A Appendix

A.1 Algorithm for Finding Simple Instruments

function FIND-SIMPLE-IVS($\mathcal{G}$, $\mathbf{X}$, Y)
  Let $\mathbf{B}$ be all nodes d-separated from Y in $\overline{\mathcal{G}}$
  Let $\mathbf{D} = De(An(\mathbf{X}))$
  Construct a flow graph $F'(\mathcal{G})$ with nodes:
   $\{V^+ \mid V \in \mathbf{D}\} \cup \{V^- | V \in An(\mathbf{X})\} \cup \{S, Y\}$
  and edges:
   $\{V^+ \to W^+ \mid V, W \in An(\mathbf{D}) \text{ and } V \leftarrow W \in \mathbf{E}\} \cup$
   $\{V^+ \to V^- \mid V \in An(\mathbf{X})\} \cup$
   $\{V^- \to W^- \mid V, W \in An(\mathbf{X}) \text{ and } V \to W \in \mathbf{E}\} \cup$
   $\{S \to V^+ \mid V \in (\mathbf{D} \cap \mathbf{M} \cap \mathbf{B})^+\} \cup$
   $\{X^- \to Y \mid X \in \mathbf{X}\}$
  Assign capacities to nodes in $\mathcal{G}'$:
    infinite capacity to S, Y;
    zero capacity to nodes of $Pa(Y) \setminus \mathbf{X}$ and
    unit capacity to all other nodes.
  if an $|\mathbf{X}|$-flow from S to Y in $F'(\mathcal{G})$ exists **then**
    return First nodes of the flow
  else return $\perp$

Figure 7: Find-Simple-IVs

A.2 Algorithm for Testing Generalized Instrumental Sets

The algorithm presented in Fig. 8 tests if $\mathbf{Z}$ is a generalized instrumental set relative to $\mathbf{X}$ and Y. For a given input it constructs consecutively instances of k'-GVDPP, with $k \leq k' \leq 2k$, and for each of them it uses a subroutine for solving GVDPP. The algorithm returns TRUE if at least one k'-GVDPP instance has a solution.

In the algorithm we use an auxiliary procedure, called ADD-ARC, which, to improve the readability, we define separately. The aim of this procedure is to compute a consecutive triple S_p, T_p, and $\mathbf{W}'_p$ for an instance of a k'-GVDPP as well as to update the set $\mathbf{C}$ appropriately.

Below we present a pseudocode for ADD-ARC:

procedure ADD-ARC($i, S, T, \mathbf{W}$)
  $S_p := S$; $T_p := T$;
  if $T = Z_i$ **then**
          ▷ Path towards the instrument
    $\mathbf{W}'_p := \mathbf{W} \cup \mathbf{Z} \cup \{X_j \mid i \geq j \vee F_j = X_j\}$
    $\mathbf{U} := \mathbf{U} \cup \{p\}$
  else
          ▷ Path towards Y
    $\mathbf{W}'_p := \mathbf{W} \cup \{Z_j \mid i \leq j \vee F_j = Z_j\} \cup \mathbf{X}$
    $\mathbf{C} := \mathbf{C} \cup \{\{u, p\} \mid u \in \mathbf{U}\}$
  $\mathbf{W}'_p := \mathbf{W}'_p \setminus \{S, T\}$
  $p := p + 1$

