Causal Layering via Conditional Entropy

Itai Feigenbaum* ITAI@ITAIFEIGENBAUM.COM
Salesforce AI Research
Lehman College, City University of New York
The Graduate Center, City University of New York

Devansh Arpit DEVANSHARPIT@GMAIL.COM
Shelby Heinecke SHELBY.HEINECKE@SALESFORCE.COM
Juan Carlos Niebles JNIEBLES@SALESFORCE.COM
Weiran Yao WEIRAN.YAO@SALESFORCE.COM
Huan Wang HUAN.WANG@SALESFORCE.COM
Caiming Xiong CXIONG@SALESFORCE.COM
Silvio Savarese SSAVARESE@SALESFORCE.COM
Salesforce AI Research

Editors: Francesco Locatello and Vanessa Didelez

Abstract

Causal discovery aims to recover information about an unobserved causal graph from the observable data it generates. Layerings are orderings of the variables which place causes before effects. In this paper, we provide ways to recover layerings of a graph by accessing the data via a conditional entropy oracle, when distributions are discrete. Our algorithms work by repeatedly removing sources or sinks from the graph. Under appropriate assumptions and conditioning, we can separate the sources or sinks from the remainder of the nodes by comparing their conditional entropy to the unconditional entropy of their noise. Our algorithms are provably correct and run in worst-case quadratic time. The main assumptions are faithfulness and injective noise, and either known noise entropies or weakly monotonically increasing noise entropies along directed paths. In addition, we require one of either a very mild extension of faithfulness, or strictly monotonically increasing noise entropies, or expanding noise injectivity to include an additional single argument in the structural functions.

Keywords: Causal Discovery, Entropy, Information Theory, Noise

1. Introduction

In the field of causality, data is generated by a causal graph. The purpose of causal discovery is to recover information about an unobserved causal graph via the observed data. One important task in causal discovery is the recovery of a topological ordering of the underlying graph, which in the context of causality is an ordering of the nodes which places causes before effects. Other than the importance of such ordering in its own right, it is also highly useful for discovering the full graph (Teyssier and Koller, 2005). A topological ordering which doesn't unnecessarily break all ties is called a *layering* (Tamassia, 2013).

Given a graph, repeated removal of sources or sinks yields a layering: we denote these algorithms as *repeatedSOUrceRemoval* (SOUR) and *repeatedSInkRemoval* (SIR), both are simple and probably known variants of Kahn's Algorithm (Kahn, 1962). Of course, in causal discovery, we are

* The author is currently affiliated with the City University of New York.

not given the graph, so implementing SOUR/SIR is not straightforward. In this paper, we propose a new method for causal discovery of layerings of discrete random variables, which implements SOUR/SIR without direct access to the graph, but with access to a conditional entropy oracle for the data instead. We show that, under some assumptions, we can separate sources from non-sources and sinks from non-sinks by comparing their conditional entropy (with appropriate conditioning) to their unconditional noise entropy. Specifically, when repeatedly removing sources and conditioning on all removed variables, we show that the conditional entropy of new sources equals the unconditional entropy of their noise, while the conditional entropy of non-sources is larger than the entropy of their noise. On the other hand, when repeatedly removing sinks and conditioning on all non-removed variables, we show that the conditional entropy of new sinks equals the unconditional entropy of their noise, while the conditional entropy of non-sinks is smaller than the entropy of their noise. Under our assumptions, our algorithms are provably correct and have a polynomial (quadratic) worst-case running-time.

We divide our assumptions to two: (i) assumptions not on noise entropies and (ii) assumptions on noise entropies. (i) includes faithfulness and injective noise (weaker than additive noise), while (ii) requires either that the noise entropies are known to us in advance, or—if the noise entropies are unknown—that the noise entropies are weakly monotonically increasing along directed paths. In addition, we require one of either a very mild extension of faithfulness, or strictly monotonically increasing noise entropies, or expanding noise injectivity to include an additional single argument in the structural functions. The known noise entropy assumption is of course much weaker than assuming known noise distribution. The weak monotonicity assumption is much weaker than the i.i.d. noise assumption, which received some attention in literature (Peters and Bühlmann, 2014; Peters et al., 2009; Xie et al., 2019). The assumptions we are not making are as important as those made: unlike many other existing methods of causal discovery, we are making no assumption about the topology of the causal graph such as sparsity, we do not assume any particular functional form such as linearity, and we do not assume any particular noise distribution.

The rest of the paper is organized as follows. Section 1.1 reviews related literature. Section 2 provides necessary definitions and preliminaries. Section 3 collates all of the assumptions made in the paper. Section 4 shows how to bound the conditional entropy of a variable with its unconditional noise entropy. Section 5 translates those bounds into causal discovery implementations of SOUR and SIR. Section 6 concludes.

1.1. Related Literature

Causal discovery is a rich and evolving field. Several detailed surveys of causal discovery exist (Zanga et al., 2022; Vowels et al., 2022; Nogueira et al., 2022; Spirtes and Zhang, 2016), as well as a book on the subject (Peters et al., 2017). As it is known that full causal discovery becomes much easier given a layering, finding such a layering plays a key role in many papers (Ruiz et al., 2022; Sanchez et al., 2022; Montagna et al., 2023; Teyssier and Koller, 2005). Information theory has been used in causal discovery for a variety of purposes (Kocaoglu et al., 2020; Branchini et al., 2023; Cabeli et al.; Runge, 2018; Marx, 2021).

There is some existing work which implements SOUR by using entropy. A somewhat related approach to ours is entropic causal inference (ECI) (Compton et al., 2022). Like our method, ECI also implements SOUR by considering noise entropy, but in a very different way. First, the main assumption ECI makes on noise entropy is that it is small, as opposed to our assumption of either

known or monotonic noise entropy. Second, ECI assumes access to both a conditional independence oracle and a minimum entropy coupling oracle, unlike our method which assumes access to a conditional entropy oracle. An interesting similarity is that ECI's conditional independence testing uses the same conditioning set we use for conditional entropy in SOUR. Another somewhat related work is by Xie et al. (2019), which studies the much more specific case of a linear model with i.i.d. noise. They too identify sources in SOUR using entropy minimization. Their method seems very different than ours, but it is possible—although we haven't been able to determine this—that it is a special case of our more general method (applied to linear models with i.i.d. noise). We note that a potential advantage of using entropy for discrete variables is its invariance under scaling, which means that our methods are not sensitive to data standardization. This stands in contrast to some existing algorithms which inadvertently (implicitly or explicitly) discover layerings by using patterns in the marginal variance in non-standardized simulated data (Reisach et al., 2021).

