Sequential Non-Ancestor Pruning

for Targeted Causal Effect Estimation With an Unknown Graph

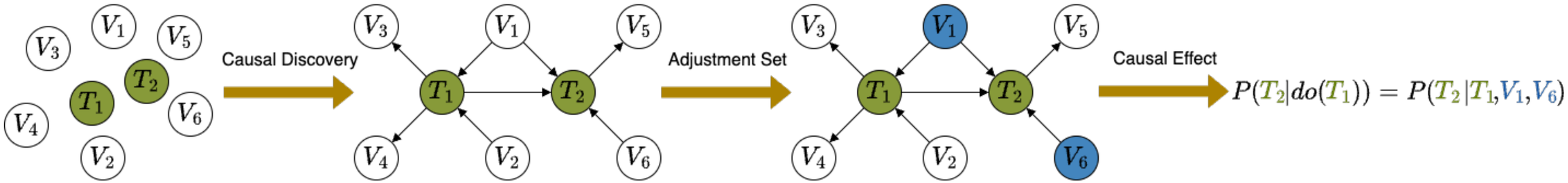
Mátyás Schubert
University of Amsterdam

Tom Claassen
Radboud University Nijmegen

Sara Magliacane
University of Amsterdam

Motivation: targeted causal effect estimation

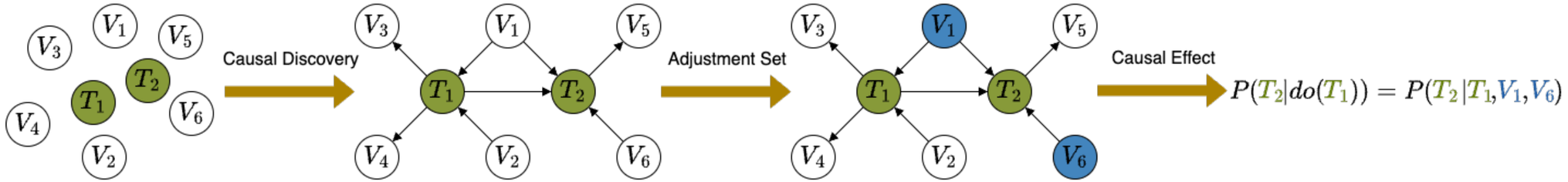
- Goal: Estimate the causal effect between a few **target variables** efficiently, when we have a **large number of variables**
- What do we need to do?
  1. We learn the graph using **causal discovery**
  2. We compute the causal effects with the **optimal adjustment set**



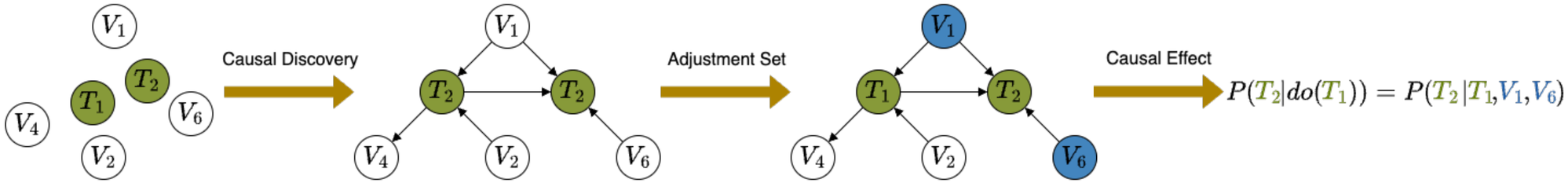
Motivation: targeted causal effect estimation

- Problems:
  - Causal discovery is **not scalable** for large number of variables
  - It might be **wasteful** to learn the complete large graph just for a few variables
- Naive solution: only consider the target variables when using causal discovery
  - Problem: we can “introduce” latent confounding, and some effects might become **unidentifiable**

Motivation: targeted causal effect estimation



- Can we find the causal effects without discovering the whole graph?

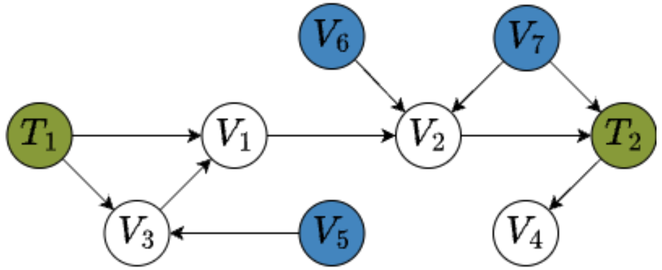


- Our goal is to discover only the **informative parts of the graph**

Targeted Causal Effect Estimation with an Unknown Graph

Definition: Given a joint distribution p over variables $\mathbf{V}$ and **targets** $\mathbf{T} \subseteq \mathbf{V}$, we define targeted causal effect estimation with an unknown graph as the task of estimating in a **computationally and statistically efficient way** the interventional distributions $P(T_i \mid do(T_j))$, for all possible pairs $T_i, T_j \in \mathbf{T}$.

- **Computationally efficient**: number of CI tests and computation time
- **Statistically efficient**: we can identify the optimal adjustment set in terms of asymptotic variance [Henckel et al., 2022]



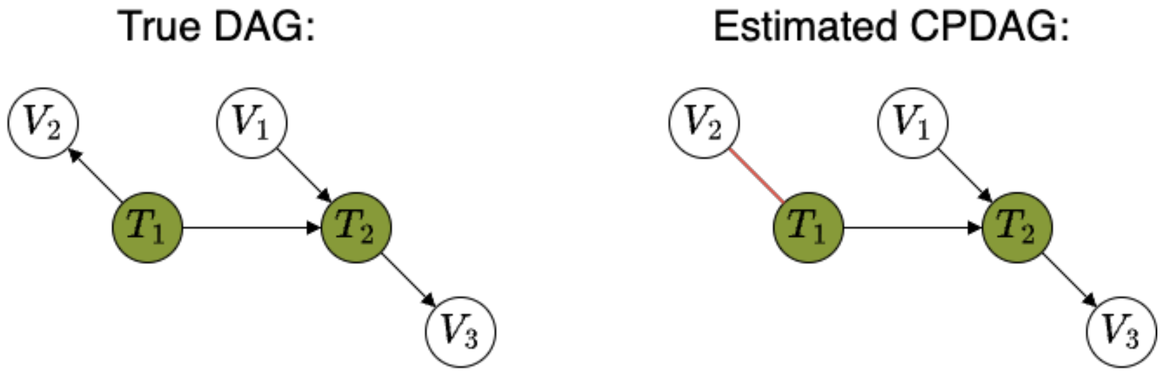
- We assume causal sufficiency: no unobserved confounders or selection bias

Related work

- **Local Causal Discovery:** MB-by-MB [Wang et al., 2014], LDECC [Gupta et al., 2023]
  - Identify the local structure around a target i.e., parents, children etc.
  - Causal effect estimation is limited to using the parents of targets: not statistically efficient
- **Local Discovery by Partitioning** [Maasch et al., 2024]
  - Causal relationship between the two targets is known a priori
  - Can learn groups of nodes that can be used for more efficient adjustment than parents, but it might not recover the optimal adjustment set
- **Confounder Blanket Learner** [Watson and Silva, 2022]
  - Recovers the causal order among multiple targets
  - Assumes all other variables are non-descendants of the targets: we find this
  - Might also not recover the optimal adjustment set