function TEST-GENERAL-IVS($\mathcal{G}$, $\mathbf{X}$, Y, $\mathbf{Z}$)
  for i in $1, \ldots, k$ **do**
    $\mathbf{W}_i :=$ a nearest separator for (Y, Z_i) in $\overline{\mathcal{G}}$
    if $\mathbf{W}_i = \perp \vee (\mathbf{W}_i \cap De(Y) \neq \emptyset)$ **then**
      return FALSE
  for all permutations π and π' of $\{1, \ldots, k\}$ **do**
    Let $(X_1, \ldots, X_k)$ corresponds to π and let
    $(Z_1, \ldots, Z_k)$, $(\mathbf{W}_1, \ldots, \mathbf{W}_k)$ correspond to π'
    if $\exists i : X_i \in \mathbf{W}_i$ **then continue**
    if $\exists i, j : i < j \wedge X_i = Z_j$ **then continue**
    for all $F_1, \ldots, F_k \in \mathbf{V} \setminus (Y \cup \mathbf{W}_1 \cup ... \mathbf{W}_k)$ **do**
      if $\exists i \neq j : F_i \in \{F_j, X_j, Z_j\}$ **then**
        continue
      $\mathbf{F} := \{F_i \mid 1 \leq i \leq k\}$
      $k' := k + |\mathbf{F} \setminus (\mathbf{X} \cup \mathbf{Z})|$
      Construct a k'-GVDPP instance
       $S_1, \ldots, S_{k'}, T_1, \ldots, T_{k'}, \mathbf{W}'_1, \ldots, \mathbf{W}'_{k'}, \mathbf{C}$:
      $p := 1$; $\mathbf{U} := \emptyset$; $\mathbf{C} := \emptyset$;
      for i in $1, \ldots, k$ **do**
        if $F_i = X_i$ **then**
          ADD-ARC $(i, X_i, Z_i, \mathbf{W}_i \cup \mathbf{F} \cup Y)$
        else if $F_i = Z_i$ **then**
          ADD-ARC $(i, Z_i, X_i, \mathbf{W}_i \cup \mathbf{F} \cup Y)$
        else
          ADD-ARC $(i, F_i, X_i, \mathbf{W}_i \cup \mathbf{F} \cup Y)$
          ADD-ARC $(i, F_i, Z_i, \mathbf{W}_i \cup \mathbf{F} \cup Y)$
      if $(S_1, \ldots, S_{k'}, T_1, \ldots, T_{k'}, \mathbf{W}'_1, \ldots, \mathbf{W}'_{k'}, \mathbf{C})$
       is a Yes-instance of k'-GVDPP
       then return TRUE
  return FALSE

Figure 8: Test-General-IVs

In the next section, we prove (see proof of Proposition A.3) that algorithm TEST-GENERAL-IVS satisfies the requirements of Lemma 5.6.

A.3 Proofs

Proof of Lemma 5.2. Since Y can only occur once in π, π contains only one edge $\mathbf{X} \to Y$ and thus $\pi[Z \sim X]$ also exists in the graph $\overline{\mathcal{G}}$.

If π is blocked by $\mathbf{W}$ and not by $\mathbf{W}'$ there is a blocking node $V \in \mathbf{W} \setminus \mathbf{W}'$ on π. Since $V \in \mathbf{W} \subseteq An(Y \cup Z)$, the subpath between V and Z exists in the moral graph $((\overline{\mathcal{G}})_{An(Y \cup Z)})^m$. Since $\mathbf{W}$ is a nearest separator, it can only contain V, if $\mathbf{W}'$ also contains it, which contradicts $\mathbf{W}'$ not blocking π. ☐

Proposition A.1. *Algorithm* TEST-SIMPLECOND-INSTRU-MENTS *(Fig. 5) satisfies the requirements of Theorem 4.2.*

Proof. Assume $\mathbf{Z}$ is a simple conditional instrumental set, then there exist a set $\mathbf{W}'$ and pairs (Z_i, π_i) satisfying Def. 3.3. Because all paths from S pass through a node of $\mathbf{Z}$, $\mathbf{W}'$ d-separates S and Y in $\mathcal{G}'$. According to lemma 5.1 calling algorithm Nearest-Separating-Set finds a nearest-separator set $\mathbf{W}$ with $(Y \perp\!\!\!\perp S \mid \mathbf{W})_{\mathcal{G}'}$, which does not contain descendants of Y. Due to Lemma 5.2 this set does not block any path π_i.