Post acceptance of our paper to the conference, we were made aware of relevant work by Gao and Aragam (2021), which has interesting connections to our causal SOUR implementation (their work does not include an implementation of SIR). Like us, the authors implement SOUR via a conditional entropy oracle, with a different algorithm and under different assumptions. In every iteration, both implementations compute the entropy of each variable conditional on all removed variables. However, while our algorithm uses these conditional entropies directly to discover new sources to be removed, their algorithm uses them to generate a sequence of conditional independence tests via additional calls to the oracle. Their algorithm has the advantage of working under a weaker form of our increasing noise entropies assumption, while our algorithm has the advantage of working under the known noise entropies assumption (and being somewhat simpler conceptually). There are therefore two cases where both algorithms work, in which we can try and compare their behavior. The first case is when both their weaker variant of the increasing noise entropies assumption and the known noise entropies assumption hold. In that case, if we make our algorithm remove all sources it detects immediately, both algorithms have the exact same output, but our algorithm performs fewer oracle calls compared to their algorithm. The second case is when (our stronger variant of) the increasing noise entropies assumption holds, in which case the two algorithms may output different layerings, and neither algorithm dominates the other in terms of the number of oracle calls.

2. Preliminaries

2.1. Graph Theory

Before we move to causal discovery, let us begin with the realm of graph theory, where the graph is known. For a directed acyclic graph (DAG) $G = (V, E)$, we denote the edge (v_i, v_j) as $v_i \to v_j$ (or equivalently $v_j \leftarrow v_i$). Note that in a DAG, at most one of the edges $v_i \to v_j$ and $v_i \leftarrow v_j$ can exist: we write $v_i - v_j$ when we want to refer to one of the edges $v_i \to v_j$ or $v_i \leftarrow v_j$ but not specify the direction, so the statement $v_i - v_j \in E$ should be read as "$v_i \to v_j \in E$ or $v_i \leftarrow v_j \in E$". For $v \in V$, we define $Par(v, G) = \{v' \in V : v' \to v \in E\}$ as the set of parents of v, and $Des(v, G) = \{v' \in V : v' \neq v$ and $\exists$ a directed path from v to v' in $G\}$ as the set of descendants of v. For any subsets $X, Y, S \subseteq V$, we denote as $X \vdash_G Y | S$ the event where X and Y are d-separated in G conditional on S (for information about d-separation, see Geiger et al. (1990)). For any $\widehat{V} \subseteq V$, we define the induced subgraph w.r.t. $\widehat{V}$ as $G^{\widehat{V}} = (\widehat{V}, \{v \to v' \in E : v \in \widehat{V}$ and $v' \in \widehat{V}\})$ as the subgraph of G corresponding to the nodes in $\widehat{V}$ and the edges between them. For a node $v \in V$,

a parent $v' \in Par(v)$ that doesn't have any other parent as a descendant $Des(v') \cap Par(v) = \emptyset$ is called an *unmediated parent* of v. Nodes with only outgoing or only incoming edges are called sources and and sinks, respectively:

Definition 1 (sources and sinks) *Let $G = (V, E)$ be a DAG. $v \in V$ is called a source in G if v has no incoming edges, and a sink if v has no outgoing edges. We denote the sets of all sources and sinks in G respectively as $SRC(G)$ and $SNK(G)$.*

Next, we define layerings (Tamassia, 2013), which are simply DAG topological orderings that allow for unbroken ties:

Definition 2 (layerings) *Let $G = (V, E)$ be a digraph. Let $L = (L_1, \ldots, L_m)$ be a tuple of mutually exclusive non-empty subsets of V ($L_i \cap L_j = \emptyset$ whenever $i \neq j$); we slightly abuse notation and overload L to also mean $L = \cup_{i=1}^{m} L_i$ when clear from context. We write $L_i < L_j$ iff $i < j$ (and similarly define $>, =, \leq, \geq$). For every $v \in V$, if $v \in L_i$, define $L(v) = L_i$. L is a layering iff it is a partition of V and for all $v, v' \in V$, $v \to v' \in E \Rightarrow L(v) < L(v')$.*

A digraph has a layering iff it is a DAG. Repeated removal of sets of sources and/or sinks yields a layering; we precisely define this process and prove its correctness in Algorithm 5 and Theorem 5 in Appendix A.[1] We refer to the special case where only sources (but not necessarily all sources) are removed in each iteration as the *repeatedSOUrceRemoval (SOUR) Algorithm*. Alternatively, we refer to the case where only sinks are removed as *repeatedSInkRemoval (SIR) Algorithm*. We provide the pseudocode for SOUR and SIR as Algorithms 1 and 2 respectively.

Algorithm 1 repeatedSOUrceRemoval (SOUR)

Input: DAG $G = (V, E)$
Output: Layering L of G

$V_{cur} \leftarrow V$
$L \leftarrow$ empty sequence
while $V_{cur} \neq \emptyset$ **do**
  $SR \leftarrow$ non-$\emptyset$ subset of $SRC(G^{V_{cur}})$
  $L.append(SR)$
  $V_{cur} \leftarrow V_{cur} - SR$
end while
return L

Algorithm 2 repeatedSInkRemoval (SIR)

Input: DAG $G = (V, E)$
Output: Layering L of G

$V_{cur} \leftarrow V$
$L \leftarrow$ empty sequence
while $V_{cur} \neq \emptyset$ **do**
  $SN \leftarrow$ non-$\emptyset$ subset of $SNK(G^{V_{cur}})$
  $L.prepend(SN)$
  $V_{cur} \leftarrow V_{cur} - SN$
end while
return L

2.2. Causality

Our goal in this paper is to recover a causal DAG's layering without knowing the graph, but with access—via a conditional entropy oracle—to data generated by it. As we will show, under some assumptions, SOUR and SIR can be implemented in this scenario. Throughout this paper, let $G_c = (V_c, E_c)$ be a DAG, which we refer to as the *causal graph*. We slightly abuse notation and consider the nodes in V_c also as random variables. We assume that the variables in V_c are connected through a structural causal model (SCM), meaning that for each $v \in V_c$, $v = f_v(Par(v, G_c) \cup \{N_v\})$

1. The literature we found technically describes special cases of Algorithm 5, but their correctness proofs very easily extend to it; we provide this extension in the appendix for the reader's convenience. We believe it is highly likely that Algorithm 5 as we wrote it already exists in literature, but we could not find a reference.

for some function f_v and noise variable N_v independent of all other noise variables. To clarify, $f_v(Par(v, G_c) \cup \{N_v\})$ is also a slight abuse of notation, which when $Par(v, G_c) = \{p_1, \ldots, p_t\}$ means $f_v(p_1, \ldots, p_t, N_v)$ (using the appropriate order of arguments). We assume that all noise variables (and hence all variables) are discrete, so the definitions of entropy and conditional entropy apply.

