Linear-Time Algorithms for Front-Door Adjustment in Causal Graphs

Marcel Wienöbst, Benito van der Zander, Maciej Liśkiewicz

Institute of Theoretical Computer Science, University of Lübeck, Germany
{m.wienoebst,b.vanderzander,maciej.liskiewicz}@uni-luebeck.de

Abstract

Causal effect estimation from observational data is a fundamental task in empirical sciences. It becomes particularly challenging when unobserved confounders are involved in a system. This paper focuses on front-door adjustment – a classic technique which, using observed mediators allows to identify causal effects even in the presence of unobserved confounding. While the statistical properties of the front-door estimation are quite well understood, its algorithmic aspects remained unexplored for a long time. In 2022, Jeong, Tian, and Bareinboim presented the first polynomial-time algorithm for finding sets satisfying the front-door criterion in a given directed acyclic graph (DAG), with an $O(n^3(n+m))$ run time, where n denotes the number of variables and m the number of edges of the causal graph. In our work, we give the first linear-time, i.e., $O(n+m)$, algorithm for this task, which thus reaches the asymptotically optimal time complexity. This result implies an $O(n(n+m))$ delay enumeration algorithm of all front-door adjustment sets, again improving previous work by a factor of n^3. Moreover, we provide the first linear-time algorithm for finding a *minimal* front-door adjustment set. We offer implementations of our algorithms in multiple programming languages to facilitate practical usage and empirically validate their feasibility, even for large graphs.

1 Introduction

Discovering and understanding causal relationships and distinguishing them from purely statistical associations is a fundamental objective of empirical sciences. For example, recognizing the causes of diseases and other health problems is a central task in medical research enabling novel disease prevention and treatment strategies. One possible approach for establishing causal relationships and analyzing causal effects is through Randomized Controlled Trials (Fisher 1936), which are considered the *gold standard of experimentation*. In practice, however, experimentation is not always possible due to costs, technical feasibility, or ethical constraints – e.g., participants of medical studies should not be assigned to smoke over extended periods of time to ascertain its harmfulness.

The goal of *causal inference* is to determine cause-effect relationships by combining observed and interventional data with existing knowledge. In this paper, we focus on the problem of deciding when causal effects can be identified from a graphical model *and* observed data and, if possible, how to estimate the strength of the effect. The model is typically represented as a directed acyclic graph (DAG), whose edges encode direct causal influences between the random variables of interest. To analyze the causal effects in such models, Pearl (1995, 2009) introduced the do-operator which performs a hypothetical intervention forcing exposure (treatment) variables $\mathbf{X}$ to take some values $\mathbf{x}$. This allows to regard the (total) *causal effect* of $\mathbf{X}$ on outcome variables $\mathbf{Y}$, denoted as $P(\mathbf{y}|do(\mathbf{x}))$, as the probability distribution of variables $\mathbf{Y}$ after the intervention.[1] The fundamental task in causal inference is to decide whether $P(\mathbf{y}|do(\mathbf{x}))$ can be expressed using only *standard* (i.e., do-operator free) probabilities and it becomes challenging when unobserved confounders (variables affecting both the treatment and outcome) are involved in a system. A variable is considered unobserved if it cannot be measured by the researcher. Fig. 1 shows DAGs of some models with unobserved confounders.

It is well known that the IDC algorithm by Shpitser and Pearl (2006a), based on the prominent do-calculus (Pearl 1995), allows solving the identifiability problem in a sound and complete way (Huang and Valtorta 2006; Shpitser and Pearl 2006b). As such, researchers could potentially apply IDC to decide identifiability. However, two drawbacks affect the widespread use of the algorithm: Firstly it runs in polynomial time of high degree which precludes computations for graphs involving a reasonable amount of variables. Secondly, the IDC algorithm computes complex expressions, even in case of small DAGs for which a simple formula exists (for details, see, e.g., (van der Zander, Liśkiewicz, and Textor 2019)). Hence, in practice, total causal effects are estimated using other methods.

One of the most popular approaches is to utilize *covariate adjustment* of the form $P(\mathbf{y}|do(\mathbf{x})) = \sum_{\mathbf{z}} P(\mathbf{y}|\mathbf{x},\mathbf{z})P(\mathbf{z})$, which is valid if $\mathbf{Z}$ satisfies the famous back-door (BD) criterion (Pearl 1995). Apart from convenient statistical properties, the BD based methods rely on efficient, sound, and complete algorithms for covariate adjustments in DAGs. In particular, utilizing the generalized BD criterion by Shpitser, VanderWeele, and Robins (2010), the algorithmic framework

[1] Following convention, for a random variable X, we use $P(x)$ as a shorthand for $P(X = x)$. By bold capital letters $\mathbf{X}, \mathbf{Y}$, etc., we denote sets of variables, and the corresponding sets of values are denoted by bold lowercase letters $\mathbf{x}, \mathbf{y}$, etc.

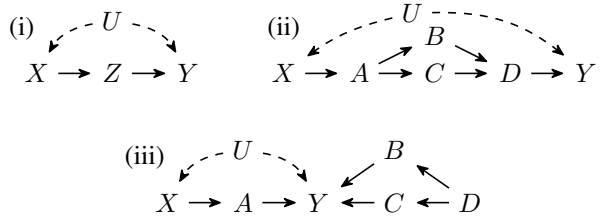

Figure 1: Causal graphs, where X is the treatment, Y the outcome, and U represents an unobserved confounder. Graph (i) is a canonical example, with the (unique) set $\{Z\}$ satisfying the front-door (FD) criterion relative to (X, Y). For graph (ii), there exist 13 FD sets; Both the algorithm of Jeong, Tian, and Bareinboim as well as our basic Algorithm 2, output $\mathbf{Z} = \{A, B, C, D\}$ of maximum size. In contrast, Algorithm 3 computes minimal FD set $\{D\}$ of size 1. Graph (iii) illustrates the non-monotonicity of the FD criterion: while both $\{A, B, C\}$ and $\{A\}$ are FD sets, neither $\{A, B\}$ nor $\{A, C\}$ nor $\{B, C\}$ satisfy the FD criterion.

provided by van der Zander, Liśkiewicz, and Textor (2014) allows to find an adjustment set in linear-time $O(n + m)$ and to enumerate all covariate adjustment sets with *delay* $O(n(n + m))$, i.e., at most $O(n(n + m))$ time passes between successive outputs.[2]

However, BD based approaches are unable to identify the causal effect in many cases involving unobserved confounders, which are commonplace in practice. For example, none of the instances in Fig. 1 can be identified via covariate adjustment, although $P(y|do(x))$ can be expressed by the formula (1) below. This illustrates the use of another classic technique, known as front-door (FD) adjustment (Pearl 1995), which is the main focus of this paper. The advantage of this approach, as seen in the example, is that it leverages observed mediators to identify causal effects even in the presence of unobserved confounding. In the general case, if a set of variables $\mathbf{Z}$ satisfies the FD criterion[3] relative to $(\mathbf{X}, \mathbf{Y})$ in a DAG G, the variables $\mathbf{Z}$ are observed and $P(\mathbf{x}, \mathbf{z}) > 0$, then the effect of $\mathbf{X}$ on $\mathbf{Y}$ is identifiable and is given by the formula

$$P(\mathbf{y}|do(\mathbf{x})) = \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{x}) \sum_{\mathbf{x}'} P(\mathbf{y}|\mathbf{x}', \mathbf{z})\ P(\mathbf{x}'), \qquad (1)$$

respectively $P(\mathbf{y}|do(\mathbf{x})) = P(\mathbf{y})$ in case $\mathbf{Z} = \emptyset$.