Every path π_i can be split into two (possible empty) paths π_i^+ and π_i^-. The first is directed away from $\mathbf{Z}$ and the second is directed towards Y. π_i^+ corresponds to a directed path π_i^+ of plus nodes in $F(\mathcal{G})$ and π_i^- to a directed path of minus nodes. A possibly existing fork F_i corresponds to a subpath $F_i^+ \to F_i^-$. These paths exist in $F(\mathcal{G})$, because π_i^+ only contains ancestors of $\mathbf{Z}$ and π_i^- only ancestors of $\mathbf{X}$.

These paths do not intersect. If, for $i < j$, there were two paths π'_i and π'_j containing a common node V^+, the corresponding paths π_i, π_j would have the form $Z_i \overset{+}{\leftarrow} V$, $Z_j \overset{+}{\leftarrow} V$ or $V \in \{Z_i, Z_j\}$. Which violates condition *(c)* of Def. 3.3, since $\pi_j[Z_j \sim V]$ has to point towards V. The same argument holds for V^-. Since each of these paths can carry an unit flow, a $|\mathbf{Z}|$-flow exists and the algorithm returns true.

In the other direction: if the algorithm returns true, it has found a set $\mathbf{W}$ and a $|\mathbf{Z}|$-flow from S to Y. The set $\mathbf{W}$ d-separates every Z_i from Y in $\overline{\mathcal{G}}$, because it d-separates S from Y. The flow can be represented as $k = |\mathbf{Z}|$ disjoint paths $\pi'_1, \ldots, \pi'_k$ from S to Y not blocked by $\mathbf{W}$. We can assume that the paths are chosen, such that their total length is minimal[5]. We can project these paths to d-connected paths $\pi_1, \ldots, \pi_k$ in $\mathcal{G}$, by dropping the $\pm$ markers from nodes, the first edge and replacing $\to V^+ \to V^- \to$ with a single fork $\leftarrow V \to$.

The first two conditions of Def. 3.3 are satisfied by the construction, and because no element of $\mathbf{Z}$ can be a descendant of Y, unless there is a path in $\overline{\mathcal{G}}$ that can only be blocked by a descendant of Y.

If path π_i intersects path π_j, there is a common node V, which corresponds to V^+ and V^- in π'_i and π'_j. W.l.o.g let V^- be in π'_j, so $V \to$ occurs in π_j, which leads to a partial ordering $\pi_i \prec \pi_j$.

If these partial orderings cannot be combined to a valid total ordering, there is a cycle of k paths $\pi_{i_1} \prec \ldots \prec \pi_{i_k} \prec \pi_{i_1}$ intersecting at nodes $V_{i_1}, \ldots, V_{i_k}$ with $V_{i_j} \leftarrow$ in π_{i_j} and $V_{i_j} \to$ in $\pi_{i_{j+1}}$ (we set $i_{k+1} = i_1$). Since the construction of $F(\mathcal{G})$ can neither lead to colliders in the projected paths nor intersections at forks, since a path containing a fork contains the $+$ and $-$ variant of the node, the paths $\pi_{i_{j+1}}$ have the form $Z_{i_{j+1}} \overset{*}{\leftarrow} V_{i_{j+1}} \leftarrow \overset{*}{-} \to V_{i_j} \overset{*}{\to} X_{i_{j+1}}$. So from every node V_{i_j} there exist directed paths to nodes in $\mathbf{X}$ and $\mathbf{Z}$. Thus we can replace each path π_{i_j} with a path $Z_{i_j} \overset{*}{\leftarrow} V_{i_j} \overset{*}{\to} X_{i_{j+1}}$. These new paths are shorter than the original paths, violating the initial assumption (this also holds if $V_{i_j} = Z_{i_j}$, $V_{i_j} = X_{i_j}$ or the new paths intersect themselves).