G_c is unknown to us, but we assume access to a conditional entropy oracle $\mathscr{H}$. For sets of random variables X, Y and S, let $\mathscr{H}(X|S)$ be the conditional entropy of X conditional on S. We denote as $X \perp\!\!\!\perp Y|S$ the case where X and Y are independent conditional on S. In the notations $\mathscr{H}(X|S)$, $X \perp\!\!\!\perp Y|S$, and $X \vdash_G Y|S$, we allow replacing singleton sets with their element (e.g. if $X = \{x\}$ then $\mathscr{H}(x|S) = \mathscr{H}(X|S)$ etc.), and when $S = \emptyset$ we allow dropping it from the notation (e.g. $\mathscr{H}(X) = \mathscr{H}(X|\emptyset)$ etc.). Whenever we drop the graph from notation, the underlying graph we refer to is G_c (e.g. we allow writing $Par(v)$ instead of $Par(v, G_c)$, $\vdash$ instead of $\vdash_{G_c}$, etc.). We also define the explicit noise graph, which makes the noise terms into explicit nodes:

Definition 3 (explicit noise graph) *The explicit noise graph $G_c^N = (V_c^N, E_c^N)$ is obtained from G_c by adding a node N_v for each $v \in V_c$, with exactly one adjacent edge $N_v \to v$. That is, $V_c^N = V_c \cup \{N_v : v \in V_c\}$ and $E_c^N = E_c \cup \{N_v \to v : v \in V_c\}$.*

We point out that d-separation implies conditional independence (Pearl, 2000):

Theorem 1 *For $X, Y, S \subseteq V_c$, $X \vdash_{G_c} Y|S \Rightarrow X \perp\!\!\!\perp Y|S$ and similarly for every $X, Y, S \subseteq V_c^N$, $X \vdash_{G_c^N} Y|S \Rightarrow X \perp\!\!\!\perp Y|S$.*

3. Assumptions

Our layering discovery method requires some assumptions, which we list here. We break our assumptions to three sets. Assumption Collection 1 contains *global* assumptions, which we implicitly make throughout the rest of the paper from this point on. Assumption Collections 2 and 3 contain *local* assumptions, which are **not implicitly assumed** but instead are only assumed when explicitly stated; we collate them here as a convenient reference point. Let us first introduce our global assumptions:

Assumption Collection 1 (global assumptions) *We always make the following assumptions.*

- *1.1 Faithfulness: For $X, Y, S \subseteq V_c$, $X \perp\!\!\!\perp Y|S \Rightarrow X \vdash Y|S$.*
- *1.2 Injective noise: For all $v \in V_c$, when holding the values of $Par(v)$ constant, f_v as a function of N_v is one-to-one.*
- *1.3 Non-constant noise: For every $v \in V_c$, N_v is not constant, or equivalently $\mathscr{H}(N_v) > 0$.*

Assumption 1.1 is common throughout the causal discovery literature (Spirtes et al., 2000), Assumption 1.2 is weaker than the Additive Noise Assumption (Hoyer et al., 2008), and Assumption 1.3 simply prevents some degenerate cases. Next, we state two local assumptions, not about noise entropy, which are used in different parts of Theorem 2 in Section 4 to bound the conditional entropy of variables by the entropy of their noise.

Assumption Collection 2 (non noise entropy local assumptions) *We occasionally make some of the following assumptions.*

2.1 *Injective noise plus one: For all $v \in V$, $v' \in Par(v)$, when holding the values of $Par(v) - \{v'\}$ constant, f_v as a function of v' and N_v is one-to-one.*

2.2 *Directed-faithfulness: For every $v \in V$, $v' \in Des(v)$, $N_v \not\perp\!\!\!\perp v'$.*

Assumption 2.1 implies Assumption 1.2, and Assumption 2.2 is weaker than faithfulness on the explicit noise graph. Finally, we state another list of local assumptions, about noise entropy, which are used for our implementation of SOUR and SIR via a conditional entropy oracle in Section 5. These assumptions allow us to use the bounds established by Theorem 2 to extract layering information.

Assumption Collection 3 (noise entropy local assumptions) *We occasionally make some of the following assumptions.*

3.1 *Known noise entropy: For all $v \in V$, $\mathscr{H}(N_v)$ is known to us.*

3.2 *Weakly increasing noise entropy: For all $v \in V$, $v' \in Des(v)$, $\mathscr{H}(N_v) \leq \mathscr{H}(N_{v'})$.*

3.3 *Strictly increasing noise entropy: For all $v \in V$, $v' \in Des(v)$, $\mathscr{H}(N_v) < \mathscr{H}(N_{v'})$.*

Assumption 3.1 is clearly much weaker than assuming the noise distribution is known in advance. Assumption 3.2 is satisfied in the case of i.i.d. noise, but is of course a weaker requirement than i.i.d. noise. The i.i.d. noise case received some attention in literature (Peters and Bühlmann, 2014; Peters et al., 2009; Xie et al., 2019). Assumption 3.3 is a bit stronger than 3.2, and rules out the i.i.d. noise case.

Next, we discuss some intuition behind the assumptions and how they are used in Sections 4 and 5. Assumption Collection 2 guarantees strict separation of sources/sinks from non-sources/non-sinks in SOUR/SIR: sources/sinks have conditional entropy equal to their unconditional noise's entropy, while non-sources/non-sinks do not. Assumption 2.1 means that a variable always carries information about its children. In SOUR, non-sources differ from sources by the existence of an unconditioned unmediated parent, and Assumption 2.1 makes sure that excluding this parent from the conditioning set increases the entropy. Assumption 2.2 can be regarded as a minor technicality, to prevent unrealistic cases where a variable's noise impacts the variable but not its descendants. With that assumption in place, a variable carries information not just about its ancestors, but also about their noise. In SIR, non-sinks differ from sinks by the existence of at least one descendant in the conditioning set, and Assumption 2.2 makes sure that including this descendant in the conditioning set decreases the entropy.

