V$. Let $W \subseteq \mathrm{pa}(v)$. Suppose there exists $Y \subseteq V \setminus (\{v\} \cup \mathrm{sib}(v))$ with $|Y| = |W|$ such that the following conditions hold:*

- *(i) The matrix A defined by $A_{i,j} = \sigma_{y_i,w_j} - \sum_{h \in H_i} \sigma_{h,w_j}\lambda_{h,y_i}$ is generically invertible, where $Y = \{y_1, \ldots, y_k\}$, $W = \{w_1, \ldots, w_k\}$ and $H_i = \mathrm{htr}(v) \cap \mathrm{pa}(y_i)$,*
- *(ii) $\mathrm{tr}(y) \cap \mathrm{pa}(v) \subseteq W \cup \{s \mid s \to v$ known to be generically identifiable$\}$ for all $y \in Y$, and*
- *(iii) for every $y \in Y$, if $h \in \mathrm{htr}(v) \cap \mathrm{pa}(y)$, then $h \to y$ is known to be generically identifiable.*

Then for each $w \in W$, $w \to v$ is generically identifiable.

Proof. By Lemma 8, these conditions are equivalent to the ones of Weihs et al. [2017, Thm 4.1]. □

D Further omitted proofs

Lemma 4. *Let z be a nonzero integer. If we choose a prime P uniformly at random from a set N of primes each larger than $B > 1$, then $\Pr(P \text{ divides } z) \leq \log_2(|z|)/(|N| \cdot \log_2(B))$.*

Proof. We claim that not more than $\log_2(|z|)/\log_2(B)$ distinct primes from N can divide z. Assume k distinct primes each larger than B divide z. Then $|z| \geq B^k$, so $k \leq \log_2(|z|)/\log_2(B)$. Hence, the probability that a random prime from N divides z is at most $\log_2(|z|)/(|N| \cdot \log_2(B))$. □

Theorem 2. *Algorithm 1 can be implemented using numbers with $\mathcal{O}(\log n)$ bits, i.e., on a Word-RAM, with an error probability tending to 0 as n goes to infinity.*

Proof. Let $\mathcal{V}$ be the set of indeterminates from Appendix B, $\Delta = \det(I - \Lambda)$ and $R_\Delta = \mathbb{Z}[\mathcal{V}][\Delta^{-1}]$. By Appendix B, $\Sigma \in R_\Delta^{n \times n}$, and $A^v_{y,w} = \tilde{N}^v_{y,w}/\Delta^2$ with $\tilde{N}^v_{y,w} \in \mathbb{Z}[\mathcal{V}]$ of degree at most $2n$ and ℓ_1 norm at most $n(n!)^2$. The algorithm chooses P uniformly from the set N of primes in $(B, 2B]$, then $a \in \mathbb{F}_P^{\mathcal{V}}$ uniformly, and computes in $\mathbb{F}_P$. If $\Delta(a) \not\equiv 0 \pmod P$ (otherwise, the failure is detected in line 2), evaluation at a followed by reduction modulo P is a ring homomorphism $\varphi\colon R_\Delta \to \mathbb{F}_P$, so the algorithm works with the matrices $\varphi(A^v_{I_v,\mathrm{pa}(v)})$. Since φ maps minors to minors,

$$\operatorname{rank}_{\mathbb{F}_P} \varphi(A^v_{I_v,\mathrm{pa}(v)}) \leq \operatorname{rank} A^v_{I_v,\mathrm{pa}(v)}. \tag{2}$$

Consider an ideal run of Algorithm 1, in which every rank is the rank of the symbolic matrix. It is a deterministic function of the input, and its control flow only depends on the rank comparisons in lines 6, 8, 13 and 15. We show that, with probability tending to 1, every rank comparison of the actual run has the same outcome as in the ideal run, i.e., all full-rank tests have the correct outcomes. For each v with $\mathrm{pa}(v) \neq \emptyset$ whose symbolic matrix reaches full rank in the ideal run, fix $Y_v \subseteq I_v$ at that moment with $\det A^v_{Y_v,\mathrm{pa}(v)} \neq 0$, and let $g_v = \Delta^{2|\,\mathrm{pa}(v)|} \det A^v_{Y_v,\mathrm{pa}(v)} \in \mathbb{Z}[\mathcal{V}] \setminus \{0\}$. Let $f = \Delta \prod_v g_v$. Then f depends only on the input.