Possible Ancestors Are All You Need

- Guo et al. [2023] show that in causal DAGs, **non-ancestors** of the targets are not part of the efficient adjustment sets of the targets
- In CPDAGs, we can estimate **definite non-ancestors/possible ancestors**
- Definite non-ancestors of $T_2 : \{V_3\}$
- Definite ancestors of $T_2 : \{T_1, V_1\}$
- Possible ancestors of $T_2 : \{T_1, V_1, V_2\}$



- For efficient adjustments in a CPDAG we can then **remove the definite non-ancestors**
- We show that these are also unnecessary for orienting the edges between targets and their efficient adjustment sets

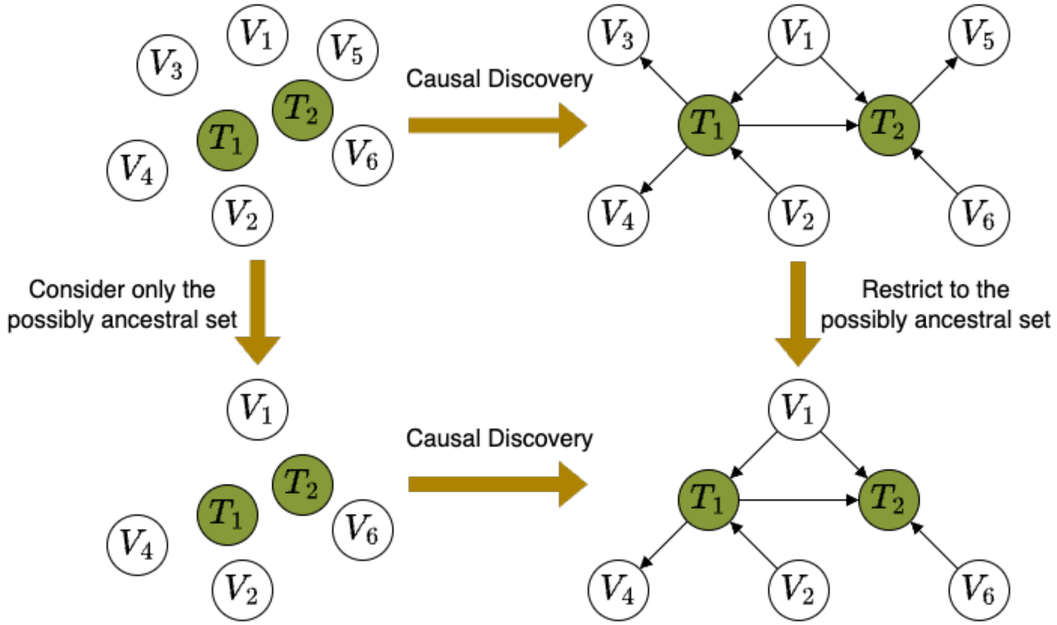
Possible Ancestors Are All You Need

- We might need the whole CPDAG G to find only the possible ancestors of the targets, denoted as $PossAn_G(\mathbf{T})$
- If we had a cheap way to **overestimate** them, that might also be enough to learn a “useful” CPDAG for targeted causal effect estimation
- To this end, we define **possibly ancestral sets** as sets closed under possible ancestral relations

Lemma 3.1: Let G be a full CPDAG over variables $\mathbf{V}$. Let $\mathbf{V}^* \subseteq \mathbf{V}$ be a **possibly ancestral set** of nodes, i.e. $PossAn_G(\mathbf{V}^*) \subseteq \mathbf{V}^*$. Let $G(\mathbf{V})|_{\mathbf{V}^*}$ be the induced subgraph of G over $\mathbf{V}^*$ and let $G(\mathbf{V}^*)$ be the CPDAG over variables $\mathbf{V}^*$. Then we have $G(\mathbf{V})|_{\mathbf{V}^*} = G(\mathbf{V}^*)$.

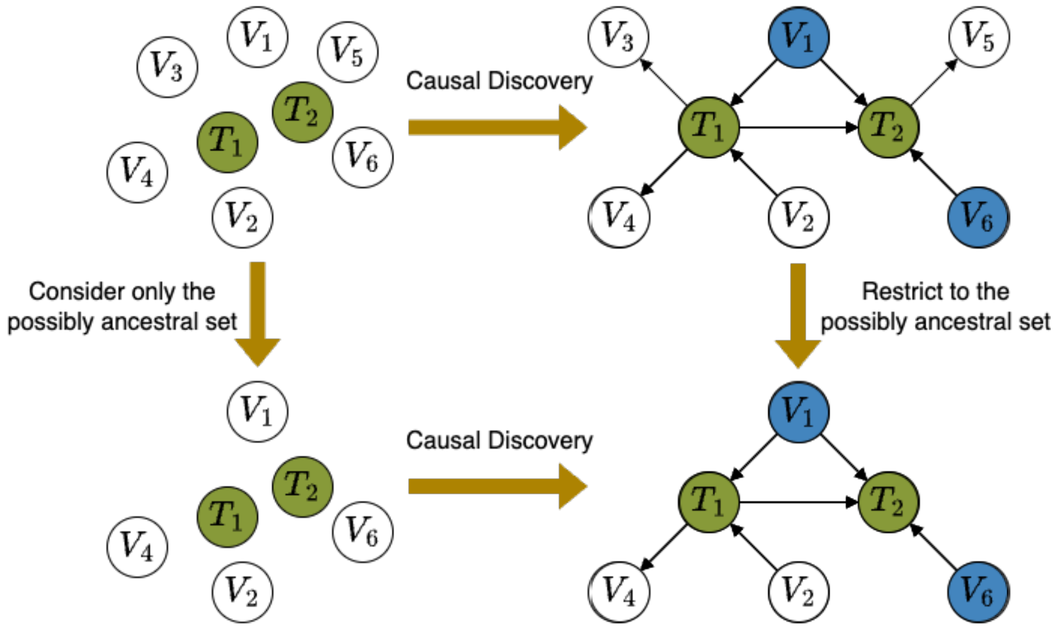
Possible Ancestors Are All You Need

- Running a causal discovery algorithm on a possibly ancestral set that includes the possible ancestors of the targets, solves the task:
  - Computationally efficient if the possibly ancestral set is small
  - As statistically efficient as the full CPDAG



Possible Ancestors Are All You Need

- Running a causal discovery algorithm on a possibly ancestral set that includes the possible ancestors of the targets, solves the task:
  - Computationally efficient if the possibly ancestral set is small
  - As statistically efficient as the full CPDAG



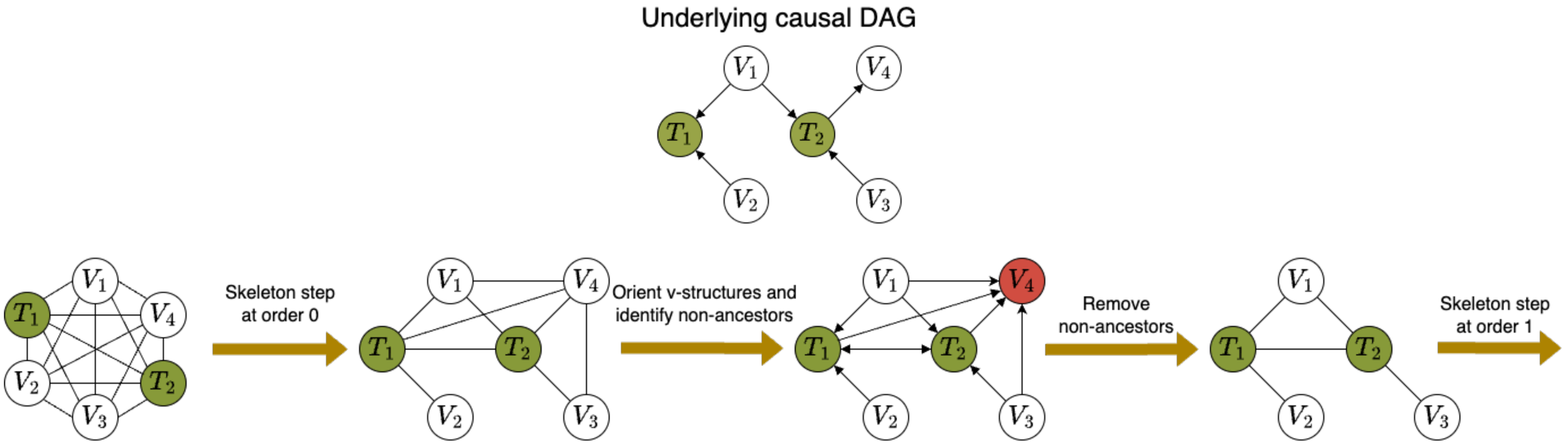
Possible Ancestors Are All You Need

The challenge is identifying a minimal set containing the **possible ancestors of the targets** in a computationally efficient way