Assumption Collection 3 is meant to help us use the separation from Assumption Collection 2 for causal discovery. Assumption 3.1 is straightforward, and intuitively means that while the noise is unobserved, it comes from a well-understood origin. In SOUR/SIR, knowledge of noise entropy allows us to compare each variable's conditional entropy to its unconditional noise's entropy: the variable is a source/sink iff the two entropies are equal. Furthermore, our separation result has directionality built into it: all non-sources/non-sinks deviate from their unconditional noise entropy in the same direction (larger for non-sources in SOUR, smaller for non-sinks in SIR). Assumptions 3.2 or 3.3 allow us to get around the need for Assumption 3.1 by enabling us to utilize this directionality to detect sources/sinks. Finally, Figure 1 summarizes the sufficiency results we establish in this paper.

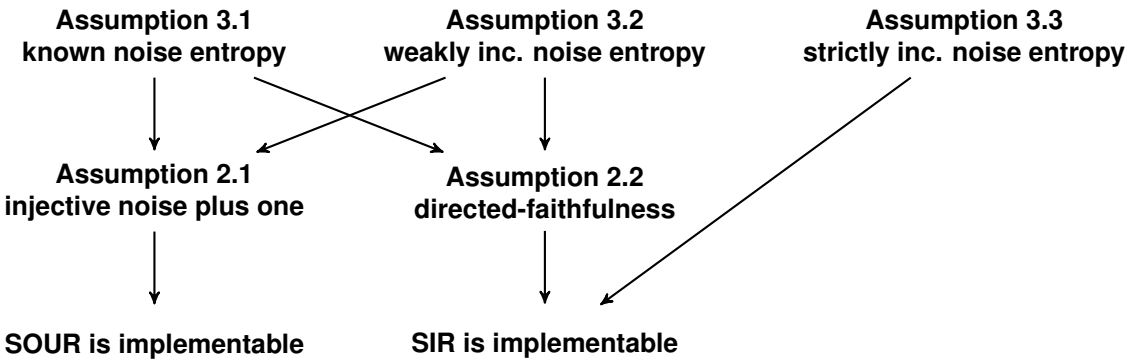

Figure 1: Local assumptions flowchart. The arrows do not represent logical implication. The source nodes in this flowchart (not to be confused with the sources we try to discover in SOUR) are assumptions from Assumption Collection 3. The sink nodes in this flowchart (not to be confused with the sinks we try to discover in SIR) state that SIR or SOUR is implementable using a conditional entropy oracle. Every directed path from a source node to a sink node in the chart represent sufficient local assumptions for sink node to hold: for example, the path "Assumption 3.1" $\rightarrow$ "Assumption 2.2" $\rightarrow$ "SIR is implementable" means that Assumptions 3.1 and 2.2 together are sufficient for SIR to be implementable using a conditional entropy oracle. Assumption Collection 1 is always implicitly assumed.

4. Entropy Bounds

Let us first provide some informal intuition for our bounds. The value of $v \in V_c$ is fully determined by $Par(v)$ and N_v. When we condition on $Par(v)$, N_v remains the only source of randomness for v. If we don't condition on anything else, then since noise injectivity implies that f_v doesn't dilute N_v's entropy, we get that the conditional entropy of v equals $\mathscr{H}(N_v)$. If the conditioning set includes—in addition to $Par(v)$—some descendant $u \in Des(v)$, then u generally carries additional information about N_v and we can expect the entropy to be reduced relatively to $\mathscr{H}(N_v)$. On the other hand, if the conditioning set excludes some unmediated parent $r \in Par(v)$, then r is a source of randomness additional to N_v. If the conditioning set also excludes all descendants of r (and thus also all descendants of v), then no additional information is given about N_v or r, so (using noise-plus-one injectivity to prevent f_v from diluting the entropy) we can generally expect a conditional entropy larger than $\mathscr{H}(N_v)$.

Theorem 2 formalizes the intuition above. In Section 5, we use it to detect sinks and sources in graphs via a conditional entropy oracle. First, we introduce a lemma:

Lemma 1 *Let $v \in V_c$ and $S \subseteq V_c - \{v\}$ s.t. $Des(v, G_c) \cap S = \emptyset$. Then $N_v \vdash_{G_c^N} S$, and therefore $N_v \perp\!\!\!\perp S$.*

Proof Consider any path of undirected edges $N_v = u_0 - u_1 - u_2 - \cdots u_k = s$ between N_v and a node $s \in S$. The adjacent edge to N_v must necessarily be the edge $N_v \rightarrow v$, since this is the only edge adjacent to N_v in the graph; in particular, $u_1 = v$. Since s is not a descendant of v, the remainder of the path $u_1 = v - \cdots - s = u_k$ must contain at least one edge of the form $u_i \leftarrow u_{i+1}$; choose i to be the minimum value so that $u_i \leftarrow u_{i+1}$ exists in the path. Since we have established $i \geq 1$, then the previous edge $u_{i-1} - u_i$ exists and must be oriented as $u_{i-1} \rightarrow u_i$; therefore, u_i

is a collider in the path, and since we are not conditioning on anything, neither u_i nor any of its descendants are conditioned on. Therefore, we have shown that $N_v \vdash_{G_c^N} S$, and therefore $N_v \perp\!\!\!\perp S$ (by Theorem 1). ■

We can now state and prove Theorem 2.

Theorem 2 (entropy bounds) *Let $v \in V_c$ and let $S \subseteq V - \{v\}$. Then:*

2.1 Assume $Par(v) \subseteq S$. Then $\mathscr{H}(v|S) \leq \mathscr{H}(N_v)$. (Conditioning on a node's parents yields a weakly lower entropy than the noise's.)[2]

2.2 Assume $Par(v) \subseteq S$ and $Des(v) \cap S = \emptyset$. Then $\mathscr{H}(v|S) = \mathscr{H}(N_v)$. (Conditioning on a node's parents but no descendants yields exactly the same entropy as the noise's.)

2.3 Assume Assumption 2.2. Assume $Par(v) \subseteq S$ and $Des(v) \cap S \neq \emptyset$. Then $\mathscr{H}(v|S) < \mathscr{H}(N_v)$. (Conditioning on a node's parents and some descendants yields a strictly lower entropy than the noise's.)

2.4 Assume Assumption 2.1. Assume that there exists $v' \in Par(v) - S$ s.t. $Des(v') \cap (S \cup Par(v)) = \emptyset$. Then $\mathscr{H}(v|S) > \mathscr{H}(N_v)$. (Failing to condition on at least one unmediated parent of a node and that parent's descendants yields a strictly higher entropy than the noise's.)