Suppose $f(a) \not\equiv 0 \pmod P$. By induction over the steps, as long as both runs have coincided, all I_v coincide; a symbolic matrix without full rank also has no full rank after evaluation by (2), and a symbolic matrix with full rank contains the Y_v rows, whose evaluated determinant $\varphi(g_v)\varphi(\Delta)^{-2|\,\mathrm{pa}(v)|}$ is nonzero. So every rank comparison has the same outcome in both runs.

So it remains to bound the probability that $f(a) \equiv 0 \pmod P$. The polynomial f is a product of at most $n+1$ determinants of matrices of size at most $n \times n$ whose entries have degree at most $2n$ and ℓ_1 norm at most $n(n!)^2$. Hence $\deg f = \mathcal{O}(n^3)$ and $\log_2 \|f\|_1 = \mathcal{O}(n^3 \log n)$. Fix a nonzero coefficient c of f; it does not depend on the random choices. By Lemma 4, $\Pr[P \mid c] \leq \log_2 \|f\|_1/(|N| \log_2 B)$. If $P \nmid c$, then $f \bmod P$ is a nonzero polynomial over $\mathbb{F}_P$, so $\Pr[f(a) \equiv 0 \pmod P] \leq \deg f/B$ by Lemma 2. By the prime number theorem, $|N| = \Theta(B/\log B)$. Choosing $B = n^5$, the error probability is $\mathcal{O}(n^3 \log n/B + n^3/B) = \mathcal{O}(\log n/n^2) \to 0$. All numbers are smaller than $(2B)^2$, so they have $\mathcal{O}(\log n)$ bits; a random prime in $(B, 2B]$ can be found by rejection sampling with trial division in $\mathcal{O}(n^{5/2} \log n)$ expected time. □

Theorem 3. *Algorithm 1 correctly computes the half-trek criterion with high probability.*

Proof. By Theorem 2, with probability tending to 1, all full-rank tests have the correct outcomes, i.e., Algorithm 1 behaves like the ideal run from the proof of Theorem 2. It therefore suffices to show that the ideal run outputs exactly the HTC-identifiable edges.

By the discussion at the beginning of Section 4, a node y satisfies condition 2 of Theorem 1 for $W = \mathrm{pa}(v)$ if and only if $y \notin \{v\} \cup \mathrm{sib}(v)$ and ($y \notin \mathrm{htr}(v)$ or y is already identified). We call such y eligible for v. Eligibility is monotone in the set of identified nodes. By lines 6, 7, 13 and 14, only eligible y are added to I_v. Hence, whenever v is pushed, the full rank of $A^v_{I_v,\mathrm{pa}(v)}$ yields a $Y \subseteq I_v$ satisfying the conditions of Theorem 1, so by induction, every pushed node is identified by the HTC. Conversely, at termination, all nodes eligible for v have been added to I_v for every node v that was not pushed, and $A^v_{I_v,\mathrm{pa}(v)}$ does not have full rank, so Theorem 1 identifies no further node. The HTC starts with the nodes without parents, which are pushed in line 9, and since eligibility is monotone, every node it identifies from a subset of the pushed nodes is pushed as well. Hence the ideal run outputs exactly the HTC-identifiable edges.

Moreover, by (2), a node is only pushed if its symbolic matrix has full rank, so the same induction shows that Algorithm 1 never outputs an edge that is not HTC-identifiable, even if some rank is computed incorrectly. □

Lemma 5. *Algorithm 2 can be implemented to run in $\mathcal{O}(2^d(n^3 + n^2 m_2))$, where d is the maximum indegree in the graph. This is $\mathcal{O}(2^d n^4)$ for dense and $\mathcal{O}(2^d n^3)$ for sparse graphs.*

Proof. The running time of Algorithm 1 is dominated by computing the rows of the matrices A^v and integrating rows into the row echelon form. The time complexity of this for a single matrix does not change, so since we now have $\mathcal{O}(n \cdot 2^d)$ instead of n such matrices, the running time bound is simply multiplied by 2^d. □