FD adjustment is an effective alternative to standard covariate adjustment (Glynn and Kashin 2018) and is met with increasing applications in real-world datasets (Bellemare, Bloem, and Wexler 2019; Gupta, Lipton, and Childers 2021; Chinco and Mayer 2016; Cohen and Malloy 2014). Recent works (Kuroki 2000; Glynn and Kashin 2018; Gupta, Lipton, and Childers 2021) have improved the understanding of the statistical properties of FD estimation and provided robust generalizations of this approach (Hünermund and Bareinboim 2019; Fulcher et al. 2020).

However, despite these advantages, the algorithmic and complexity-theoretical aspects of FD adjustment remained unexplored for a long time. Very recently, Jeong, Tian, and Bareinboim (2022) have provided the first polynomial-time algorithm for finding an FD adjustment set with an $O(n^3(n + m))$ run time. This amounts to $O(n^5)$ for dense graphs, which does not scale well even for a moderate number of variables. The authors also gave an algorithm for enumerating all FD sets, which has delay $O(n^4(n + m))$. Yet, it remained open, whether these tasks can be solved more efficiently.

Additionally, the $O(n^3(n + m))$ algorithm by Jeong, Tian, and Bareinboim (2022) always returns the maximum size FD set $\mathbf{Z}$, which is often unpractical as it hinders the estimation of FD adjustment formula (1), which sums over all possible values $\mathbf{z}$. Example (ii) in Fig. 1 illustrates this issue: while $\{A, B, C, D\}$ is a valid FD adjustment, it is not *minimal* since its proper subset, e.g., $\{A\}$, satisfies the FD criterion, as well. In this work, we address this issue of finding a minimal FD set. A feature, which may make the problem difficult to solve, is the non-monotonicity of the FD criterion, illustrated in Fig. 1.(iii).

The main contributions of this work are threefold:

- We present the first linear-time, that is $O(n + m)$, algorithm, for finding an FD adjustment set. This run time is asymptotically optimal, as the size of the input is $\Omega(n + m)$.
- For enumeration of FD adjustment sets, we provide an $O(n(n + m))$-delay algorithm.
- We give the first linear-time algorithm for finding a *minimal* FD set.

Thus, our results show that, in terms of computational complexity, the problems of finding and enumerating FD sets are not harder than for the well-studied covariate adjustment and indeed, our algorithms match the run time of the methods used in this setting. In addition, our work offers implementations of the new algorithms in multiple programming languages to facilitate practical usage and we empirically validate their feasibility, even for large graphs.

2 Preliminaries

A directed graph $G = (\mathbf{V}, \mathbf{E})$ consists of a set of vertices (or variables) $\mathbf{V}$ and a set of directed edges $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$. In case of a directed edge $A \to B$, vertex A is called a *parent* of B and B is a *child* of A. In case there is a *causal* path $A \to \cdots \to B$, then A is called an *ancestor* of B and B is a *descendant* of A. Vertices are descendants and ancestors of themselves, but not parents/children. The sets of parents, children, ancestors, and descendants of a vertex V are denoted by $\boldsymbol{Pa}(V)$, $\boldsymbol{Ch}(V)$, $\boldsymbol{An}(V)$, and $\boldsymbol{De}(V)$, and they generalize to sets $\mathbf{V}$ in the natural way. We consider only acyclic graphs (DAGs), i.e., if $B \in \boldsymbol{De}(A)$, then there is no edge $B \to A$. We denote by $G_{\overline{\mathbf{S}}}$ the graph obtained by removing from G all edges $\to S$ for every $S \in \mathbf{S}$, and by $G_{\underline{\mathbf{S}}}$, the graph obtained by removing $\leftarrow S$ for every $S \in \mathbf{S}$.

The statement $(\mathbf{A} \perp\!\!\!\perp \mathbf{B} \mid \mathbf{C})_G$ in a DAG G holds for pairwise disjoint sets of vertices $\mathbf{A}, \mathbf{B}, \mathbf{C} \subset \mathbf{V}$ if $\mathbf{A}$ and $\mathbf{B}$

[2] By n, we denote the number of variables/vertices, by m the number of edges in the causal graph.

[3] For a definition of the front-door criterion, see Sec. 2. We term sets satisfying this criterion *FD sets*.

are *d-separated* in G given $\mathbf{C}$ – that is, if there is no *open* path from some vertex $A \in \mathbf{A}$ to a vertex $B \in \mathbf{B}$ given $\mathbf{C}$. A *path* is a sequence of adjacent, pairwise different vertices and it is *open* if, for any collider Y on the path (that is $X \to Y \leftarrow Z$), we have $\boldsymbol{De}(Y) \cap \mathbf{C} \neq \emptyset$, and, for any non-collider Y, we have $Y \notin \mathbf{C}$. In this work, we sometimes consider *ways* instead of paths. A way may contain a vertex up to two times and is open given $\mathbf{C}$ if, for any collider Y, we have $Y \in \mathbf{C}$ and, for any non-collider Y, we have $Y \notin \mathbf{C}$. In case there is an open way between two sets of vertices, there is also an open path and vice versa (see Appendix A for details). Hence, ways can be used to determine d-separation and they make up the traversal sequence of the well-known Bayes-Ball algorithm (Shachter 1998) for testing d-separation in linear time. A path (or way) from A to B is called a *back-door* (BD) path (or way) if it starts with the edge $A \leftarrow$. For a set of vertices $\mathbf{A}$, we often speak of *proper* back-door (BD) paths (or ways), which are such that the path $A \leftarrow \ldots B$ for $A \in \mathbf{A}$ and $B \in \mathbf{B}$ does not contain any other vertex in $\mathbf{A}$.

Let $G = (\mathbf{V}, \mathbf{E})$ be a DAG and let $\mathbf{I}, \mathbf{R}$, with $\mathbf{I} \subseteq \mathbf{R}$, be subsets of vertices. Given pairwise disjoint $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subset \mathbf{V}$, set $\mathbf{Z}$, with the constraint $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$, satisfies the *front-door criterion* relative to $(\mathbf{X}, \mathbf{Y})$ in G if (Pearl 1995):

FD(1). The set $\mathbf{Z}$ intercepts all directed paths from $\mathbf{X}$ to $\mathbf{Y}$.