Proof

2.1 Compute:

$$\begin{aligned}
&\mathscr{H}(v|S) \\
&= \mathscr{H}(f_v(Par(v) \cup \{N_v\})|S) \\
&\leq \mathscr{H}(Par(v) \cup \{N_v\}|S) \\
&= \mathscr{H}(Par(v)|S) + \mathscr{H}(N_v|S \cup Par(v)) && \text{chain rule} \\
&= 0 + \mathscr{H}(N_v|S) && Par(v) \subseteq S \\
&= \mathscr{H}(N_v|S) \\
&\leq \mathscr{H}(N_v).
\end{aligned}$$

2.2 Compute:

$$\begin{aligned}
&\mathscr{H}(v|S) \\
&= \mathscr{H}(f_v(Par(v) \cup \{N_v\})|S) \\
&= \mathscr{H}(Par(v) \cup \{N_v\}|S) && Par(v) \subseteq S \text{ and Assumption 1.2} \\
&= \ldots = \mathscr{H}(N_v|S) && \text{as in the proof of Theorem 2.1} \\
&= \mathscr{H}(N_v). && N_v \perp\!\!\!\perp S \text{ by Lemma 1}
\end{aligned}$$

2. Theorem 2.1 does not actually require Assumption 1.2, but we will only be using this part of the theorem in conjunction with Theorem 2.2 which does require that assumption, so there is no need to separate the cases further.

2.3 Compute: Assumption 2.2 implies that N_v is dependent on S (since S contains at least one element from $Des(v)$), and therefore $\mathscr{H}(N_v|S) < \mathscr{H}(N_v)$. So now we have:

$$\mathscr{H}(v|S) \leq \mathscr{H}(N_v|S) < \mathscr{H}(N_v),$$

where the first inequality is by the proof of Theorem 2.1.

2.4 We make two initial observations:

(1) $N_v \perp\!\!\!\perp S \cup Par(v)$. This is because $Des(v') \cap S = \emptyset$ and $Des(v) \subset Des(v')$, so $Des(v) \cap S = \emptyset$ and trivially $Des(v) \cap Par(v) = \emptyset$. Thus Lemma 1 implies $N_v \perp\!\!\!\perp S \cup Par(v)$.

(2) $N_{v'} \perp\!\!\!\perp S \cup (Par(v) - \{v'\}) \cup Par(v')$. This is because $Des(v') \cap (S \cup (Par(v) - \{v'\})) = \emptyset$ by assumption, and trivially $Des(v') \cap Par(v') = \emptyset$, so Lemma 1 implies $N_{v'} \perp\!\!\!\perp S \cup (Par(v) - \{v'\}) \cup Par(v')$.

Compute:

$$\begin{aligned}
&\mathscr{H}(v|S) \\
&= \mathscr{H}(f_v(Par(v) \cup \{N_v\})|S) \\
&= \mathscr{H}(f_v((Par(v) - \{v'\}) \cup \{N_v, v'\})|S) \\
&\geq \mathscr{H}(f_v((Par(v) - \{v'\}) \cup \{N_v, v'\})|S \cup (Par(v) - \{v'\})) \\
&= \mathscr{H}(N_v, v'|S \cup (Par(v) - \{v'\})) && \text{Assumption 2.1} \\
&= \mathscr{H}(N_v|S \cup Par(v)) + \mathscr{H}(v'|S \cup (Par(v) - \{v'\})) && \text{chain rule} \\
&= \mathscr{H}(N_v) + \mathscr{H}(v'|S \cup (Par(v) - \{v'\})) && \text{Observation (1)}
\end{aligned}$$

To complete our proof, it is sufficient to show that $\mathscr{H}(v'|S \cup (Par(v) - \{v'\})) > 0$:

$$\begin{aligned}
&\mathscr{H}(v'|S \cup (Par(v) - \{v'\})) \\
&= \mathscr{H}(f_{v'}(Par(v') \cup \{N_{v'}\})|S \cup (Par(v) - \{v'\})) \\
&\geq \mathscr{H}(f_{v'}(Par(v') \cup \{N_{v'}\})|S \cup (Par(v) - \{v'\}) \cup Par(v')) \\
&= \mathscr{H}(N_{v'}|S \cup (Par(v) - \{v'\}) \cup Par(v')) && \text{Assumption 1.2} \\
&= \mathscr{H}(N_{v'}) && \text{Observation (2)} \\
&> 0 && \text{Assumption 1.3}
\end{aligned}$$

■

5. Causal Layering Algorithms

In this section, we use Theorem 2 to implement SOUR and SIR when we can only access the graph via a conditional entropy oracle. Looking at the psuedocode of SOUR in Algorithm 1 (resp. SIR in

Algorithm 2), we see that information about edges is only used in line 4, to identify a non-empty subset of sources (resp. sinks). With certain assumptions and conditioning, sources' (resp. sinks') conditional entropy is equal to their noise entropy, while the conditional entropy of non-sources (resp. non-sinks) is larger (resp. smaller) than their noise entropy. We can use this separation to implement line 4 without knowledge of the underlying graph, but with a conditional entropy oracle. SOUR can be implemented subject to Assumptions 2.1 and either 3.1 or 3.2. SIR can be implemented subject to Assumptions 2.2 and either 3.1 or 3.2. SIR can also be implemented subject just to Assumption 3.3.

Algorithm 3 SOUR (Causal Discovery)

Assumptions: 2.1 & (3.1 or 3.2)
Input: Variables V_c, entropy oracle $\mathscr{H}$
Output: A Layering L of G_c

1: $V_{cur} \leftarrow V_c$
2: $L \leftarrow$ empty sequence
3: **while** $V_{cur} \neq \emptyset$ **do**
4: **if** As. 2.1 & 3.1 hold **then**
5: $SR \leftarrow$ non-$\emptyset$ subset of $\{v \in V_{cur} : \mathscr{H}(v|V_c - V_{cur}) = \mathscr{H}(N_v)\}$
6: **else if** As. 2.1 & 3.2 hold **then**
7: $SR \leftarrow$ non-$\emptyset$ subset of $\arg\min_{v \in V_{cur}} \mathscr{H}(v|V_c - V_{cur})$
8: **end if**
9: $L.append(SR)$
10: $V_{cur} \leftarrow V_{cur} - SR$
11: **end while**
12: **return** L

Algorithm 4 SIR (Causal Discovery)

Assumptions: 2.2 & (3.1 or 3.2), or 3.3
Input: Variables V_c, entropy oracle $\mathscr{H}$
Output: A Layering L of G_c