Lemma 6. *Algorithm 2 can be implemented using numbers with $\mathcal{O}(\log n)$ bits with an error probability tending to 0 as n goes to infinity. More precisely, we only need to pass d times more identity tests successfully compared to Algorithm 1.*

Proof. We use the notation of the proof of Theorem 2; in particular, (2) holds for all matrices $\varphi(A^v_{I_{v,W},W})$ computed by Algorithm 2. We assume that htr and tr are computed by graph search, so that eligibility is decided exactly. Unlike in the proof of Theorem 2, we do not compare with the ideal run, since this would require one identity test per pair (v, W) whose matrix reaches full rank, i.e., up to $n2^d$ tests. Instead, we only compare the outputs.

For a set K of directed edges, y is eligible for (v, W) with respect to K if y satisfies condition 2 of Theorem 1 when the edges in K are taken as the edges known to be generically identifiable. Eligibility is monotone in K. Let $\Phi(K)$ be the union of K and all $w \to v$ with $w \in W$ for some allowed $W \subseteq \mathrm{pa}(v)$ and some Y of eligible nodes for (v, W) with respect to K such that $|Y| = |W|$ and $\det A^v_{Y,W} \neq 0$. Then Φ is monotone, and by Theorem 1, its least fixed point K^* is the set of edges identified by the edgewise HTC.

Starting from $K_0 = \emptyset$, we choose triples (v_r, W_r, Y_r) with Y_r eligible for (v_r, W_r) with respect to K_{r-1}, $|Y_r| = |W_r|$ and $\det A^{v_r}_{Y_r,W_r} \neq 0$, such that $K_r = K_{r-1} \cup \{w \to v_r : w \in W_r\} \supsetneq K_{r-1}$, until $K_R = K^*$. This is possible since $\Phi(K_{r-1}) \subseteq K^*$, and $\Phi(K_{r-1}) \neq K_{r-1}$ as long as $K_{r-1} \neq K^*$. Every round adds an edge, so $R \leq |D| \leq dn$. Let $g_r = \Delta^{2|W_r|} \det A^{v_r}_{Y_r,W_r} \in \mathbb{Z}[\mathcal{V}] \setminus \{0\}$ and $f = \Delta \prod_{r=1}^{R} g_r$. Then f depends only on the input.

The algorithm is sound, regardless of the random choices: By (2), a pair (v, W) is only pushed if its symbolic matrix has full rank. Since only nodes eligible with respect to the edges marked so far are added to $I_{v,W}$, induction over the marked edges shows that every marked edge lies in K^*.

For completeness, suppose $f(a) \not\equiv 0 \pmod P$, and let K_{out} be the set of marked edges at termination. If $K_{\mathrm{out}} \neq K^*$, let r be minimal with $\{w \to v_r : w \in W_r\} \not\subseteq K_{\mathrm{out}}$. Then $K_{r-1} \subseteq K_{\mathrm{out}}$, so every node of Y_r is eligible for (v_r, W_r) with respect to K_{out}. If the matrix of (v_r, W_r) did not have full rank at termination, then $Y_r \subseteq I_{v_r,W_r}$, since Algorithm 2 adds every node to I_{v_r,W_r} as soon as it becomes eligible; but the evaluated determinant $\varphi(g_r)\varphi(\Delta)^{-2|W_r|}$ of the Y_r rows is nonzero, so the matrix has full rank after all. Hence (v_r, W_r) was pushed and popped, and all $w \to v_r$ with $w \in W_r$ were marked, a contradiction. Thus $K_{\mathrm{out}} = K^*$.