FD(2). There is no unblocked proper BD path from $\mathbf{X}$ to $\mathbf{Z}$, i.e., $(\mathbf{Z} \perp\!\!\!\perp \mathbf{X})_{G_{\underline{\mathbf{X}}}}$.

FD(3). All proper BD paths from $\mathbf{Z}$ to $\mathbf{Y}$ are blocked by $\mathbf{X}$, i.e., $(\mathbf{Z} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\mathbf{Z}}}}$.

The set $\mathbf{I}$ consists of variables that *must* be included in the FD set, the set of variables $\mathbf{R}$ consists of the ones that *can* be used. We consider graphs consisting *only* of directed edges. Bidirected edges $A \leftrightarrow B$ are frequently used to represent confounding and can be replaced by $A \leftarrow U \to B$, where U is a new variable, which is not in $\mathbf{R}$, in order to make use of the algorithms presented here.

The following algorithmic idea for finding a set $\mathbf{Z}$ satisfying the FD criterion relative to $(\mathbf{X}, \mathbf{Y})$ (if such a set exists) was recently given by Jeong, Tian, and Bareinboim (2022):

(i) Let $\mathbf{Z}_{(\mathrm{i})} \subseteq \mathbf{R}$ be the set of all variables $Z \in \mathbf{R}$, which satisfy $(Z \perp\!\!\!\perp \mathbf{X})_{G_{\underline{\mathbf{X}}}}$.

(ii) Let $\mathbf{Z}_{(\mathrm{ii})} \subseteq \mathbf{Z}_{(\mathrm{i})}$ be the set of all $Z \in \mathbf{Z}_{(\mathrm{i})}$, for which $\exists \mathbf{S} \subseteq \mathbf{Z}_{(\mathrm{i})}$ s.t. $(\{Z\} \cup \mathbf{S} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\{Z\} \cup \mathbf{S}}}}$.

(iii) If $\mathbf{I} \subseteq \mathbf{Z}_{(\mathrm{ii})}$ and $\mathbf{Z}_{(\mathrm{ii})}$ intercepts all causal paths from $\mathbf{X}$ to $\mathbf{Y}$, then output $\mathbf{Z}_{(\mathrm{ii})}$, else output $\perp$.

This algorithm is correct because all vertices *not* in $\mathbf{Z}_{(\mathrm{ii})}$ cannot be in any set satisfying the FD criterion, as they are not in $\mathbf{R}$ or would violate FD(2) and/or FD(3). It follows from this maximality of $\mathbf{Z}_{(\mathrm{ii})}$ that if $\mathbf{I}$ is not a subset of $\mathbf{Z}_{(\mathrm{ii})}$ or if $\mathbf{Z}_{(\mathrm{ii})}$ does not satisfy FD(1), then no set does (if some set $\mathbf{Z}$ satisfies FD(1), then any superset does as well). Jeong, Tian, and Bareinboim (2022) showed that step (i) can be performed in time $O(n(n+m))$, step (ii) in time $O(n^3(n+m))$ and step (iii) in time $O(n+m)$.

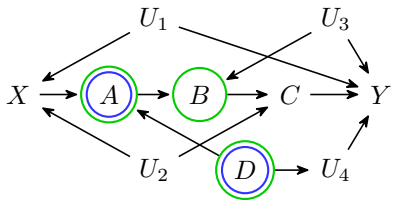

Figure 2: Running example for the algorithms for finding FD sets in $O(n+m)$ given in this section. Nodes in $\mathbf{Z}_{(\mathrm{i})}$ are marked green and nodes in $\mathbf{Z}_{(\mathrm{ii})}$ are marked blue.

3 A Linear-Time Algorithm for Finding Front-Door Adjustment Sets

In this section, we show how step (i) and (ii) can be performed in time $O(n+m)$, which leads to the first linear-time algorithm for finding front-door adjustment sets. For (i), the Bayes-Ball algorithm (Shachter 1998) can be used. It computes all variables d-connected to a given set of variables in time $O(n+m)$. As the Bayes-Ball algorithm forms the basis for many parts of this work, we provide a brief introduction in Appendix A.

Lemma 1. *It is possible to find $\mathbf{Z}_{(\mathrm{i})} \subseteq \mathbf{R}$, i.e., all vertices Z in $\mathbf{R}$ with $(Z \perp\!\!\!\perp \mathbf{X})_{G_{\underline{\mathbf{X}}}}$, in time $O(n+m)$.*

Proof. Start Bayes-Ball (Shachter 1998) (Algorithm 5 in Appendix A) at $\mathbf{X}$ in the DAG $G_{\underline{\mathbf{X}}}$. Precisely the vertices $\mathbf{N}$ not reached by the algorithm satisfy $(Z \perp\!\!\!\perp \mathbf{X})_{G_{\underline{\mathbf{X}}}}$. Hence, $\mathbf{Z}_{(\mathrm{i})} = \mathbf{N} \cap \mathbf{R}$. □

We exemplify the algorithms in this section with the running example in Fig. 2. Here, the goal is to find an FD set relative to X and Y. The variables U_1 to U_4 are unobserved, hence $\mathbf{R} = \{A, B, C, D\}$. Moreover, we have $\mathbf{I} = \emptyset$. In the graph, vertex C is reachable via the BD path $X \leftarrow U_2 \to C$ from X, whereas the remaining vertices in $\mathbf{R}$, A, B and D, are not reachable by such a path. Hence, $\mathbf{Z}_{(\mathrm{i})} = \{A, B, D\}$.

It remains to show how to execute step (ii), i.e., to compute $\mathbf{Z}_{(\mathrm{ii})}$, in time $O(n+m)$. Thus, it is our task, given a set $\mathbf{Z}_{(\mathrm{i})} \subseteq \mathbf{R}$ disjoint with $\mathbf{X}$ and $\mathbf{Y}$, which contains all vertices satisfying (i), to decide for every $Z \in \mathbf{Z}_{(\mathrm{i})}$ whether there exists a set $\mathbf{S} \subseteq \mathbf{Z}_{(\mathrm{i})}$ with $(\{Z\} \cup \mathbf{S} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\{Z\} \cup \mathbf{S}}}}$.

First, we define the notion of a *forbidden* vertex v:

Definition 1. *A vertex V is* forbidden *if it is not in $\mathbf{Z}_{(\mathrm{ii})}$. Hence, by definition this is the case if (a) $V \notin \mathbf{Z}_{(\mathrm{i})}$ or (b) there exists no $\mathbf{S} \subseteq \mathbf{Z}_{(\mathrm{i})}$ for V such that $(\{V\} \cup \mathbf{S} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\{V\} \cup \mathbf{S}}}}$.*

Our goal will be to find all *forbidden* vertices. The remaining vertices then make up the sought after set $\mathbf{Z}_{(\mathrm{ii})}$. We utilize the following key lemma.