1: $V_{cur} \leftarrow V_c$
2: $L \leftarrow$ empty sequence
3: **while** $V_{cur} \neq \emptyset$ **do**
4: **if** As. 2.2 & 3.1 hold **then**
5: $SN \leftarrow$ non-$\emptyset$ subset of $\{v \in V_{cur} : \mathscr{H}(v|V_{cur} - \{v\}) = \mathscr{H}(N_v)\}$
6: **else if** As. 2.2 & 3.2 or As. 3.3 hold **then**
7: $SN \leftarrow$ non-$\emptyset$ subset of $\arg\max_{v \in V_{cur}} \mathscr{H}(v|V_{cur} - \{v\})$
8: **end if**
9: $L.prepend(SN)$
10: $V_{cur} \leftarrow V_{cur} - SN$
11: **end while**
12: **return** L

Algorithms 3 and 4 present the causal discovery implementations of SOUR and SIR respectively. By $\arg\min$/$\arg\max$ we mean the set of all minimizing/maximizing arguments and not just an arbitrary one (in case the minimizing/maximizing argument is unique, then $\arg\min$/$\arg\max$ is a singleton set). Lines 4-8 in Algorithms 3 and 4 replace line 4 in Algorithms 1 and 2. Therefore, to guarantee the correctness of the causal discovery implementations, we must show that the replacement lines accomplish the same function as the original line 4. Specifically, we need to show that the replacement lines produce a non-empty subset of $SRC(G_c^{V_{cur}})$ for SOUR and $SNK(G_c^{V_{cur}})$ for SIR. This is where Theorem 2 comes in handy. Theorem 3 proves that Algorithms 3 and 4 implement SOUR and SIR (Algorithms 1 and 2 respectively), thus—by Theorem 5 in the appendix—Algorithms 3 and 4 produce a layering of G_c:

Theorem 3 (correctness of Algorithms 3 and 4) *The following statements hold whenever the relevant algorithm reaches line 4. By SOUR we refer to Algorithm 3 and by SIR we refer to Algorithm 4.*

3.1 Assume As. 2.1. In SOUR, $SRC(G_c^{V_{cur}}) = \{v \in V_{cur} : \mathscr{H}(v|V_c - V_{cur}) = \mathscr{H}(N_v)\}$.

3.2 Assume As. 2.1 & 3.2. In SOUR, $\arg\min_{v \in V_{cur}} \mathscr{H}(v|V_c - V_{cur}) \subseteq SRC(G_c^{V_{cur}})$.

3.3 Assume As. 2.2. In SIR, $SNK(G_c^{V_{cur}}) = \{v \in V_{cur} : \mathscr{H}(v|V_{cur} - \{v\}) = \mathscr{H}(N_v)\}$.

3.4 Assume As. 2.2 & 3.2, or 3.3. In SIR, $\arg\max_{v \in V_{cur}} \mathscr{H}(v|V_{cur} - \{v\}) \subseteq SNK(G_c^{V_{cur}})$.

Proof Note that V_{cur} is monotonically shrinking throughout iterations in both SOUR and SIR. For each proof, we assume the algorithm worked correctly up until the current iteration: that is, in all previous iterations, $SR \subseteq SRC(G_c^{V_{cur}})$ for SOUR and $SN \subseteq SNK(G_c^{V_{cur}})$ for SIR (where V_{cur} is the value of V_{cur} at that iteration). That is, SOUR only removed sources and SIR only removed sinks from the induced subgraph of each previous iteration.

3.1 Let $v \in V_{cur}$. Consider $v \in SRC(G_c^{V_{cur}})$. Since v is a source, all nodes in $Par(v, G_c)$ are no longer in V_{cur}, and thus $Par(v, G_c) \subseteq V_c - V_{cur}$. On the other hand, since $v \in V_{cur}$, v has never been removed and so the nodes in $Des(v, G_c)$ were never sources in previous iterations, and therefore $Des(v, G_c) \subseteq V_{cur}$, thus $Des(v, G_c) \cap (V_c - V_{cur}) = \emptyset$. Theorem 2.2 therefore implies that $\mathscr{H}(v|V_c - V_{cur}) = \mathscr{H}(N_v)$.

Consider instead $v \notin SRC(G_c^{V_{cur}})$. Since $v \notin SRC(G_c^{V_{cur}})$, then $Par(v, G_c) \cap V_{cur} \neq \emptyset$, so there exists some $v^* \in Par(v, G_c) \cap V_{cur}$. We claim that there exists an unmediated parent $v' \in Par(v, G_c) \cap V_{cur}$ of v, meaning that $Des(v', G_c) \cap Par(v, G_c) = \emptyset$. We will construct a finite sequence $u_0, u_1, u_2, \ldots, u_k$ which satisfies the following properties:

- $u_0 = v^*$
- $u_{i+1} \in Des(u_i, G_c)$ for all i
- $u_i \in Par(v, G_c) \cap V_{cur}$ for all i
- $Des(u_k, G_c) \cap Par(v, G_c) = \emptyset$

Once we construct this sequence, we can set $v' = u_k$ and our claim is proven. As v' is an ancestor of all nodes in $Des(v', G_c)$ and $v' \in V_{cur}$, then none of the nodes in $Des(v', G_c)$ have been removed either (they were never sources since they have an unremoved ancestor), meaning $Des(v', G_c) \subseteq V_{cur}$ so $Des(v', G_c) \cap (V_c - V_{cur}) = \emptyset$. Since also $v' \in Par(v, G_c) \cap V_{cur}$ and $Des(v', G_c) \cap Par(v, G_c) = \emptyset$, we can apply Theorem 2.4 to establish $\mathscr{H}(v|V_c - V_{cur}) > \mathscr{H}(N_v)$.

All that is left is to construct the sequence. For every i, if $Des(u_i, G_c) \cap Par(v, G_c) = \emptyset$, we can simply set $k = i$ and end the sequence. If $Des(u_i, G_c) \cap Par(v, G_c) \neq \emptyset$, then there exists some $u_{i+1} \in Des(u_i, G_c) \cap Par(v, G_c)$; furthermore, since $u_i \in V_{cur}$ and u_i is an ancestor of u_{i+1}, it follows that u_{i+1} was never removed from V_{cur} either, and therefore $u_{i+1} \in Par(v, G_c) \cap V_{cur}$. Since each element of the sequence is a descendant of the previous one, the sequence is moving down a directed path (potentially skipping some nodes along the path), and since G_c is acyclic, this means that the sequence contains no repetitions. As the V_c is finite, the sequence must be finite, and therefore it must end with some u_i satisfying $Des(u_i, G_c) \cap Par(v, G_c) = \emptyset$, so k must be finite.