We finally estimate the error probability. The polynomial f is a product of Δ and at most $|D|$ determinants of matrices of size at most $d \times d$ whose entries have degree at most $2n$ and ℓ_1 norm at most $n(n!)^2$. With $\sum_r |W_r| \leq \sum_v |\mathrm{pa}(v)|^2 \leq d|D|$, this gives

$$\deg f \leq n + 2nd|D|, \qquad \log_2 \|f\|_1 \leq n \log_2 n\,(1 + 4d|D|). \tag{3}$$

For Algorithm 1, the same bounds hold without the factor d, which is the precise sense in which we only need to pass d times more identity tests. As in the proof of Theorem 2,

$$\Pr[K_{\mathrm{out}} \neq K^*] \leq \frac{\log_2 \|f\|_1}{|N| \log_2 B} + \frac{\deg f}{B}. \tag{4}$$

With $d < n$ and $|D| < n^2$, we have $\deg f = \mathcal{O}(n^4)$ and $\log_2 \|f\|_1 = \mathcal{O}(n^4 \log n)$, so choosing $B = n^6$, the error probability is $\mathcal{O}(\log n / n^2) \to 0$, and all numbers have $\mathcal{O}(\log n)$ bits. □

E Further benchmarks

Figures 6 to 14 show the execution times of SEMID and our program on the test cases for $k \in \{1, \ldots, 9\}$. For a description of how these test cases were generated, see Section 5.

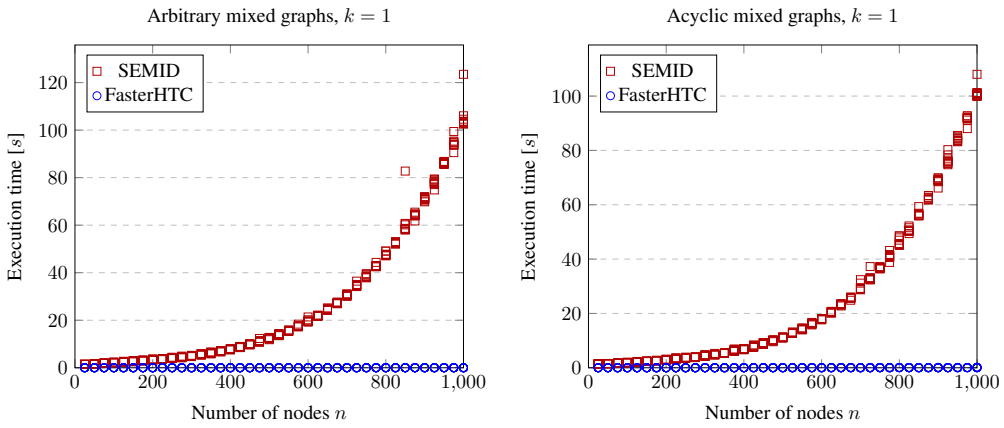

Figure 6: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 1$.

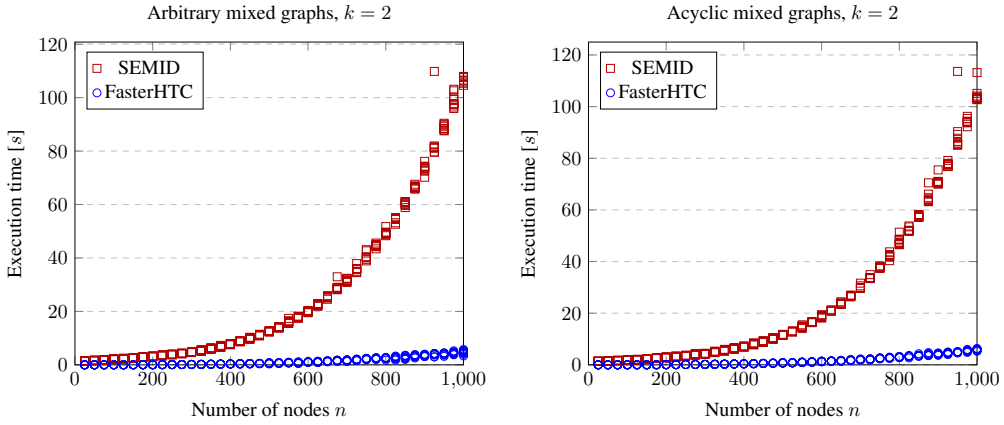

Figure 7: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 2$.