In other words, we want to remove as many **definite non-ancestors of the targets** as possible

Sequential Non-Ancestor Pruning (SNAP)

- We introduce SNAP to **iteratively** identify and remove definite non-ancestors
- SNAP adapts the PC-style skeleton search: at every iteration, it orients v-structures, and identifies and removes some definite non-ancestors



Sequential Non-Ancestor Pruning (SNAP)

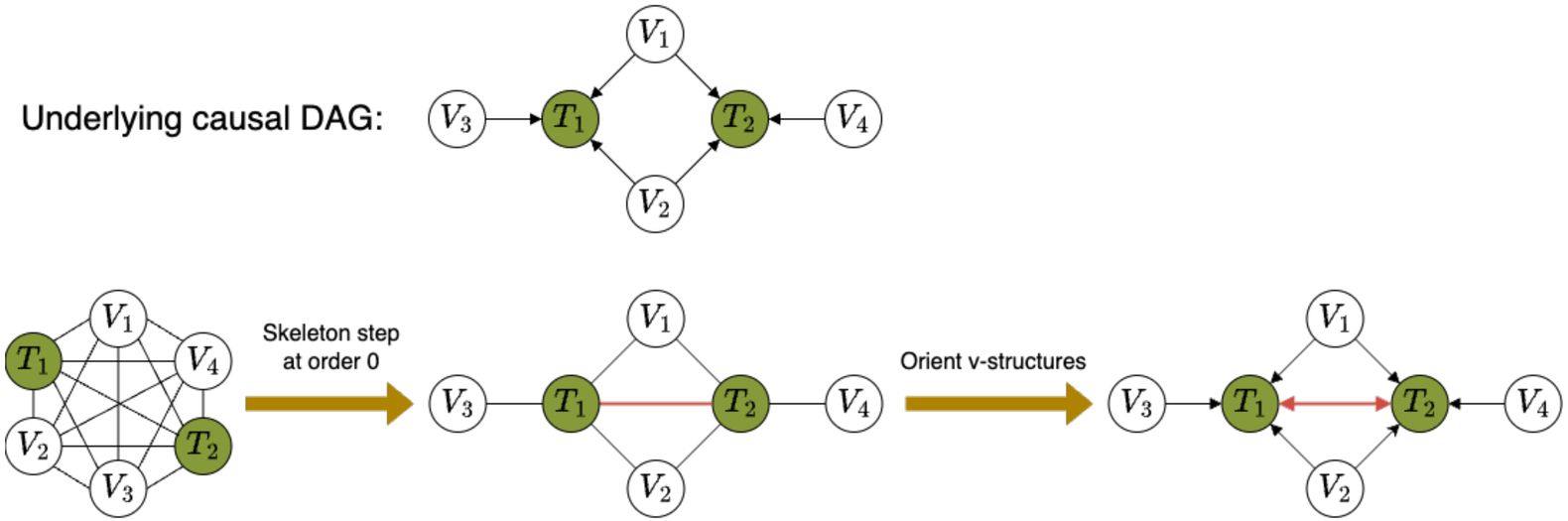
Algorithm 1 Sequential Non-Ancestor Pruning - SNAP(k)

Require: Vertices $\mathbf{V}$, targets $\mathbf{T} \subseteq \mathbf{V}$, Maximum order k

$\hat{\mathbf{V}}^0 \leftarrow \mathbf{V}$, $\hat{U}^{-1} \leftarrow$ fully connected undirected graph over $\mathbf{V}$
for $i \in 0..k$ **do**
  $\hat{U}^i \leftarrow$ induced subgraph of $\hat{U}^{i-1}$ over $\hat{\mathbf{V}}^i$
  for $X \in \hat{\mathbf{V}}^i$, $Y \in Adj_{\hat{U}^i}(X)$ **do** ▷ Learn skeleton step at order i
    for $\mathbf{S} \subseteq Adj_{\hat{U}^i}(X) \setminus \{Y\}$ s.t. $|\mathbf{S}| = i$ **do**
      if $X \perp\!\!\!\perp Y | \mathbf{S}$ **then**
        Delete the edge $X - Y$ from $\hat{U}^i$
        $sepset(X, Y) \leftarrow sepset(Y, X) \leftarrow \mathbf{S}$
        break
  if $i < 2$ **then** ▷ Orient v-structures
    $\hat{G}^i \leftarrow$ OrientVstructPC($\hat{U}^i$, *sepset*) (Alg. 3)
  else
    $\hat{G}^i$, *sepset* $\leftarrow$ OrientVstructRFCI($\hat{U}^i$, *sepset*) (Alg. 4)
    $\hat{U}^i \leftarrow$ skeleton of $\hat{G}^i$
  $\hat{\mathbf{V}}^{i+1} \leftarrow$ all $V \in \hat{\mathbf{V}}^i$ with a possibly directed path to any $T \in \mathbf{T}$ in $\hat{G}^i$ ▷ Prune non-ancestors
$\hat{G}^k \leftarrow$ induced subgraph of $\hat{G}^k$ over $\hat{\mathbf{V}}^{k+1}$
return $\hat{\mathbf{V}}^{k+1}, \hat{G}^k$

Sequential Non-Ancestor Pruning (SNAP)

- Standard v-structure orientations indicate non-ancestry, when **all CI tests** restricted to a low order are available [Wienöbst and Liskiewicz, 2020]



Example from [Wienöbst and Liskiewicz, 2020]

- This might not hold for PC skeleton search, which performs a **subset of CI tests**
- We show that using RFCI-style v-structure orientations instead makes it hold

Sequential Non-Ancestor Pruning (SNAP)

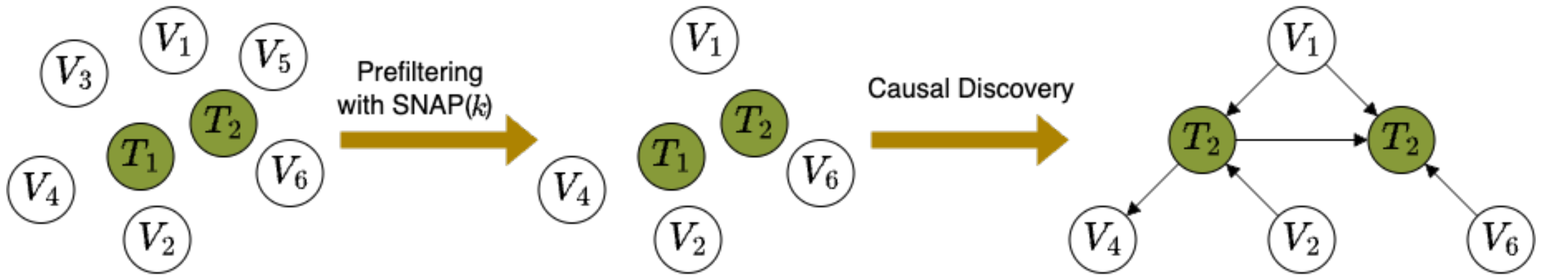
- Reminder: $\mathbf{V}^* \subseteq \mathbf{V}$ is a possibly ancestral set of nodes, i.e. $PossAn_G(\mathbf{V}^*) \subseteq \mathbf{V}^*$
- We show that at every iteration, the remaining variables are a possibly ancestral set containing the targets, and thus their possible ancestors