So we have pairs (Z_i, π_i), where $Z_i \in \mathbf{Z}$ is the first node of π_i.

If two paths π_i, π_j with $i < j$ intersect in a node V, π'_i contains V^+ and π'_j contains V^-. Since π'_j starts in Z_j^+, V cannot be Z_j and Z_j does not occur in π_i. Because V cannot be a fork, π_i contains $V \leftarrow$ and π_j contains $\to V \to$, which satisfies condition *(c)*.

The runtime is given by the maximum runtime of the nearest separator algorithm and the maximum flow algorithm, which can both run in $\mathcal{O}(nm)$. ☐

Proposition A.2. *Algorithm* Find-Simple-Instruments *(Fig. 7) satisfies the requirements of Theorem 4.1 .*

Proof. We will base the proof on the algorithm Test-SimpleCond-Instruments.

Assume there exists simple instruments $\mathbf{Z}$. Then $\mathbf{Z} \subseteq De(An(\mathbf{X})) \cap \mathbf{M} \cap \mathbf{B}$, so $F'(\mathcal{G})$ is a supergraph of $F(\mathcal{G})$ and the flow found by Test-SimpleCond-Instruments also exists in $F'(\mathcal{G})$. Thus Find-Simple-Instruments returns a set.

Assume Find-Simple-Instruments returns a set $\mathbf{Z}$. Then it satisfies the conditions of Def. 3.1 that do not depend on paths. Algorithm Find-Simple-Instruments will choose the empty set for $\mathbf{W}$. The flow found by Find-Simple-Instruments can be represented as k disjoint paths $\pi_1, \ldots, \pi_k$ starting at S and pointing to nodes of $\mathbf{M}$. Each path $\pi_i = V_1^+ \ldots V_{l_i}^-$ consists of a (positive empty) subpath of plus nodes followed by a subpath of minus nodes, each node being associated with a node $De(An(\mathbf{X}))$ of $\mathcal{G}$. Due to the construction of $F'(\mathcal{G})$, every node V_j^+ on the plus subpath is an ancestor of the node $V_1 \in \mathbf{Z}$ associated with the first node V_1^+. Likewise every node V_j^- belongs to an ancestor of $\mathbf{X}$. Thus all the nodes participating in the flow also exists in $F(\mathcal{G})$ and Test-SimpleCond-Instruments returns true. ☐

Proof of Lemma 5.4. If there exists a solution, it is easy to see that the game can be won by moving pebble p_i along the path p_i, whenever pebble p_i can be moved.

If the game can be won, the pebbles trace paths p_i through the graph that satisfy the conditions of k-

[5] Such paths can be found efficiently by a min-cost-max-flow algorithm.

GVDPP: Each p_i is a directed path from S_i to T_i and does not contain a node of $\mathbf{W}_i$. Two paths p_i, p_j with $\{i,j\} \notin \mathbf{C}$ cannot intersect each other in a non-endpoint node, otherwise there would be a node X that is visited by pebble p_i and p_j. Assume w.l.o.g. p_i moves to node X first. Then the game cannot be won, since pebble p_j on an ancestor of X must move before pebble p_i, but cannot move to node X as long as p_i is on that node. □

Proof of Lemma 5.5. There are only $\mathcal{O}((n+1)^k)$ different configurations of pebble placements and $\mathcal{O}((n+1)k)$ different transitions of one pebble moving to another node, so a reachability search on the graph of all configurations can be done in $\mathcal{O}(k(n+1)^{k+1})$. □

Proposition A.3. *Algorithm* Test-General-Instru-ments *(Fig. 8) satisfies the requirements of Lemma 5.6.*

Proof. If $\mathbf{Z}$ is a generalized instrumental set relative to $\mathbf{X}$ and Y, algorithm Test-General-IVs will return true: Let $(Z_1, \mathbf{W}_1, \pi_1), \ldots, (Z_k, \mathbf{W}_k, \pi_k)$ be the triples of Def. 3.2. We consider the iteration in which the algorithm tests the same permutation of $\mathbf{X}, \mathbf{Z}$. These triples remain valid, if we replace the $\mathbf{W}_i$ with the nearest separating sets used by the algorithm, because a nearest separator does not contain descendants of Y and neither blocks path π_i nor contains X_i if such a set exists due to Lemmas 5.2 and 5.1. There is no $i < j$ with $X_i = Z_j$, since Z_j must not appear in path π_i.