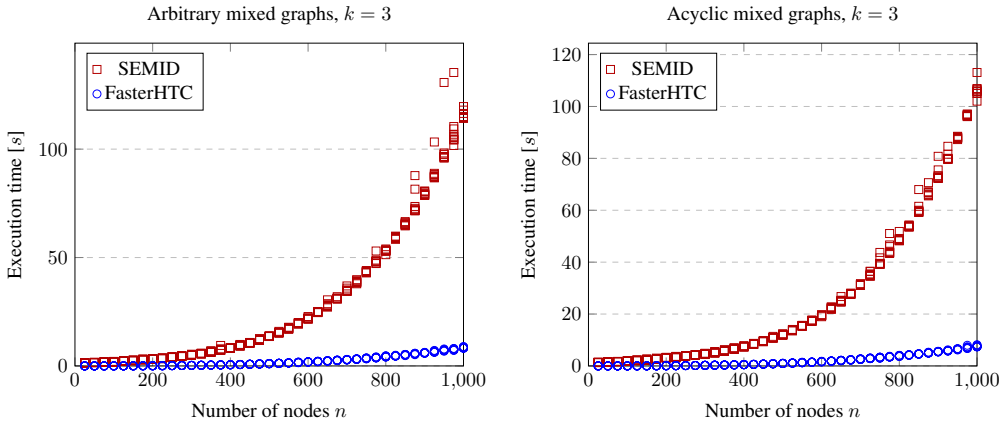

Figure 8: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 3$.

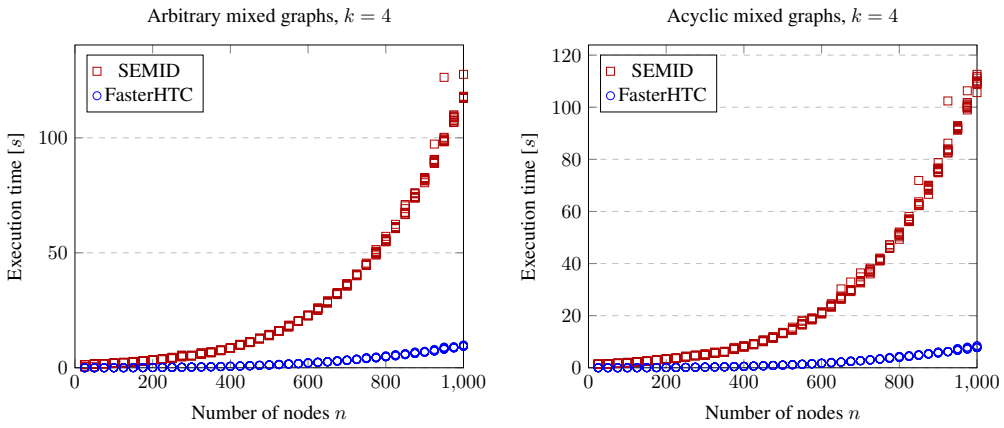

Figure 9: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 4$.

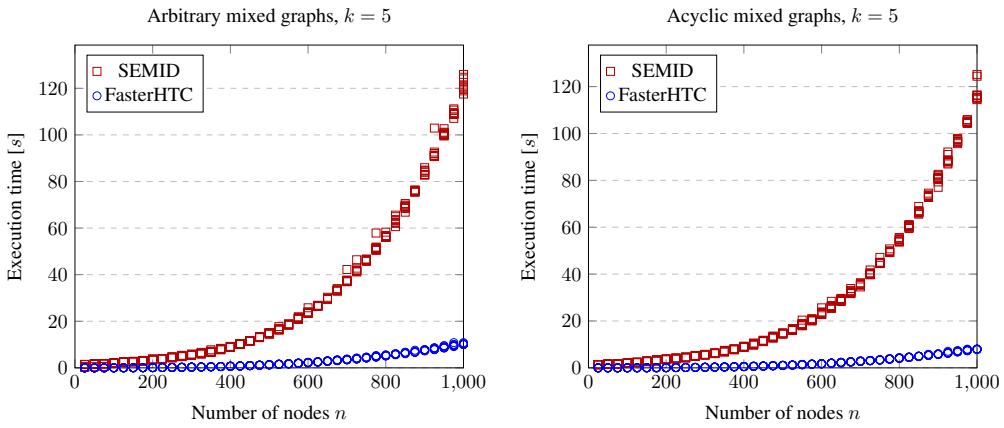

Figure 10: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 5$.