Theorem 3.1: Given oracle conditional independence tests, at each step $i = 0, \ldots, k$ of SNAP(k), $\hat{\mathbf{V}}^{i+1}$ contains $PossAn(\mathbf{T})$. Additionally, $\hat{\mathbf{V}}^{i+1}$ is a possibly ancestral set, i.e. $PossAn_G(\hat{\mathbf{V}}^i) \subseteq \hat{\mathbf{V}}^{i+1}$.

- As we have shown before, this set is statistically efficient for the targeted causal effect estimation

Sequential Non-Ancestor Pruning (SNAP)

- **Pre-filtering** with SNAP(k): stop at any iteration k and run standard causal discovery algorithms on the remaining variables



- Considering fewer and fewer variables, as the size of the conditioning set increases, leads to **significantly fewer higher order tests** in practice

Sequential Non-Ancestor Pruning (SNAP)

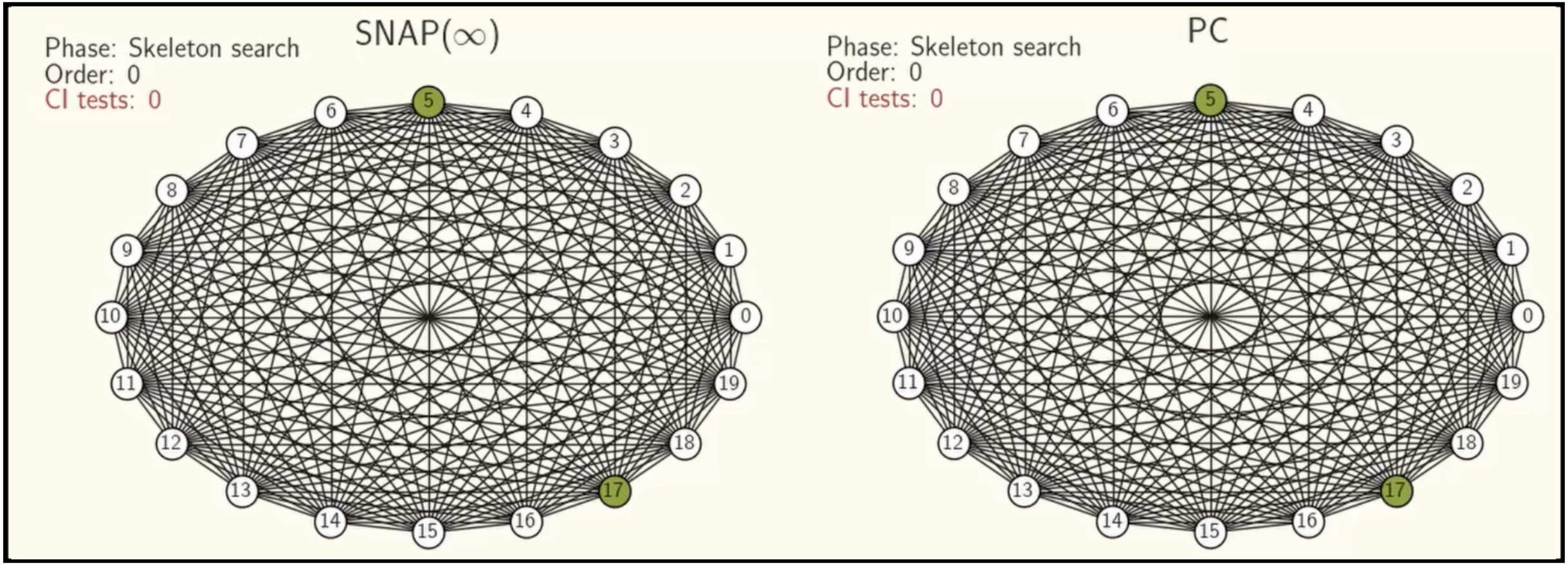
- SNAP(∞) extends SNAP(k) to a **stand-alone causal discovery algorithm**
- SNAP(∞) runs SNAP(k) *until completion*, i.e. $k = |\mathbf{V}| - 2$, and then runs Meek's orientation rules to identify definite non-ancestors one last time

Theorem 3.2: Given oracle conditional independence tests, let $\hat{G}$ be the output graph of SNAP(∞) for targets $\mathbf{T}$. Then, SNAP(∞) returns $\hat{\mathbf{V}} = PossAn(\mathbf{T})$. Additionally, SNAP($\infty$) is **sound and complete** over the possible ancestors $\mathbf{T}$, i.e.

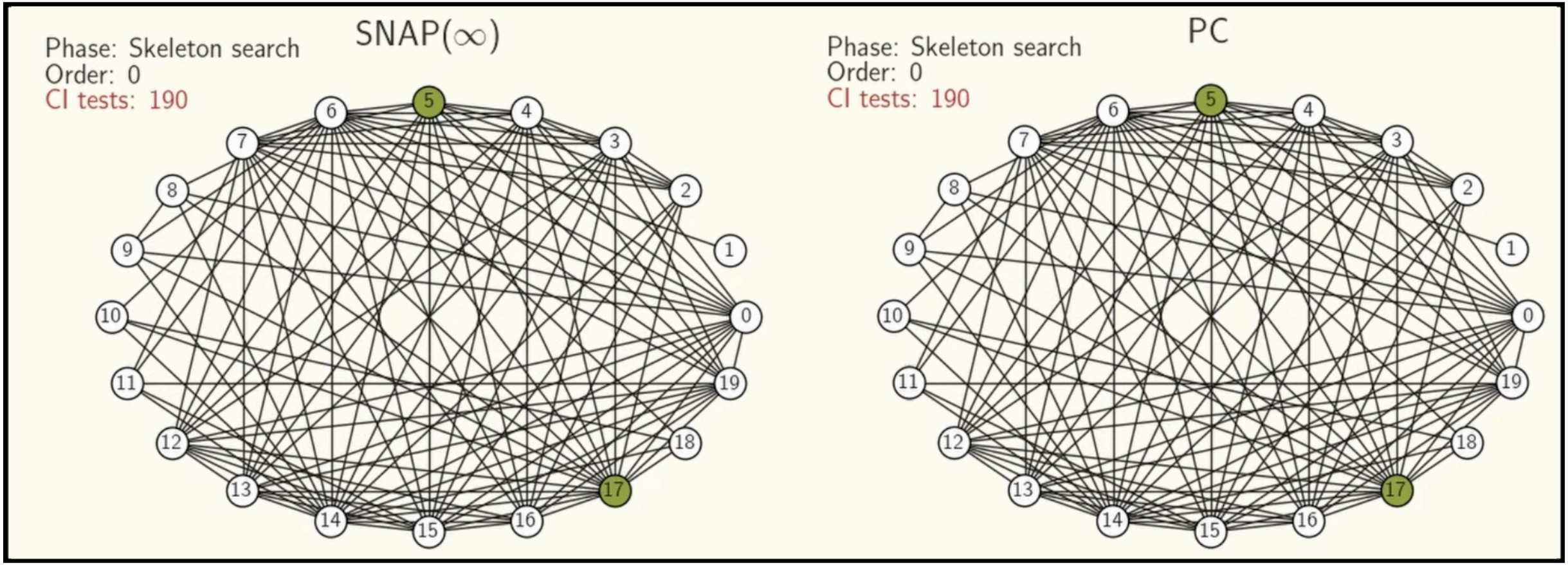
$$\hat{G} = G|_{PossAn(\mathbf{T})}$$

- We show that worst-case complexity of SNAP(∞) matches PC if the maximum degree of the graph is ≥ 2