Lemma 2. *Let G be a DAG and $\mathbf{X}$, $\mathbf{Y}$ disjoint sets of vertices. Vertex V is* forbidden *if, and only if,*

(A) $V \notin \mathbf{Z}_{(\mathrm{i})}$,
(B) $V \leftarrow \mathbf{Y}$, *or*
(C) *there exists an open BD way π (consisting of at least three variables) from V to $\mathbf{Y}$ given $\mathbf{X}$, and all its nonterminal vertices are forbidden.*

Proof. We show two directions. First, if $V \in \mathbf{Z}_{(\mathrm{i})}$, there is no edge $V \leftarrow \mathbf{Y}$ and there exists no open BD way with only forbidden vertices from V to $\mathbf{Y}$ given $\mathbf{X}$, then there exists a set $\mathbf{S} \subseteq \mathbf{Z}_{(\mathrm{i})}$, for which $(\{V\} \cup \mathbf{S} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\{V\}\cup\mathbf{S}}}}$ holds. It can be constructed by choosing, for every open way from V to $\mathbf{Y}$, a non-forbidden vertex W and its set $\mathbf{S}_W$ (which fulfills $(\{W\}\cup\mathbf{S}_W \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\{W\}\cup\mathbf{S}_W}}}$) and taking the union $\bigcup_W(\{W\} \cup \mathbf{S}_W) = \mathbf{S}$. In particular, by taking W into $\mathbf{S}$, we close the open way it is on as W is a non-collider (it is not in $\mathbf{X}$ as it is non-forbidden by definition) and hence the way is cut, due to the removal of outgoing edges from W. By adding all vertices in $\mathbf{S}_W$, it holds that W and the vertices in $\mathbf{S}_W$ have no open BD way to $\mathbf{Y}$ given $\mathbf{X}$. For this, note that if $(\{W\}\cup\mathbf{S}_W \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\{W\}\cup\mathbf{S}_W}}}$ is true, then we also have for every set $\mathbf{S}' \supseteq \{W\} \cup \mathbf{S}_W$ that $(\{W\} \cup \mathbf{S}_W \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\mathbf{S}'}}}$. Hence, taking the union $\bigcup_W(\{W\} \cup \mathbf{S}_W)$ will not open any previously closed ways.

Second, if V is not in $\mathbf{Z}_{(\mathrm{i})}$, then V is forbidden by Definition 1 (part *(a)*) and if there exists an open BD way π with only forbidden vertices, a set $\mathbf{S}$ satisfying *(b)* can never be found (the way could only be closed by adding one of its vertices to $\mathbf{S}$, but all of them are forbidden). □

Algorithm 1: Finding the set $\mathbf{Z}_{(\mathrm{ii})}$ in time $O(n+m)$.

input : DAG $G = (\mathbf{V}, \mathbf{E})$ and sets $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}_{(\mathrm{i})} \subset \mathbf{V}$.
output : Set $\mathbf{Z}_{(\mathrm{ii})}$.

```
1  Initialize visited[V, inc], visited[V, out] and
     continuelater[V] with false for all V ∈ V.
2  forbidden[V] := true if V ∉ Z_(i) else false.
3  function visit(G, V, edgetype)
4      visited[V, edgetype] := true
5      forbidden[V] := true
6      if V ∉ X then
7          foreach W ∈ Ch(V) do
8              if not visited[W, inc] then
                   visit(G, W, inc)
9          end
10         if edgetype = out then
11             foreach W ∈ Pa(V) do
12                 if not visited[W, out] then
13                     if forbidden[W] then
14                         visit(G, W, out)
15                     else
16                         continuelater[W] := true
17                     end
18                 end
19             end
20     end
21     if continuelater[V] and not visited[V, out]
         then visit(G, V, out)
22 end
23 foreach Y ∈ Y do
24     if not visited[Y, out] then visit(G, Y, out)
25 end
26 return V \ {V | forbidden[V] = true}
```

In Fig. 2, X, Y, all U_i and C are forbidden as they are not in $\mathbf{Z}_{(\mathrm{i})}$ (condition (A) of Lemma 2). Moreover, vertex B is forbidden, as there is a BD way from it via forbidden vertex U_3 to Y (condition (C) of Lemma 2). In contrast, A and D are not forbidden and it follows that $\mathbf{Z}_{(\mathrm{ii})} = \{A, D\}$.

Lemma 2 suggests an algorithmic approach for finding all forbidden vertices in linear time (a formal description is given in Algorithm 1). First, we mark all vertices $V \notin \mathbf{Z}_{(\mathrm{i})}$ as forbidden (line 2). We then start a graph search, similar to Bayes-Ball, at $\mathbf{Y}$ visiting only forbidden vertices. Each vertex is visited at most twice, once through an incoming and once through an outgoing edge. For this, when handling a vertex V, we iterate over those of its neighbors, which could extend the path over forbidden vertices with which V was reached. As in Bayes-Ball, this depends on the direction of the edges (collider/non-collider) and membership in $\mathbf{X}$. If a neighbor $W \notin \mathbf{X}$ is a child of V, we visit it and it is consequently marked forbidden (line 8), as we either have (B) that $W \leftarrow \mathbf{Y}$ (if $V \in \mathbf{Y}$) or (C) an open *BD way* from W over V to $\mathbf{Y}$ via forbidden vertices. If $W \notin \mathbf{X}$ is a parent of V and already marked forbidden (line 13), we visit it (if it has not already been visited through an outgoing edge $W \rightarrow$). If it is not marked forbidden (line 16), we record in $\mathrm{continuelater}[W]$ that it is connected to $\mathbf{Y}$ via a *non-BD way* over forbidden vertices. However, we do not visit it directly, as our search only visits forbidden vertices. If it later becomes forbidden, we continue the search at that point (line 21). Note that we do not need to consider the case $W \in \mathbf{X}$, as is shown in the proof [4] of Theorem 1.

Theorem 1. *Given a DAG and sets $\mathbf{X}, \mathbf{Y}, \mathbf{Z}_{(\mathrm{i})}$, Algorithm 1 computes the set $\mathbf{Z}_{(\mathrm{ii})}$ in time $O(n + m)$.*

For the graph in Fig. 2, first the vertices not in $\mathbf{Z}_{(\mathrm{i})}$, i.e., X, Y, U_1 to U_4 and C, are marked forbidden. The graph search starts at Y and visits U_3 and U_4 as they are parents of Y *and* already marked forbidden. Vertex B is visited as child of U_3 and thus marked forbidden as well. In contrast D, as parent of U_4 is not visited and hence not marked forbidden (it is merely marked in $\mathrm{continuelater}$). This corresponds to the fact that there is no set $\mathbf{S}$ for which $(\{B\} \cup \mathbf{S} \perp\!\!\!\perp \{Y\} \mid \{X\})_{G_{\underline{\{B\}\cup\mathbf{S}}}}$ holds, whereas $(\{D\} \perp\!\!\!\perp \{Y\} \mid \{X\})_{G_{\underline{\{D\}}}}$ is true (i.e., $\mathbf{S} = \emptyset$ contradicts condition *(b)* in Definition 1). The search then halts and A and D remain as non-forbidden vertices, which hence make up $\mathbf{Z}_{(\mathrm{ii})}$.