3.2 Let $u \in V_{cur}$. Our proof of Theorem 3.1 established that if $q \in SRC(G_c^{V_{cur}})$, then $\mathscr{H}(q|V_c - V_{cur}) = \mathscr{H}(N_q)$, and if $q \notin SRC(G_c^{V_{cur}})$, then $\mathscr{H}(q|V_c - V_{cur}) > \mathscr{H}(N_q)$. Assume $u \notin SRC(G_c^{V_{cur}})$. In that case, $\mathscr{H}(u|V_c - V_{cur}) > \mathscr{H}(N_u)$; also, u has at least one ancestor $u' \in$

$SRC(G_c^{V_{cur}})$, and for that ancestor $\mathscr{H}(u'|V_c - V_{cur}) = \mathscr{H}(N_{u'})$. Assumption 3.2 implies $\mathscr{H}(N_{u'}) \leq \mathscr{H}(N_u)$, and therefore we have

$$\mathscr{H}(u'|V_c - V_{cur}) = \mathscr{H}(N_{u'}) \leq \mathscr{H}(N_u) < \mathscr{H}(u|V_c - V_{cur}).$$

Thus, $u \notin \arg\min_{v \in V_{cur}} \mathscr{H}(v|V_c - V_{cur})$.

3.3 Let $v \in V_{cur}$. Since $v \in V_{cur}$, and v is a child of all nodes in $Par(v, G_c)$, then no node in $Par(v, G_c)$ has been a sink in any previous iteration, and therefore $Par(v, G_c) \subseteq V_{cur}$, so $Par(v, G_c) \subseteq V_{cur} - \{v\}$. If $v \in SNK(G_c^{V_{cur}})$, then $Des(v, G_c) \cap V_{cur} = \emptyset$, and since also $Par(v, G_c) \subseteq V_{cur} - \{v\}$, Theorem 2.2 implies that $\mathscr{H}(v|V_{cur} - \{v\}) = \mathscr{H}(N_v)$. If instead $v \notin SNK(G_c^{V_{cur}})$, then $Des(v, G_c) \cap V_{cur} \neq \emptyset$, so by Theorem 2.3 $\mathscr{H}(v|V_{cur} - \{v\}) < \mathscr{H}(N_v)$.

3.4 Let $u \in V_{cur}$. First, consider the case where Assumptions 2.2 & 3.2 hold. Because Assumption 2.2 holds, our proof of Theorem 3.3 established that if $q \in SNK(G_c^{V_{cur}})$, then $\mathscr{H}(q|V_{cur} - \{q\}) = \mathscr{H}(N_q)$, and if $q \notin SNK(G_c^{V_{cur}})$, then $\mathscr{H}(q|V_{cur} - \{q\}) < \mathscr{H}(N_q)$. Assume $u \notin SNK(G_c^{V_{cur}})$. In that case, $\mathscr{H}(u|V_{cur} - \{u\}) < \mathscr{H}(N_u)$; also, u has at least one descendant $u' \in SNK(G_c^{V_{cur}})$, and for that descendant $\mathscr{H}(u'|V_{cur} - \{u'\}) = \mathscr{H}(N_{u'})$. Assumption 3.2 implies $\mathscr{H}(N_{u'}) \geq \mathscr{H}(N_u)$, and therefore we have

$$\mathscr{H}(u'|V_{cur} - \{u'\}) = \mathscr{H}(N_{u'}) \geq \mathscr{H}(N_u) > \mathscr{H}(u|V_{cur} - \{u\}).$$

Thus, $u \notin \arg\max_{v \in V_{cur}} \mathscr{H}(v|V_{cur} - \{v\})$.

Alternatively, assume Assumption 3.3. In the proof of Theorem 3.3, we have shown that if $q \in SNK(G_c^{V_{cur}})$ then $\mathscr{H}(q|V_{cur} - \{q\}) = \mathscr{H}(N_q)$. That claim did not rely on Assumption 2.2 and therefore still holds. On the other hand, since we only ever remove sinks, all the parents of nodes in V_{cur} are also still in V_{cur}, and therefore Theorem 2.1 implies that $\mathscr{H}(q|V_{cur} - \{q\}) \leq \mathscr{H}(N_q)$ for all $q \in V_{cur}$. Assume $u \notin SNK(G_c^{V_{cur}})$. As we have established, $\mathscr{H}(u|V_{cur} - \{u\}) \leq \mathscr{H}(N_u)$; also, u has at least one descendant $u' \in SNK(G_c^{V_{cur}})$, and for that descendant $\mathscr{H}(u'|V_{cur} - \{u'\}) = \mathscr{H}(N_{u'})$. Assumption 3.3 implies $\mathscr{H}(N_{u'}) > \mathscr{H}(N_u)$, and therefore we have

$$\mathscr{H}(u'|V_{cur} - \{u'\}) = \mathscr{H}(N_{u'}) > \mathscr{H}(N_u) \geq \mathscr{H}(u|V_{cur} - \{u\}).$$

Thus, $u \notin \arg\max_{v \in V_{cur}} \mathscr{H}(v|V_{cur} - \{v\})$.

■

Finally, we present a straightforward observation regarding the running time of our algorithms. Note that the bound given is worst-case.

Theorem 4 (running time of Algorithms 3 and 4) *Algorithms 3 and 4 make $O(|V_c|^2)$ oracle calls.*

Proof In every iteration, the algorithms perform one oracle call for each element in V_{cur}. Furthermore, in the beginning $V_{cur} = V_c$, and in every iteration at least one element is removed from V_{cur}. Thus the number of oracle calls is bounded from above by $\sum_{i=1}^{|V_c|} i = O(|V_c|^2)$. ■

6. Conclusion

In this paper, we introduced a new method of causal discovery algorithms for discrete data. Our algorithms recover a layering of the causal graph, which they can only access via a conditional entropy oracle. In fact, our algorithms implement the SOUR and SIR algorithms from graph theory, but without direct access to the graph. The key idea behind our algorithms is that, with appropriate assumptions and conditioning, sources and sinks can be separated from the other nodes in the graph based on comparison between their conditional entropy and the unconditional entropy of their noise. The sources in SOUR and sinks in SIR have conditional entropy equal to the unconditional entropy of their noise. On the other hand, the non-sources in SOUR have conditional entropy larger than the unconditional entropy of their noise, while the non-sinks in SIR have conditional entropy smaller than the unconditional entropy of their noise. Our implementations of SOUR and SIR are provably correct and make $O(|V_c|^2)$ oracle calls in the worst-case. Our algorithms do not make many of the assumptions that are commonly made in literature, but they do need to make an assumption on noise entropies, namely that they are either known or monotonically increasing.