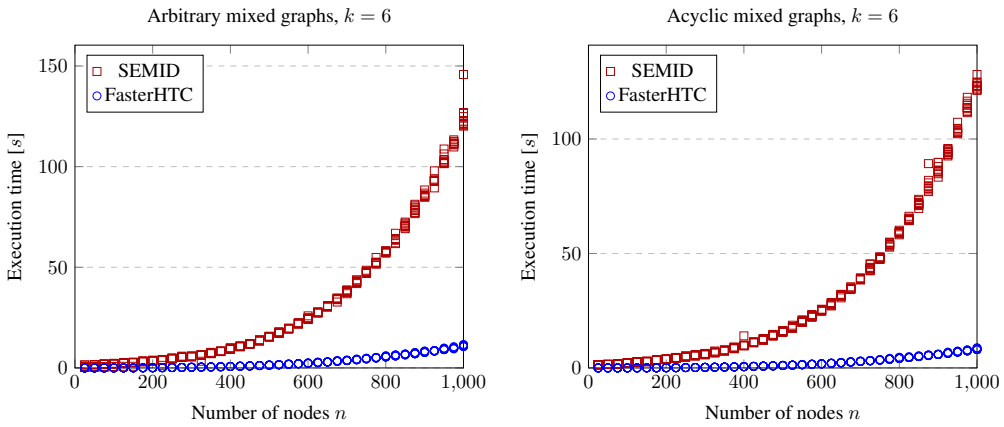

Figure 11: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 6$.

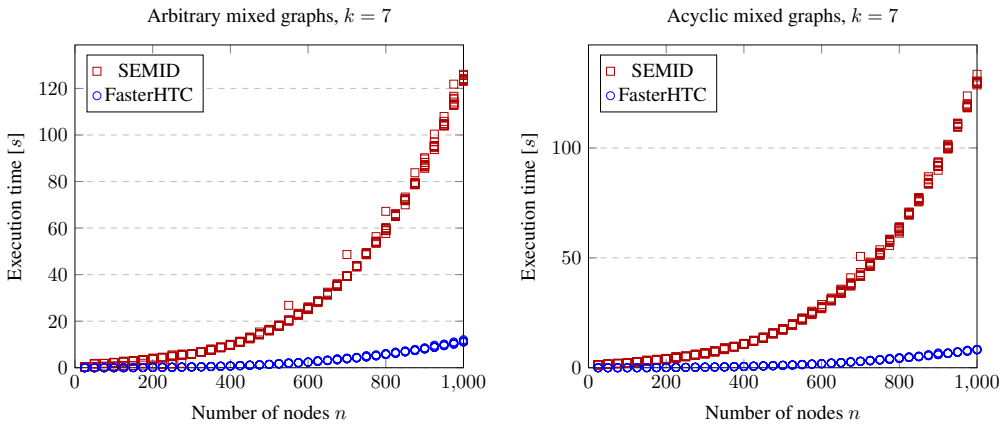

Figure 12: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 7$.