As the remaining step of testing whether $\mathbf{Z}_{(\mathrm{ii})}$ fulfills condition FD(1) can be performed in linear-time as well (Jeong, Tian, and Bareinboim 2022), overall linear-time follows. The full algorithm for obtaining an FD set $\mathbf{Z}$ is given in Algorithm 2.

Theorem 2. *Given a DAG and sets $\mathbf{X}$ and $\mathbf{Y}$, Algorithm 2 finds an FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$, or decides that such a set does not exist, in time $O(n + m)$.*

Proof. According to Lemma 1 and Theorem 1, the algorithm correctly computes $\mathbf{Z}_{(\mathrm{i})}$ and $\mathbf{Z}_{(\mathrm{ii})}$. The algorithm verifies the conditions that $\mathbf{I} \subseteq \mathbf{Z}_{(\mathrm{ii})}$ and, by following directed edges,

[4] We defer some proofs to the appendix. Theorem 1 is proved in Appendix B.

Algorithm 2: Finding an FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ in time $O(n+m)$.

input : A DAG $G = (\mathbf{V}, \mathbf{E})$ and sets $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{I} \subseteq \mathbf{R} \subseteq \mathbf{V}$.
output : Set $\mathbf{Z}$ with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$ or $\perp$.

```
Start the Bayes-Ball (Shachter 1998) algorithm at X in G_X. Let N be the non-visited vertices.
Compute Z_(i) := N ∩ R and Z_(ii) by Algorithm 1.
Start a depth-first search (DFS) at X following only directed edges, which stops at vertices in Z_(ii) ∪ Y. Let W be the set of visited vertices.
if I ⊆ Z_(ii) and W ∩ Y = ∅ then
    return Z_(ii)
else
    return ⊥
end
```

that $\mathbf{Z}_{(\text{ii})}$ intercepts all causal paths from $\mathbf{X}$ to $\mathbf{Y}$. As Jeong, Tian, and Bareinboim (2022) have shown, this makes $\mathbf{Z}_{(\text{ii})}$ an FD set. □

Revisiting Fig. 2, the set $\mathbf{Z}_{(\text{ii})} = \{A, D\}$ indeed blocks all causal paths between X and Y and is hence a valid FD set. This can be checked by starting a DFS at X, which follows the edge $X \to A$ and does not continue from there as $A \in \mathbf{Z}_{(\text{ii})}$. As X has no other children, the search terminates.

It follows from Theorem 2, that given a DAG and sets $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$, one can verify in linear-time whether the set $\mathbf{Z}$ is an FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ by setting $\mathbf{I} = \mathbf{R} = \mathbf{Z}$ and calling Algorithm 2 as it finds $\mathbf{Z}$ if and only if $\mathbf{Z}$ is an FD set.

Moreover, with standard techniques developed by van der Zander, Liśkiewicz, and Textor (2014) and applied by Jeong, Tian, and Bareinboim (2022) to the FD criterion, it is possible to enumerate all FD sets with delay $O(n \cdot \text{find}(n, m))$, where $\text{find}(n, m)$ is the time it takes to find an FD set in a graph with n vertices and m edges. Hence, with the linear-time Algorithm 2 for finding FD sets presented in this work, an $O(n(n+m))$ delay enumeration algorithm follows directly. This improves the previous best run time of $O(n^4(n+m))$ by Jeong, Tian, and Bareinboim (2022) by a factor of n^3.

Corollary 1. *There exists an algorithm for enumerating all front-door adjustment sets in a DAG $G = (\mathbf{V}, \mathbf{E})$ with delay $O(n(n+m))$.*

A linear-time delay enumeration algorithm appears to be out-of-reach because even for the simpler tasks of enumerating d-separators and back-door adjustment sets, the best known delay is again $O(n(n+m))$ (van der Zander, Liśkiewicz, and Textor 2014).

4 Finding Minimal Front-Door Adjustment Sets

The method in the previous section guarantees us to find an FD set (if it exists) in linear-time. The set it will return, however, is the maximum size FD set, as the least amount of variables are pruned from $\mathbf{R}$ in order to satisfy condition FD(2) and FD(3). From a practical point-of-view using this set for FD adjustment appears artificial and impedes the evaluation of the FD formula. In this section, we discuss the problem of finding FD adjustment sets of small size. More precisely we aim to find *minimal* FD sets, that is, sets for which no proper subset satisfies the FD criterion. We remark that minimal FD sets are not necessarily of minimum size. E.g., in graph (ii) in Fig. 1 the FD set $\{B, C\}$ is minimal, but $\{A\}$ is also an FD set and has smaller sizer. It is an open problem to efficiently compute minimum size FD sets.

The obvious algorithm for finding a minimal FD set is to find a non-minimal set and remove vertices one-by-one until no more vertices can be removed. This trivial approach has been successfully used to find minimal back-door adjustment sets in polynomial time (van der Zander, Liśkiewicz, and Textor 2019), but it is not applicable to FD sets. Figure 1 (iii) shows a DAG with a front-door adjustment set $\mathbf{Z} = \{A, B, C\}$. The BD path $B \leftarrow D \to C \to Y$ does not violate condition FD(3), since it is not proper (in other words, $(\{A, B, C\} \perp\!\!\!\perp \{Y\} \mid \{X\})_{G_{\underline{\{A,B,C\}}}}$ holds as the path disconnects without the outgoing edges from C, which are ignored in FD(3) when $C \in \mathbf{Z}$). Removing C from the FD set turns it into a unblocked proper path, hence $\{A, B\}$ is not an FD set. Neither is $\{A, C\}$ nor $\{B, C\}$. Since no single vertex can be removed from $\mathbf{Z}$, one might believe $\mathbf{Z}$ to be minimal. However, the only minimal FD set is $\{A\}$.