Now consider the iteration in which each F_i is the fork on path π_i or, if π_i does not have a fork and is a directed path from Z_i to Y, is Z_i. Since the paths end at Y, this fork cannot be Y. It cannot be in $\mathbf{W}_i$ or π_i was blocked. A path π_i cannot intersect π_j at a fork, since no arrow at a fork points to that fork violating condition three of Def. 3.2. So the condition $\exists i \neq j : F_i \in \{F_j, X_j, Z_j\}$ is false, and the algorithm does not abort there. The set $\mathbf{F}$ is the set of all forks, including Z_i for paths π_i without fork.

The algorithm now creates a $(k' = k + |\mathbf{F} \setminus (\mathbf{X} \cup \mathbf{Z})|)$-GVDPP instance, whose solution are the paths π_i, whereby paths containing a fork are split into two directed paths $\pi_{i,\leftarrow}, \pi_{i,\rightarrow}$ and the tail $\rightarrow Y$ of every path is removed. It does this by calling Add-Arc for all endpoint node pairs of the split paths in the innermost loop.

The sets $\mathbf{W}'_i$ created by Add-Arc do not contain a non-endpoint node of the respective $\pi_{i,\leftarrow}$ or $\pi_{i,\rightarrow}$: This is true for the initial $\mathbf{W}_i \cup \mathbf{F} \cup Y$ as argued above. An arc $\pi_{i,\leftarrow}$ towards Z_i cannot contain another Z_j. For $j > i$ this follows directly from Def. 3.2 *(c)*, for $j < i$, $\pi_{i,\leftarrow}[Z_i \sim Z_j]$ should point towards Z_j, but this path is directed towards Z_i. For the same reason, it cannot contain a node X_j with $j < i$. It also cannot contain X_j if X_j is a fork on π_j. An arc $\pi_{i,\rightarrow}$ cannot contain a Z_j with $j > i$ due to condition *(c)* of Def. 3.2. If π_j is a directed path from Z_j to Y, π_i cannot contain Z_j with $j < i$ as $\pi_j[Z_j \sim Y]$ should point towards Z_j. π_i cannot contain a X_j with $j < i$ because then $\pi_j[X_j \sim Y]$ should point towards X_j, but is an edge $X_j \rightarrow Y$. For $j > i$, $\pi_i[X_j \sim Y]$ should point towards X_j, but all edges on $\pi_{i,\rightarrow}$ point towards Y. Since π_i is a path, X_i, Z_i can only occur in one of the arcs $\pi_{i,\leftarrow}, \pi_{i,\rightarrow}$.

Also all intersecting arcs are included in the set $\mathbf{C}$: The set $\mathbf{U}$ contains the indices of all upwards directed arcs, i.e. when Add-Arc for $\pi_{i,\rightarrow}$ is called, $\mathbf{U}$ contains the indices of all $\pi_{j,\leftarrow}$ with $j < i$. Arcs directed in the same direction, i.e. $\pi_{i,\rightarrow}$ and $\pi_{j,\rightarrow}$ or $\pi_{i,\leftarrow}$ and $\pi_{j,\leftarrow}$, cannot intersect each other, since the edges at an intersection have to point in opposite directions. So only intersections between an upward $\pi_{i,\leftarrow}$ and a downward directed arc $\pi_{j,\rightarrow}$ occur, with $i < j$ due to condition *(c)* of Def. 3.2 . These are precisely the arcs contained in $\mathbf{C}$.