Acknowledgments

We thank Yu Bai for fruitful discussions.

References

Nicola Branchini, Virginia Aglietti, Neil Dhir, and Theodoros Damoulas. Causal entropy optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 8586–8605. PMLR, 2023.

Vincent Cabeli, Honghao Li, Marcel da Câmara Ribeiro-Dantas, Franck Simon, and Hervé Isambert. Reliable causal discovery based on mutual information supremum principle for finite datasets. In *WHY21 workshop, 35rd Conference on Neural Information Processing Systems*.

Spencer Compton, Kristjan Greenewald, Dmitriy A Katz, and Murat Kocaoglu. Entropic causal inference: Graph identifiability. In *International Conference on Machine Learning*, pages 4311–4343. PMLR, 2022.

Ming Gao and Bryon Aragam. Efficient bayesian network structure learning via local markov boundary search. *Advances in Neural Information Processing Systems*, 34:4301–4313, 2021.

Dan Geiger, Thomas Verma, and Judea Pearl. d-separation: From theorems to algorithms. In *Machine Intelligence and Pattern Recognition*, volume 10, pages 139–148. Elsevier, 1990.

Patrik Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. *Advances in neural information processing systems*, 21, 2008.

Arthur B Kahn. Topological sorting of large networks. *Communications of the ACM*, 5(11):558–562, 1962.

Murat Kocaoglu, Sanjay Shakkottai, Alexandros G Dimakis, Constantine Caramanis, and Sriram Vishwanath. Applications of common entropy for causal inference. *Advances in neural information processing systems*, 33:17514–17525, 2020.

Alexander Marx. Information-theoretic causal discovery. 2021.

Francesco Montagna, Nicoletta Noceti, Lorenzo Rosasco, Kun Zhang, and Francesco Locatello. Causal discovery with score matching on additive models with arbitrary noise. *arXiv preprint arXiv:2304.03265*, 2023.

Ana Rita Nogueira, Andrea Pugnana, Salvatore Ruggieri, Dino Pedreschi, and João Gama. Methods and tools for causal discovery and causal inference. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 12(2):e1449, 2022.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, 2000.

Jonas Peters and Peter Bühlmann. Identifiability of gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, 2014.

Jonas Peters, Dominik Janzing, Arthur Gretton, and Bernhard Schölkopf. Detecting the direction of causal time series. In *Proceedings of the 26th annual international conference on machine learning*, pages 801–808, 2009.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

Alexander Reisach, Christof Seiler, and Sebastian Weichwald. Beware of the simulated dag! causal discovery benchmarks may be easy to game. *Advances in Neural Information Processing Systems*, 34:27772–27784, 2021.

Gabriel Ruiz, Oscar Hernan Madrid Padilla, and Qing Zhou. Sequentially learning the topological ordering of causal directed acyclic graphs with likelihood ratio scores. *arXiv preprint arXiv:2202.01748*, 2022.

Jakob Runge. Conditional independence testing based on a nearest-neighbor estimator of conditional mutual information. In *International Conference on Artificial Intelligence and Statistics*, pages 938–947. PMLR, 2018.

Pedro Sanchez, Xiao Liu, Alison Q O'Neil, and Sotirios A Tsaftaris. Diffusion models for causal discovery via topological ordering. *arXiv preprint arXiv:2210.06201*, 2022.

Peter Spirtes and Kun Zhang. Causal discovery and inference: concepts and recent methodological advances. In *Applied informatics*, volume 3, pages 1–28. SpringerOpen, 2016.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Roberto Tamassia. *Handbook of graph drawing and visualization*. CRC press, 2013.

Marc Teyssier and Daphne Koller. Ordering-based search: a simple and effective algorithm for learning bayesian networks. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pages 584–590, 2005.

Matthew J Vowels, Necati Cihan Camgoz, and Richard Bowden. D'ya like dags? a survey on structure learning and causal discovery. *ACM Computing Surveys*, 55(4):1–36, 2022.

Feng Xie, Ruichu Cai, Yan Zeng, Jiantao Gao, and Zhifeng Hao. An efficient entropy-based causal discovery method for linear structural equation models with iid noise variables. *IEEE transactions on neural networks and learning systems*, 31(5):1667–1680, 2019.

Alessio Zanga, Elif Ozkirimli, and Fabio Stella. A survey on causal discovery: theory and practice. *International Journal of Approximate Reasoning*, 151:101–129, 2022.

Appendix A. Repeated Removal of Sources/Sinks

Repeated removal of sources/sinks in a DAG yields a layering; we precisely define this process in Algorithm 5, which we call the *Repeated Removal (RR) Algorithm*. This is a simple generalized variant of the well-known Kahn's Algorithm (Kahn, 1962), but we include a correctness proof here for completeness.

Algorithm 5 RepeatedRemoval (RR)

Input: DAG $G = (V, E)$
Output: Layering L of G

$V_{cur} \leftarrow V$
$L_{start} \leftarrow$ empty sequence
$L_{end} \leftarrow$ empty sequence
while $V_{cur} \neq \emptyset$ **do**
  $SR, SN \leftarrow$ subsets of $SRC(G^{V_{cur}}), SNK(G^{V_{cur}})$ s.t. $SR \cup SN \neq \emptyset$
  $L_{start}.append(SR)$
  $L_{end}.prepend(SN)$
  $V_{cur} \leftarrow V_{cur} - (SR \cup SN)$
end while
$L \leftarrow concatenate(L_{start}, L_{end})$
return L

Theorem 5 (correctness of RR) *RR outputs a layering of its input (regardless of the specific choices made for SR and SN in each iteration).*

Proof Suppose $v \to v' \in E$. We need to show that $L(v) < L(v')$. We make the following useful observation: as long as $v, v' \in V_{cur}$, $v \notin SN$ and $v' \notin SR$. We break into cases:

1. Assume $v \in L_{start}$. If $v' \in L_{end}$, the result is trivial. If $v' \in L_{start}$, then our observation implies that v' must have been **appended** to L_{start} at a strictly later iteration than v. Therefore, v' is appended to L_{start} when v is already in L_{start}, implying $L_{start}(v) < L_{start}(v')$.

2. Assume $v \in L_{end}$. Then our observation implies that $v' \in L_{end}$, and furthermore that v must be added to L_{end} at a strictly later iteration than v'. Therefore, v is **prepended** to L_{end} when v' is already in L_{end}, implying $L_{end}(v) < L_{end}(v')$.

■