While it would be possible to use a modified version of this strategy, by iteratively removing a variable W from the non-minimal set $\mathbf{Z}$ if there exists an FD set with $\mathbf{R} = \mathbf{Z} \setminus \{W\}$, a statement which could be checked for each variable using Algorithm 2, this would yield a time complexity of $O(n(n+m))$. In this section, we present a linear-time $O(n+m)$ algorithm, which moreover reveals structural insights of (minimal) FD sets. We begin with a formal definition:

Definition 2. *An FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ is* $\mathbf{I}$-inclusion minimal*, if and only if, no proper subset $\mathbf{Z}' \subset \mathbf{Z}$ with $\mathbf{I} \subseteq \mathbf{Z}'$ is an FD set relative to $(\mathbf{X}, \mathbf{Y})$. If $\mathbf{I} = \emptyset$, we call the set* (inclusion) minimal.[5]

The following lemma allows us to characterize minimal FD sets:

Lemma 3. *Let G be a DAG and $\mathbf{X}$, $\mathbf{Y}$ be disjoint sets of vertices. An FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ is $\mathbf{I}$-inclusion minimal if it can be written as $\mathbf{Z} = \mathbf{I} \cup \mathbf{Z}_{XY} \cup \mathbf{Z}_{ZY}$ such that*

1. *For each $Z \in \mathbf{Z}_{XY}$, there exists a directed proper path from $\mathbf{X}$ to Z to $\mathbf{Y}$ containing no vertex of $\mathbf{Z} \setminus \{Z\}$, and*
2. *for each $Z \in \mathbf{Z}_{ZY}$, there exists a proper BD way from $\mathbf{I} \cup \mathbf{Z}_{XY}$ to $\mathbf{Y}$ containing an edge $Z \to$ (i.e., an edge facing in the direction of $\mathbf{Y}$) but no edge $Z' \to$ for $Z' \in \mathbf{Z} \setminus \{Z\}$ which contains no vertex of $\mathbf{X}$ and each collider is opened by $\mathbf{Z}$.*

Proof. Suppose there exists a smaller FD set $\mathbf{Z}' \subset \mathbf{Z}$ with $\mathbf{I} \subseteq \mathbf{Z}'$. Let Z be a vertex of $\mathbf{Z} \setminus \mathbf{Z}'$. If Z is in $\mathbf{Z}_{\text{XY}}$, $\mathbf{Z}'$ does not block all directed paths from $\mathbf{X}$ to $\mathbf{Y}$ and is no FD set.

So $\mathbf{Z}_{\text{XY}} \subseteq \mathbf{Z}'$ and Z is in $\mathbf{Z}_{\text{ZY}}$. Let π be the BD way from $\mathbf{I} \cup \mathbf{Z}_{\text{XY}}$ to $\mathbf{Y}$ containing the edge $Z \to$ from point 2. Let

[5] In the majority of cases, we only speak of *minimal FD sets* and omit the word inclusion for $\mathbf{I} = \emptyset$.

Algorithm 3: Finding an $\mathbf{I}$-*inclusion minimal* front-door adjustment set $\mathbf{Z}$ in linear time.

input : A DAG $G = (\mathbf{V}, \mathbf{E})$ and sets $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{I} \subseteq \mathbf{R} \subseteq \mathbf{V}$.
output : Minimal FD set $\mathbf{Z}_{\min}$ with $\mathbf{I} \subseteq \mathbf{Z}_{\min} \subseteq \mathbf{R}$ or $\perp$ if no FD set exists.

1 Compute the set $\mathbf{Z}_{(\text{ii})}$ with Algorithm 2; If it outputs $\perp$, then return $\perp$ and stop.
2 Let $\mathbf{Z}_{\text{An}} \subseteq \mathbf{Z}_{(\text{ii})} \cap \boldsymbol{An}(\mathbf{Y})$ be a maximal set such that each $V \in \mathbf{Z}_{\text{An}}$ is a parent of $\mathbf{Y}$ or there exists a directed path from V to $\mathbf{Y}$ containing no vertex of $\mathbf{X} \cup (\mathbf{Z}_{(\text{ii})} \setminus \{V\})$.
3 Let $\mathbf{Z}_{\text{XY}} \subseteq \mathbf{Z}_{\text{An}} \cap \boldsymbol{De}(\mathbf{X})$ be a maximal set such that, for each $V \in \mathbf{Z}_{\text{XY}}$, there exists a directed proper path from $\mathbf{X}$ to V containing no vertex of $\mathbf{I} \cup (\mathbf{Z}_{\text{An}} \setminus \{V\})$.
4 Let $\mathbf{Z}_{\text{ZY}} \subseteq \mathbf{Z}_{\text{An}}$ be a maximal set such that, for each $V \in \mathbf{Z}_{\text{ZY}}$, there exists a proper BD way from $\mathbf{I} \cup \mathbf{Z}_{\text{XY}}$ to V containing no edge $Z_{\text{An}} \rightarrow$ with $Z_{\text{An}} \in \mathbf{I} \cup \mathbf{Z}_{\text{An}}$ and no vertex of $\mathbf{X}$, and all colliders are in $\mathbf{I} \cup \mathbf{Z}_{\text{An}}$.
5 **return** $\mathbf{Z}_{\min} := \mathbf{I} \cup \mathbf{Z}_{\text{XY}} \cup \mathbf{Z}_{\text{ZY}}$

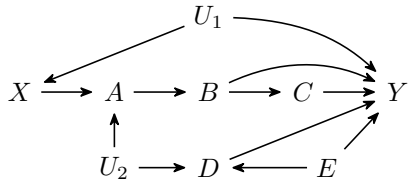

Figure 3: Example graph for finding a minimal FD set.

Z_l be the last vertex of $\mathbf{Z}'$ on π. It exists since π starts in $\mathbf{I} \cup \mathbf{Z}_{\text{XY}} \subseteq \mathbf{Z}'$. It occurs with an incoming edge $Z_l \leftarrow$ since $Z \rightarrow$ is the only outgoing edge $\mathbf{Z} \rightarrow$ on π. So the sub-way of π from Z_l to $\mathbf{Y}$ exists in $G_{\underline{\mathbf{Z}'}}$, is a BD way, and is not blocked by $\mathbf{X}$. Hence, $\mathbf{Z}'$ is no FD set. □

The intuition behind Lemma 3 is to include only vertices that are necessary for identifying the effect of $\mathbf{X}$ on $\mathbf{Y}$. The set $\mathbf{Z}_{\text{XY}}$ is needed to block causal paths between $\mathbf{X}$ and $\mathbf{Y}$, while the set $\mathbf{Z}_{\text{ZY}}$ is needed to disconnect BD paths from $\mathbf{Z}_{\text{XY}}$ to $\mathbf{Y}$. The conditions are chosen, such that each path is blocked by exactly one vertex, so no vertex from $\mathbf{Z}_{\text{XY}} \cup \mathbf{Z}_{\text{ZY}}$ can be removed from the FD set without opening a path. Algorithm 3 shows how sets $\mathbf{Z}_{\text{XY}}$ and $\mathbf{Z}_{\text{ZY}}$ can be computed, by first calling Algorithm 2 and constructing $\mathbf{Z}_{\text{An}}$ (defined in the algorithm).