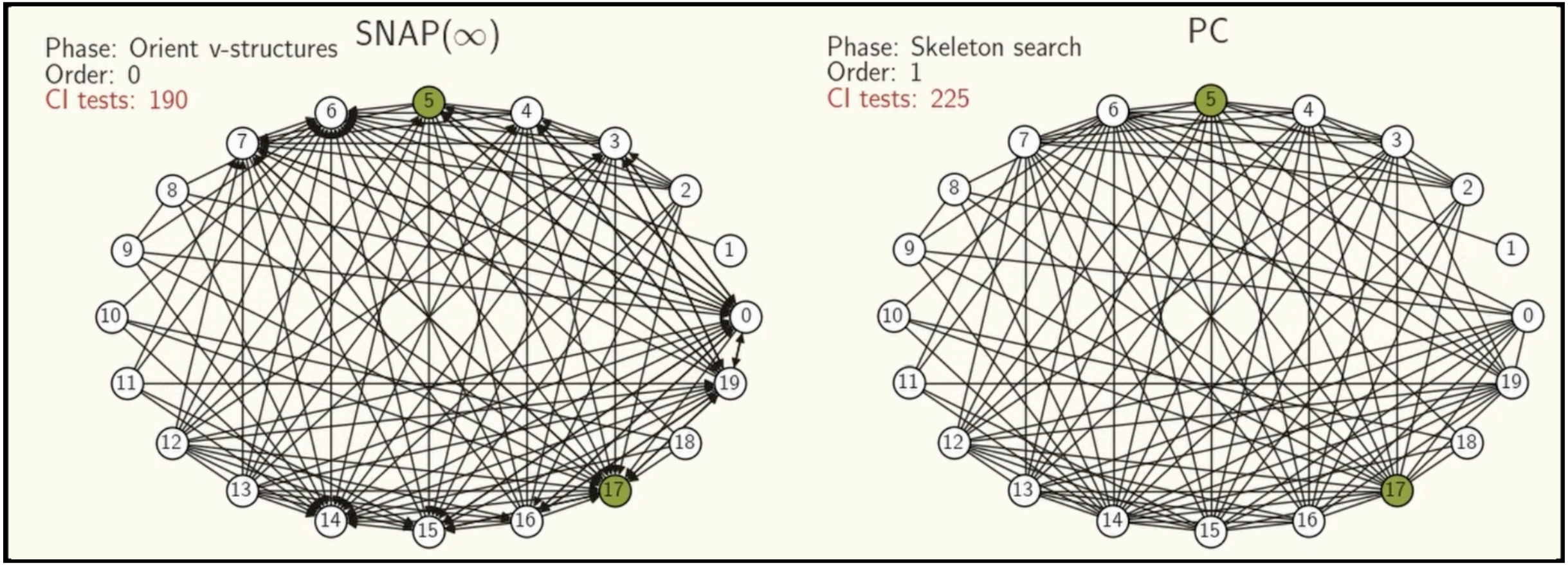
Sequential Non-Ancestor Pruning (SNAP)



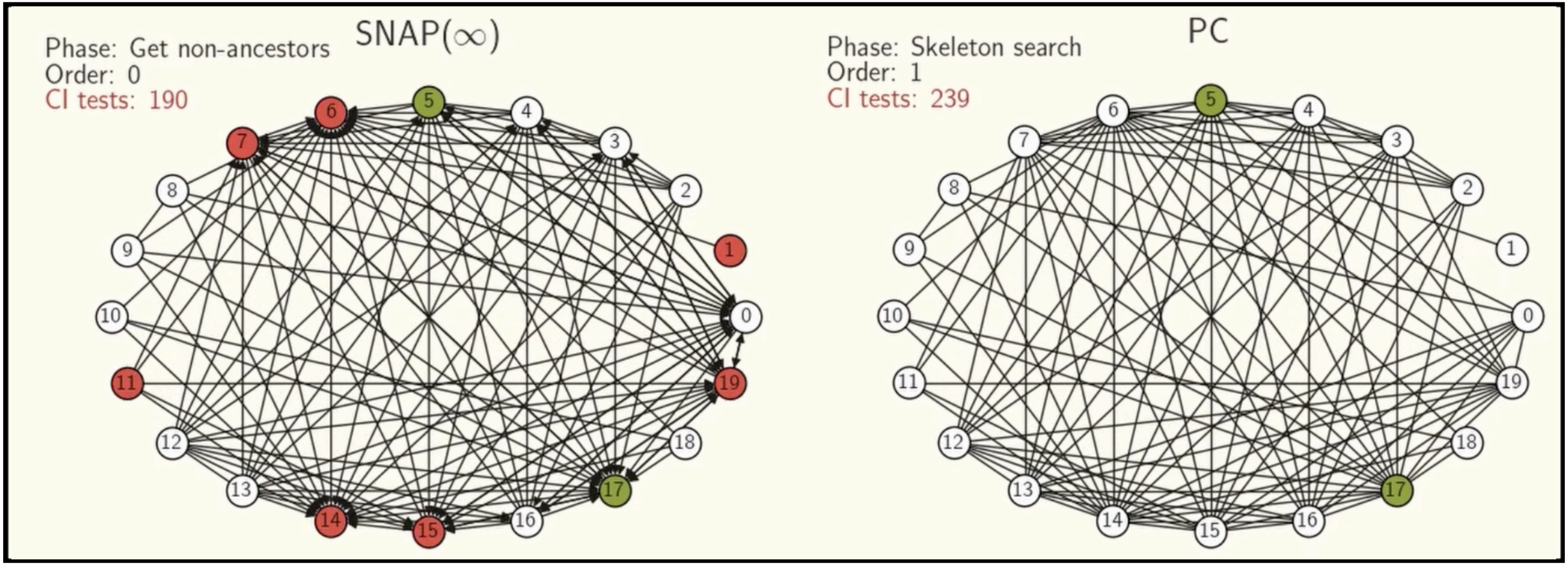
Sequential Non-Ancestor Pruning (SNAP)