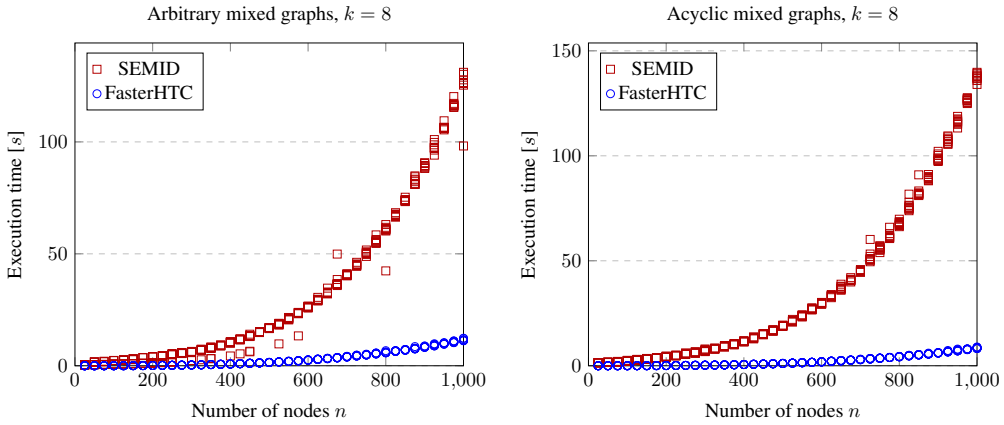

Figure 13: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 8$. The left plot shows the results for arbitrary mixed graphs. Here, the same phenomenon that can be observed for $k = 10$ occurs: A second curve of test cases for which the half-trek criterion fails, and SEMID thus takes significantly less time, starts to form, although here, for the majority of test cases, the half-trek criterion succeeds.

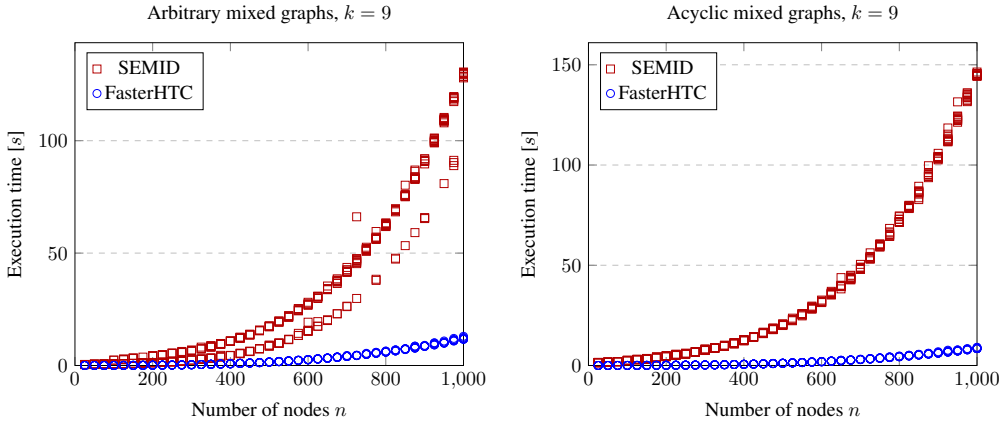

Figure 14: These plots show the execution times of SEMID and our program FasterHTC on the randomly generated mixed graphs and acyclic mixed graphs for $k = 9$. The left plot shows the results for arbitrary mixed graphs. Just like for $k = 8$ and $k = 10$, a second curve of test cases on which the half-trek criterion fails, and SEMID thus takes significantly less time, can be observed.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The equivalent edgewise criteria are proven in Section 3 and Appendix C. The correctness of the proposed algorithm and its running time are proven in Section 4, Appendix A.2 and Appendix D. Section 5 and Appendix E contain benchmarks that show that our algorithm outperforms SEMID in practice.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We state that the algorithm implements the half-trek criterion, which is not complete. We mention that the algorithm is randomized with one-sided error and we bound the error probability with a mathematically rigorous proof in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All theoretical claims are stated with precise assumptions and mathematically sound proofs either accompany the lemmas, propositions, and theorems, or they are given in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: In Section 5, we describe how exactly the test cases were generated. Our code, all test cases, testing scripts and a Dockerfile providing an environment in which all programs can be run are provided in the supplementary material, together with a detailed explanation on how to reproduce the results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our code and all test cases are provided in the supplementary material. For more details, refer to our justification for question 4.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For each configuration (parameter k and arbitrary/acyclic), and each number of nodes n, we generated 10 test cases and executed both programs on all of them. As can be seen in the plots in Section 5 and Appendix E, the execution times of the tested programs are very consistent over multiple runs. We also compared the outputs, and on all test cases, the outputs of SEMID and our program agreed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our code is provided in the supplementary material. We do not see any other ethical issues.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: There are no direct societal impacts of our work.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: There are no such data or models in our work.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All code packages used are cited in the work.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our documented code is provided in the supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our work does not include crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our work does not include crowdsourcing experiments or research with human subjects.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Our work does not involve LLMs as an important, original or non-standard component.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.