Theorem 3. *Given a DAG G and sets $\mathbf{X}$ and $\mathbf{Y}$, Algorithm 3 finds an $\mathbf{I}$-inclusion minimal FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$, with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$, or decides that such a set does not exist, in linear time.*

These concepts are exemplified in Fig. 3, where the goal is to find a minimal FD set with respect to X and Y with $\mathbf{R} = \{A, B, C, D, E\}$ and $\mathbf{I} = \emptyset$. The FD set $\mathbf{Z}_{(\text{ii})} = \{A, B, C, D, E\}$ is returned by Algorithm 2. Vertices B, C, D, and E are in $\mathbf{Z}_{\text{An}}$ as they are parents of $\mathbf{Y}$ and in $\mathbf{Z}_{(\text{ii})}$. Based on this set, we first have $\mathbf{Z}_{\text{XY}} = \{B\}$ as every path from X to C goes through B and there is no path to D and E. Moreover, $\mathbf{Z}_{\text{ZY}} = \{D, E\}$ as both are needed to block BD paths from B to Y. Notably, D alone does not suffice as there would be the BD path $D \leftarrow E \rightarrow Y$ violating FD(3), i.e., the statement $(\mathbf{Z} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{X})_{G_{\underline{\mathbf{Z}}}}$. By including E, this condition is satisfied because outgoing edges from variables in $\mathbf{Z}$ are removed in $G_{\underline{\mathbf{Z}}}$. Hence $\mathbf{Z}_{\min} = \mathbf{Z}_{\text{XY}} \cup \mathbf{Z}_{\text{ZY}} = \{B, D, E\}$ is a minimal FD set.

From this theorem, we can conclude that the conditions of Lemma 3 are "if and only if" conditions:

Corollary 2. *An FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ is $\mathbf{I}$-inclusion minimal if* and only if *it can be written as $\mathbf{Z} = \mathbf{I} \cup \mathbf{Z}_{XY} \cup \mathbf{Z}_{ZY}$ such that $\mathbf{Z}_{XY}$ and $\mathbf{Z}_{ZY}$ satisfy conditions 1. and 2. given in Lemma 3.*

Proof. The "if"-direction is Lemma 3. The reverse follows from the algorithm of Theorem 3. For a given set $\mathbf{Z}$, apply the algorithm with $\mathbf{R} = \mathbf{Z}$. The algorithm returns a minimal set $\mathbf{Z}' \subseteq \mathbf{Z}$ that satisfies the conditions of Lemma 3. If $\mathbf{Z}$ is minimal, $\mathbf{Z} = \mathbf{Z}'$ and $\mathbf{Z}$ satisfies the conditions, too. □

Corollary 3. *An $\mathbf{I}$-inclusion minimal FD set $\mathbf{Z}$ relative to $(\mathbf{X}, \mathbf{Y})$ is a subset of $\mathbf{I} \cup \boldsymbol{An}(\mathbf{Y})$.*

As we show empirically in Appendix E, the minimal FD sets returned by Algorithm 3 are often much smaller compared to the maximal FD set. However, we reemphasize that they are not guaranteed to have *minimum* size, i.e., there might exist an FD set of smaller cardinality, and that it is open whether it is possible to compute such sets efficiently.

5 Experiments

We have shown that the run time of the algorithms proposed in this work scales linearly in the size of the graph. In this section, we empirically show that this translates to practical implementations, which are able to handle hundreds of thousands of variables.

We implement our methods in Python, Julia, and JavaScript to enable wide practical usage in the causal inference community.[6] In the main paper, we show the run time for the Python implementation and compare it to the author's implementation of the algorithm for finding FD sets given in Jeong, Tian, and Bareinboim (2022) (JTB). In Appendix E, we also evaluate our Julia and Javascript implementations and provide more detailed results. We ran the experiments on a single core of the AMD Ryzen Threadripper 3970X 32-core processor on a 256GB RAM machine.[7] Figure 4 shows the average run time of the algorithms in seconds. Per instance,

[6]To facilitate adoption, we base on the `networkx` Python package, which is the graph library used by causal inference packages such as `DoWhy` (Sharma, Kiciman et al. 2019). Similarly, in Julia we allow for easy integration into `CausalInference.jl` (Schauer, Keller, and Wienöbst 2023) and in Javascript into the `DAGitty` environment (Textor et al. 2016).

[7]The experiments can be replicated within a few days on a desktop computer without any problems.

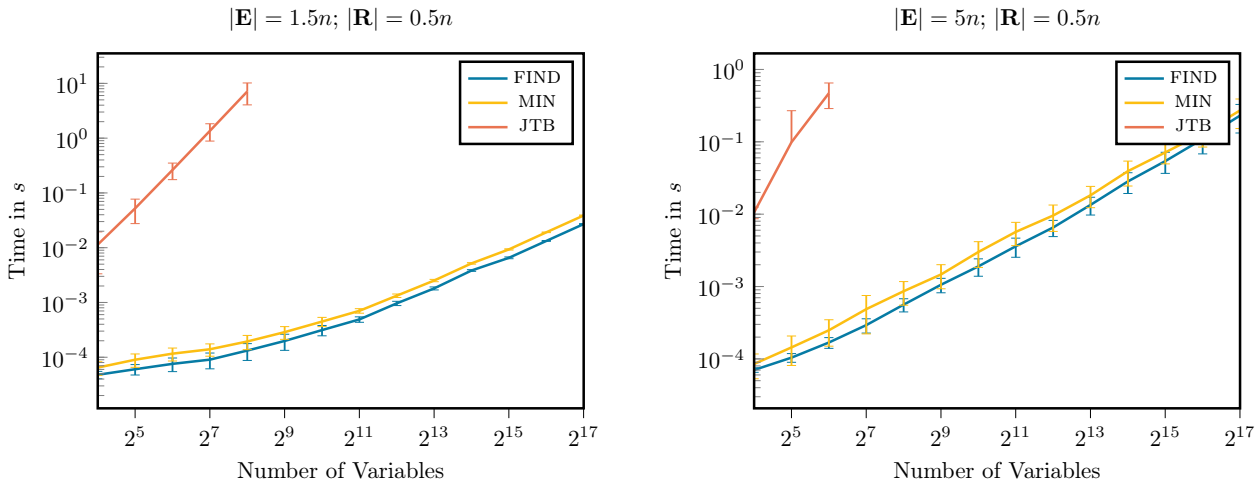

Figure 4: Log-Log plot of the average run time in seconds for (Jeong, Tian, and Bareinboim 2022) (JTB), Algorithm 2 (FIND) and Algorithm 3 (MIN) on Erdős-Rényi graphs with $1.5n$ (left) and $5n$ (right) edges, corresponding to expected vertex degree 3 and 10. $|\mathbf{X}|, |\mathbf{Y}|$ are random integers between 1 and 3; $|\mathbf{I}| = 0$ and $|\mathbf{R}| = 0.5n$. For each choice of n, we average over 50 graphs.

each algorithm was given a time limit of 30 seconds (we only report results for parameter choices for which each instance was solved within the allocated time).