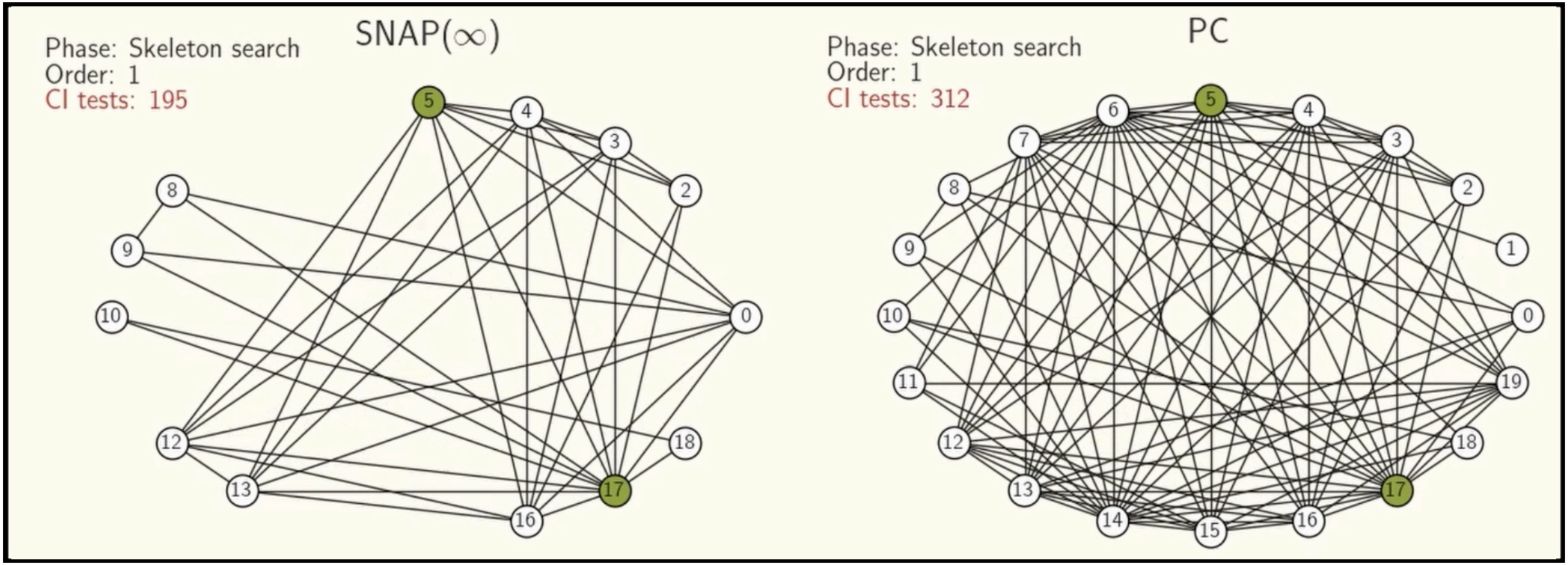
Sequential Non-Ancestor Pruning (SNAP)



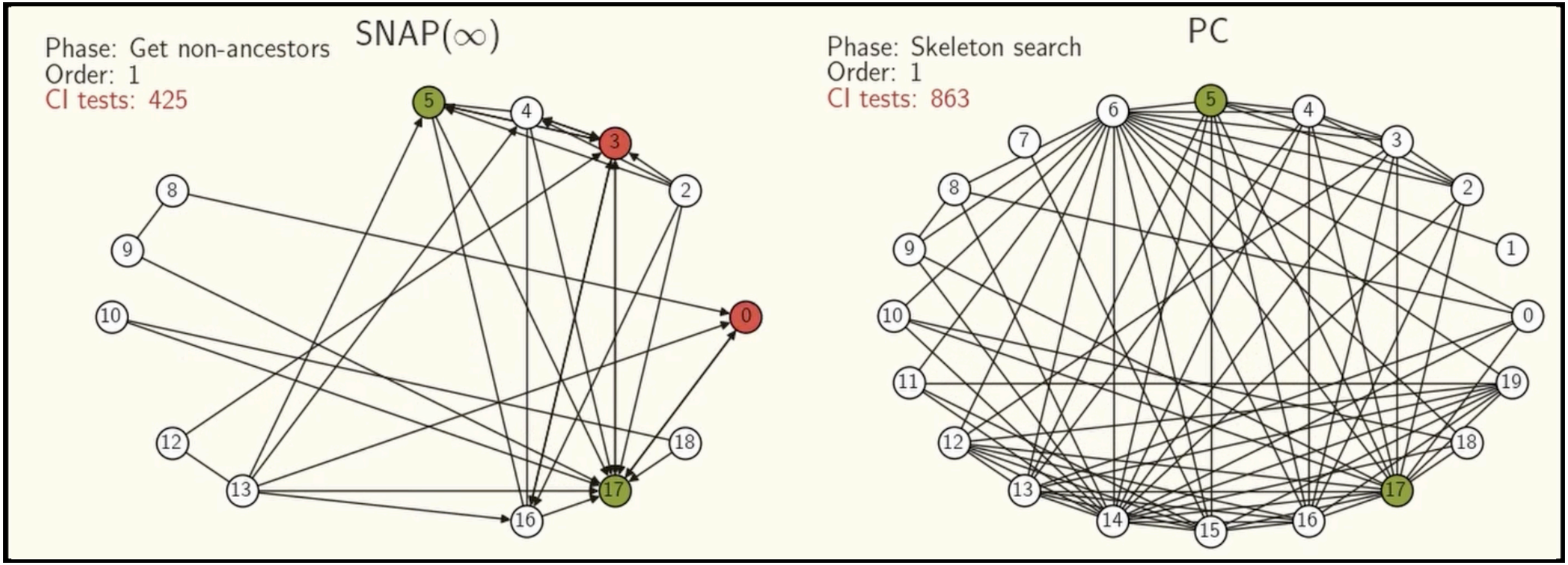
Sequential Non-Ancestor Pruning (SNAP)



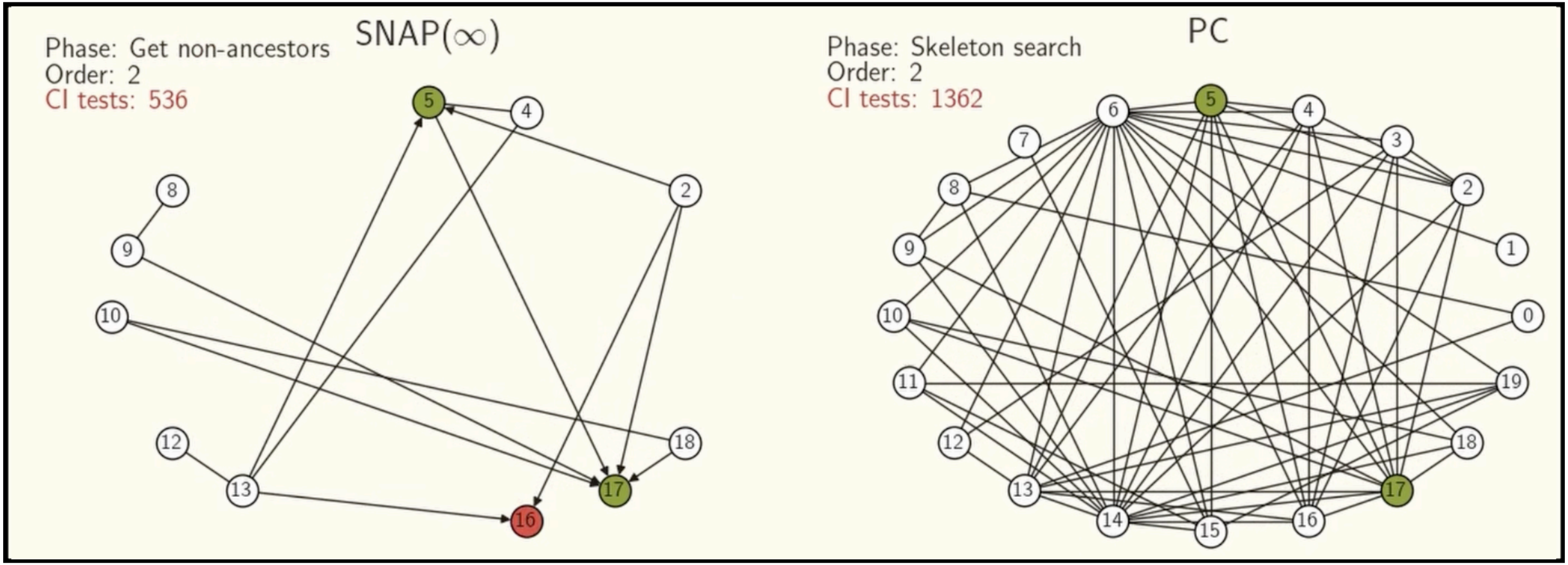
Sequential Non-Ancestor Pruning (SNAP)