Thus the generated k'-GVDPP instance is solvable and the algorithm returns true.

In the other direction, if the algorithm returns true, $\mathbf{Z}$ is a general instrumental set: Then the algorithm has fixed permutations $X_1, \ldots, X_k$, $Z_1, \ldots, Z_k$, sets $\mathbf{W}_1, \ldots, \mathbf{W}_k$ and paths $\pi_1 = \pi_{1,\leftarrow}\pi_{1,\rightarrow} \rightarrow Y, \ldots \pi_k = \pi_{k,\leftarrow}\pi_{k,\rightarrow} \rightarrow Y$, where $\pi_{i,\leftarrow}, \pi_{i,\rightarrow}$ denote the paths in the solution of k'-GVDPP for the nodes added by the call of Add-Arc for i in the respective direction. $\mathbf{W}_i$ does not contain descendants of Y, d-separates Z_i from Y in $\overline{\mathcal{G}}$ and does not block π_i.

For $1 \leq i < j \leq k$ the node Z_j does not occur as non-endpoint node in π_i, since it was added to the forbidden nodes in $\mathbf{W}'_\cdot$. If paths π_i and π_j have a common node, this node is either an endpoint node (of an arc) or the indices of the corresponding arcs were added to $\mathbf{C}$.

If two endpoints nodes are the same, this node cannot be a fork in π_i or π_j, since there is no F_i with $F_i \in \{F_j, X_j, Z_j\}$ for $i \neq j$. There is also no $Z_i = Z_j$ or $X_i = X_j$, thus the intersecting node has to be $Z_i = X_j$ with $i \leq j$ due to the test after the $\mathbf{X}$ permutation. If $i = j$, it is not an intersection. Because $F_i \neq Z_i$ and $F_j \neq X_j$, both $\pi_i[Z_i \sim Y]$ and $\pi_j[Z_j \sim X_j]$ point towards $Z_i = X_j$ satisfying 3.2 *(c)*.

If it is only an endpoint node in one arc, it can only be a node of $\mathbf{X} \cup \mathbf{Z}$, because all other endpoint nodes are in $\mathbf{F}$ and thus forbidden. Within $\pi_{i,\leftarrow}$ only a node X_j

with $i < j$ and $F_j \neq X_j$ can occur, leading to subpaths in π_i and π_j of $\leftarrow X_j \leftarrow$ and $\rightarrow X_j \rightarrow$. Within $\pi_{i,\rightarrow}$ only Z_j with $i > j$ and $F_j \neq Z_j$ occurs, leading to subpaths $\rightarrow Z_j \rightarrow$ and $Z_j \leftarrow$. Both intersections are allowed by Def. 3.2 *(c)*.

If the paths intersect at a non-endpoint node V, it only occurs in arcs with indices in $\mathbf{C}$, i.e. $\pi_{i,\leftarrow}$ and $\pi_{j,\rightarrow}$ with $i < j$. Thus the subpaths are $\leftarrow V \leftarrow$ and $\rightarrow V \rightarrow$, which is also allowed by Def. 3.2 *(c)*.

Thus all conditions of Def. 3.2 are satisfied and $\mathbf{Z}$ is a generalized instrumental set. □

Proof of Corollary 5.7. This follows from lemma 5.5 and 5.6. The time spend to construct a k'-GVDPP instance is negligible compared to the time needed to solve it. Note that we can change $(n+1)$ to n as the node Y cannot occur in the paths. □

A.4 Singleton Sets as Active Instruments

If a set $\mathbf{Z} = \{Z\}$ contains only a single node, it obviously is a generalized instrumental set if and only if it is a simple conditional instrumental set, so the algorithm Test-SimpleCond-Instruments can be used for testing. It is, however, still from theoretical interest to explore the relation between singleton generalized instruments and a conditional instrument. We first define a restricted version of a conditional instrument:

Definition A.4 (Active Conditional Instrument)**.** *Given a DAG $\mathcal{G}$, let c be the path coefficient of the edge $X \rightarrow Y$ and let $\mathcal{G}_c$ be the graph obtained from $\mathcal{G}$ by deleting the edge $X \rightarrow Y$. Variable Z is said to be an* active conditional instrument *relative to $X \rightarrow Y$, if there exists a set $\mathbf{W}$ and a path π from Z to X such that*

(a) $\mathbf{W}$ *does not block the path π in $\mathcal{G}_c$,*
(b) $\mathbf{W}$ *d-separates Z from Y in $\mathcal{G}_c$ and* $\mathbf{W}$ *consists of non-descendants of Y, and*
(c) path π is active in $\mathcal{G}_c$.

Lemma A.5. *Every active conditional instrument relative to $X \rightarrow Y$ is a conditional instrument relative to $X \rightarrow Y$.*

Proof. Since $\mathbf{W}$ does not block the path π between Z and X in $\mathcal{G}_c$ the condition $(Z \not\perp\!\!\!\perp X \mid \mathbf{W})_{\mathcal{G}_c}$ is satisfied. □

There exists a Z which is a conditional instrument relative to $X \rightarrow Y$, but which does not satisfy the conditions of an active conditional instrument:

$$Z \longrightarrow W \longleftarrow X \longrightarrow Y$$

However, every singleton generalized instrumental set is an active conditional instrument:

Lemma A.6. *A set $\{Z\}$ is a generalized instrumental set relative to $\{X\}$ and Y if and only if Z is an active conditional instrument relative to $X \rightarrow Y$.*

Proof. Let the triple $(Z, \mathbf{W}, \pi)$ satisfies the conditions of Def. 3.2. The path π is of the form $\pi' \rightarrow Y$, where π' is a path between Z and X. It is easy to see that $\mathbf{W}$ and π' satisfy the conditions of Def. A.4.

Below we show that the opposite implication is true, too. So, let the set $\mathbf{W}$ and path π satisfy the conditions of Def. A.4. Since $\mathbf{W}$ does not block the path π in $\mathcal{G}_c$, we have $X \notin \mathbf{W}$. We show that the triple $(Z, \mathbf{W}, \pi \rightarrow Y)$ satisfies the conditions of a generalized instrumental set.

Set $\mathbf{W}$ consists of non-descendants of Y due to condition 1 of Def. A.4. Node Z is a non-descendant of Y, otherwise the directed path between Y and Z could not be blocked by $\mathbf{W}$ using only non-descendants. Path π ends with X and does not contain Y, since $\mathbf{W}$ does not block π in $\mathcal{G}_c$, but d-separates Z and Y in $\mathcal{G}_c$ and we have that $Y \notin \mathbf{W}$. So, $\pi \rightarrow Y$ is an unblocked path between Z and Y including the edge $X \rightarrow Y$.

$\mathbf{W}$ d-separates Z from Y in $\mathcal{G}_c = \overline{\mathcal{G}}$ due to condition 2 of Def. A.4. Set $\mathbf{W}$ does not block the path $\pi \rightarrow Y$, since it does not block π in $\mathcal{G}_c$ due to 4. Finally, note that the last condition of Def. 3.2 is always true for a singleton set. □

In (van der Zander et al., 2015) we define a class of *ancestral instruments* and it is not difficult to see that every active conditional instrument is also an ancestral instrument. Thus, from Theorem 4.3 in (van der Zander et al., 2015) we can conclude that there exists an algorithm which for given X, Y, and Z returns a set $\mathbf{W}$ that satisfies the properties of active conditional instrument (Def. A.4) relative to $X \rightarrow Y$, if such a set exists; Otherwise it returns $\perp$. The running time of the algorithm is $\mathcal{O}(nm)$.