The results confirm our theoretical findings as FIND (Algorithm 2) outperforms JTB by a factor of more than 10^4 on medium-sized instances (the gap widens with the number of variables). For the sparser graphs (expected degree 3), JTB does not terminate in under 30 seconds in case of more than 256 vertices; for denser graphs (expected degree 10) the performance degrades faster and only the instances up to 64 vertices are solved within the time limit. In the Appendix D, we also provide a run-time comparison for the real-life DAGs from the `bnlearn` repository (Scutari 2010), which further validate our findings, showing that FIND yields a sizeable improvement even for smaller graphs.

These gains also translate to MIN which has merely a slightly higher run time compared to FIND, while giving significantly smaller FD sets, as we report in more detail in Appendix E. In particular, the FD sets returned by FIND and JTB grow linearly in the number of vertices, even when MIN computes FD sets of size zero or one.

6 Conclusions

We have developed efficient algorithms for solving several tasks related to estimation of causal effects via FD adjustment, including finding a minimal FD set in asymptotically optimal run time. Our work shows that, from the algorithmic perspective, these tasks are not harder than for the celebrated back-door (BD) adjustment. We also offer implementations of our algorithms which significantly outperform previous methods allowing practical usage in empirical studies even for very large instances.[8]

An important open problem is whether *minimum size* FD sets can be computed efficiently. For back-door adjustment, this is the case as there exists an $O(n^3)$ algorithm (van der Zander, Liśkiewicz, and Textor 2019), while it is not clear if a polynomial-time algorithm exists in the FD case as well. Moreover, throughout this work, we assume that the causal DAG is known and correctly specified by the researcher. It would be interesting to combine our approach with causal discovery methods by developing algorithms for causal models representing a class of DAGs, e.g., a Markov equivalence class, instead of a single one. In such settings, the results could provide a means to construct more "robust" front-door adjustment sets if there are several options for DAGs which can be learned from data.

Another issue is that FD adjustment is still not as thoroughly understood as covariate adjustment, for which a *complete* graphical characterization exists (Shpitser, VanderWeele, and Robins 2010), such that a set $\mathbf{Z}$ can be used to compute the causal effect of $\mathbf{X}$ on $\mathbf{Y}$ using the covariate adjustment formula if, *and only if*, it satisfies the adjustment criterion. For the FD criterion, the analogous statement does not hold and we consider it an important direction for future work to extend it and our algorithms in this way. Finally, in the classical FD criterion it has to hold that there is no BD path from $\mathbf{X}$ to $\mathbf{Z}$. In practice, this can be rather restrictive. Recent works have relaxed this assumption (Hünermund and Bareinboim 2019; Fulcher et al. 2020) by instead demanding that a set $\mathbf{W}$ exists, which blocks all such BD paths (FD(3) is generalized in the same way) and this is also an interesting avenue for future research.

[8]The implementations and code to reproduce the experiments are available at https://github.com/mwien/frontdoor-adjustment.

Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) grant 471183316 (ZA 1244/1-1).

References

Bellemare, M. F.; Bloem, J. R.; and Wexler, N. 2019. The paper of how: Estimating treatment effects using the front-door criterion. Technical report, Working paper.

Chinco, A.; and Mayer, C. 2016. Misinformed speculators and mispricing in the housing market. *The Review of Financial Studies*, 29(2): 486–522.

Cohen, L.; and Malloy, C. J. 2014. Friends in high places. *American Economic Journal: Economic Policy*, 6(3): 63–91.

Fisher, R. A. 1936. Design of experiments. *British Medical Journal*, 1(3923): 554.

Fulcher, I. R.; Shpitser, I.; Marealle, S.; and Tchetgen Tchetgen, E. J. 2020. Robust inference on population indirect causal effects: the generalized front door criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(1): 199–214.

Glynn, A. N.; and Kashin, K. 2018. Front-door versus back-door adjustment with unmeasured confounding: Bias formulas for front-door and hybrid adjustments with application to a job training program. *Journal of the American Statistical Association*, 113(523): 1040–1049.

Gupta, S.; Lipton, Z.; and Childers, D. 2021. Estimating treatment effects with observed confounders and mediators. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, 982–991. PMLR.

Huang, Y.; and Valtorta, M. 2006. Pearl's calculus of intervention is complete. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, 217–224. AUAI Press.

Hünermund, P.; and Bareinboim, E. 2019. Causal inference and data fusion in econometrics. *arXiv preprint arXiv:1912.09104*.

Jeong, H.; Tian, J.; and Bareinboim, E. 2022. Finding and Listing Front-door Adjustment Sets. In *Proceedings of the Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS)*, 33173–33185.

Kuroki, M. 2000. Selection of post-treatment variables for estimating total effect from empirical research. *Journal of the Japan Statistical Society*, 30(2): 115–128.

Pearl, J. 1995. Causal diagrams for empirical research. *Biometrika*, 82(4): 669–688.

Pearl, J. 2009. *Causality*. Cambridge University Press. ISBN 0-521-77362-8.

Schauer, M.; Keller, M.; and Wienöbst, M. 2023. CausalInference.jl (v0.10). https://github.com/mschauer/CausalInference.jl. Accessed: 2023-08-04.

Scutari, M. 2010. Learning Bayesian Networks with the bnlearn R Package. *Journal of Statistical Software*, 35(3): 1–22.

Shachter, R. D. 1998. Bayes-Ball: The Rational Pastime (for Determining Irrelevance and Requisite Information in Belief Networks and Influence Diagrams). In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, 480–487. Morgan Kaufmann.

Sharma, A.; Kiciman, E.; et al. 2019. DoWhy: A Python package for causal inference. https://github.com/microsoft/dowhy. Accessed: 2023-08-04.

Shpitser, I.; and Pearl, J. 2006a. Identification of Conditional Interventional Distributions. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, 437–444. AUAI Press.

Shpitser, I.; and Pearl, J. 2006b. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, volume 2, 1219–1226. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Shpitser, I.; VanderWeele, T.; and Robins, J. 2010. On the Validity of Covariate Adjustment for Estimating Causal Effects. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, 527–536. AUAI Press.

Textor, J.; van der Zander, B.; Gilthorpe, M. S.; Liśkiewicz, M.; and Ellison, G. T. 2016. Robust causal inference using directed acyclic graphs: the R package 'dagitty'. *International journal of epidemiology*, 45(6): 1887–1894.

van der Zander, B.; Liśkiewicz, M.; and Textor, J. 2014. Constructing Separators and Adjustment Sets in Ancestral Graphs. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, 907–916.

van der Zander, B.; Liśkiewicz, M.; and Textor, J. 2019. Separators and adjustment sets in causal graphs: Complete criteria and an algorithmic framework. *Artificial Intelligence*, 270: 1–40.