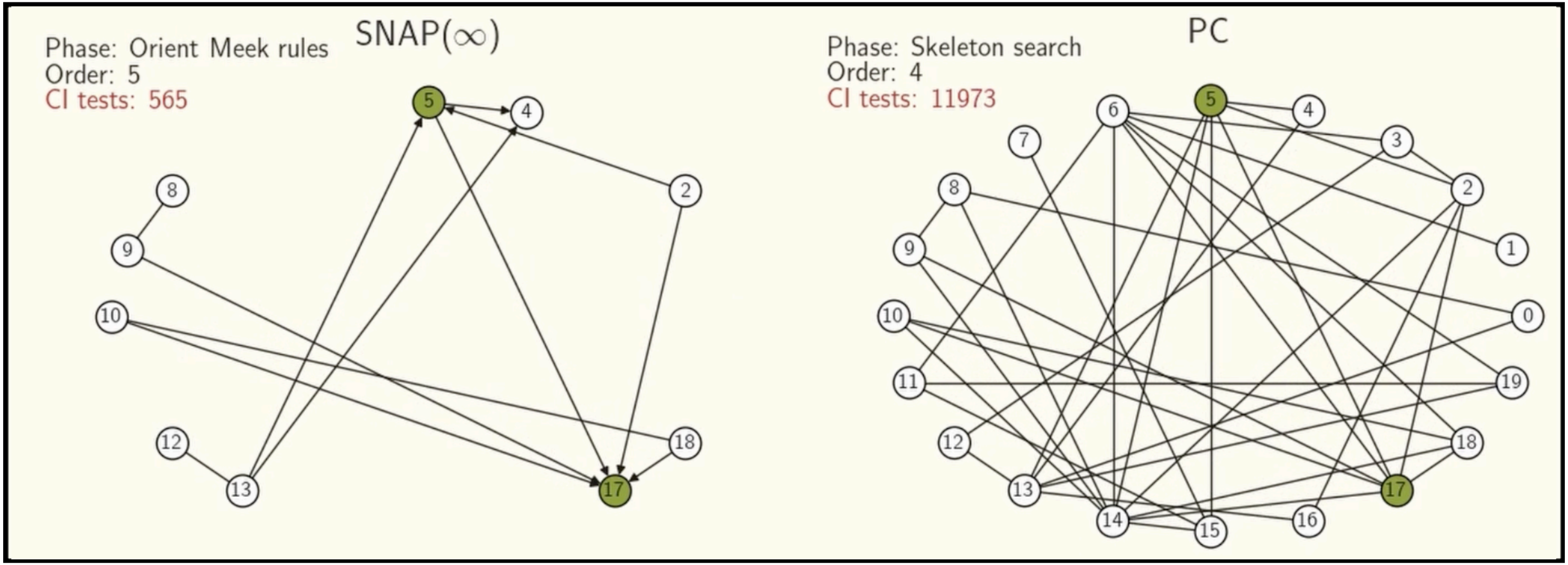
Sequential Non-Ancestor Pruning (SNAP)



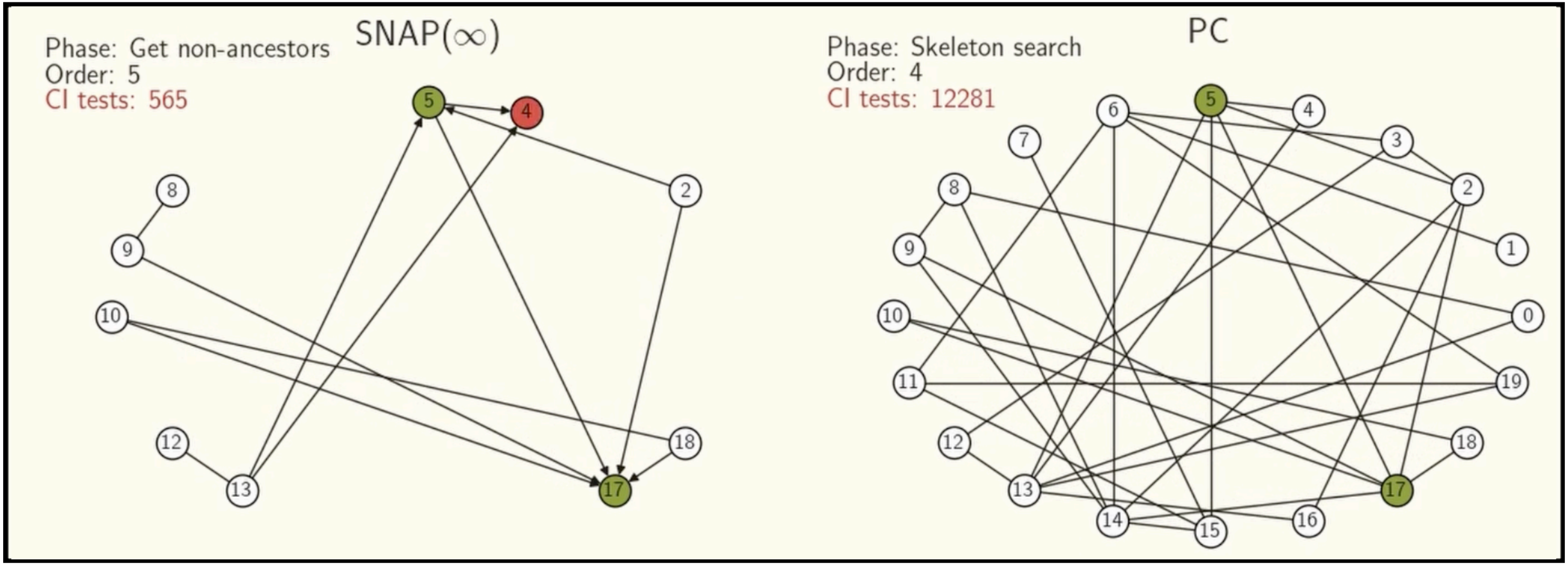
Sequential Non-Ancestor Pruning (SNAP)



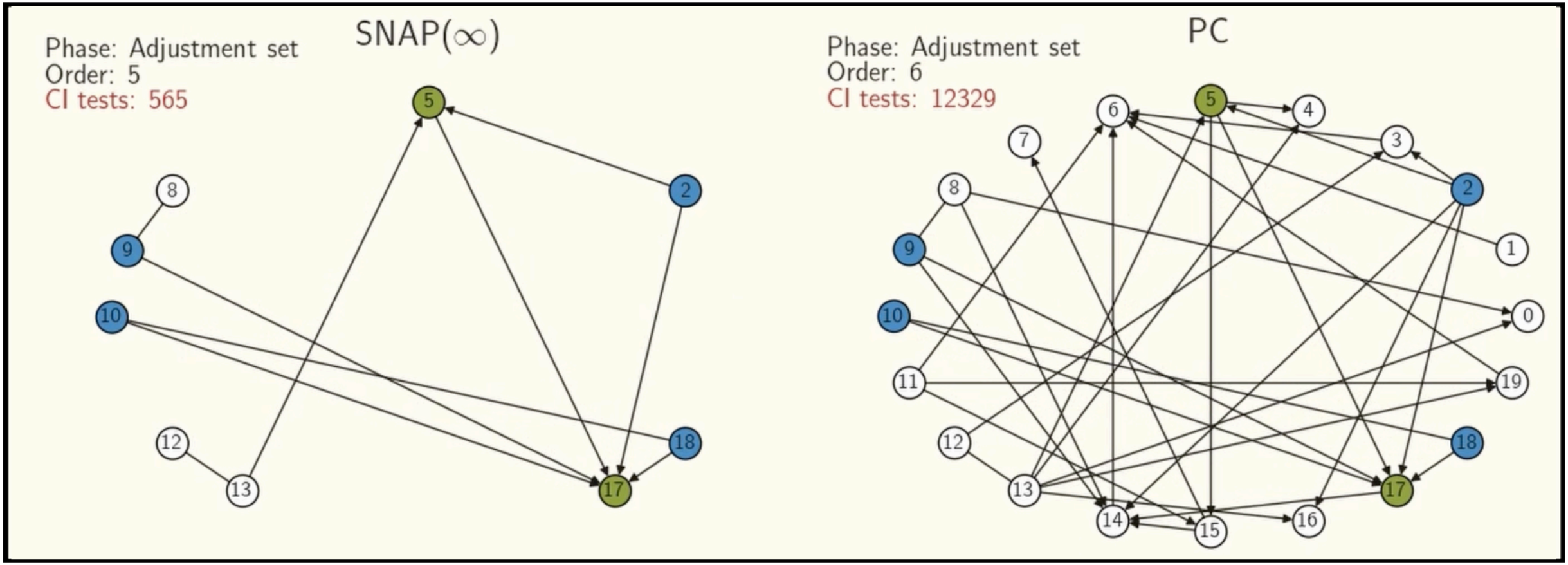
Sequential Non-Ancestor Pruning (SNAP)



Sequential Non-Ancestor Pruning (SNAP)

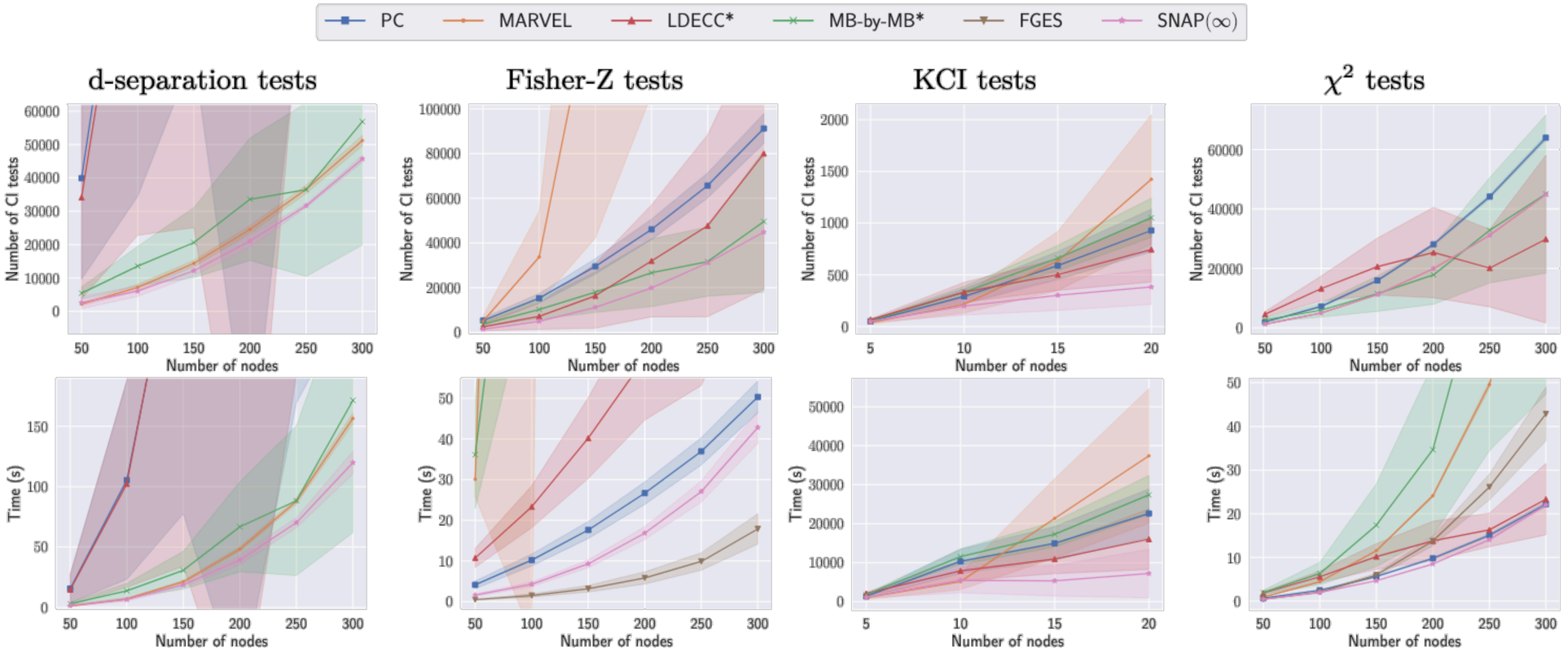


Sequential Non-Ancestor Pruning (SNAP)



Experiments (synthetic graphs)

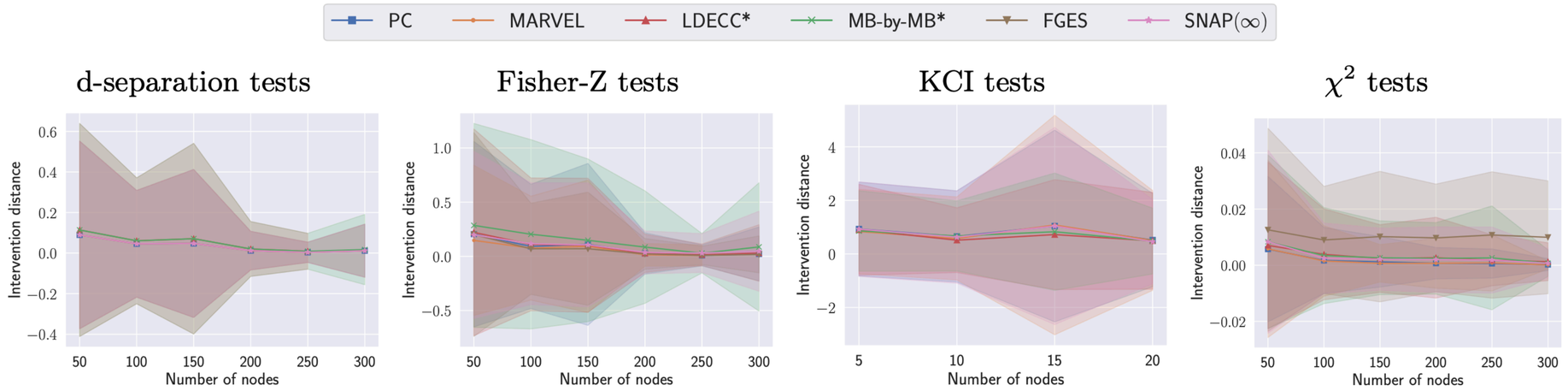
- SNAP(∞) is **one of the best methods across domains** in terms of number of CI tests and computation time, with a comparable intervention distance



4 targets, expected degree of 3, maximum degree of 10 and 1000 data points

Experiments (synthetic graphs)

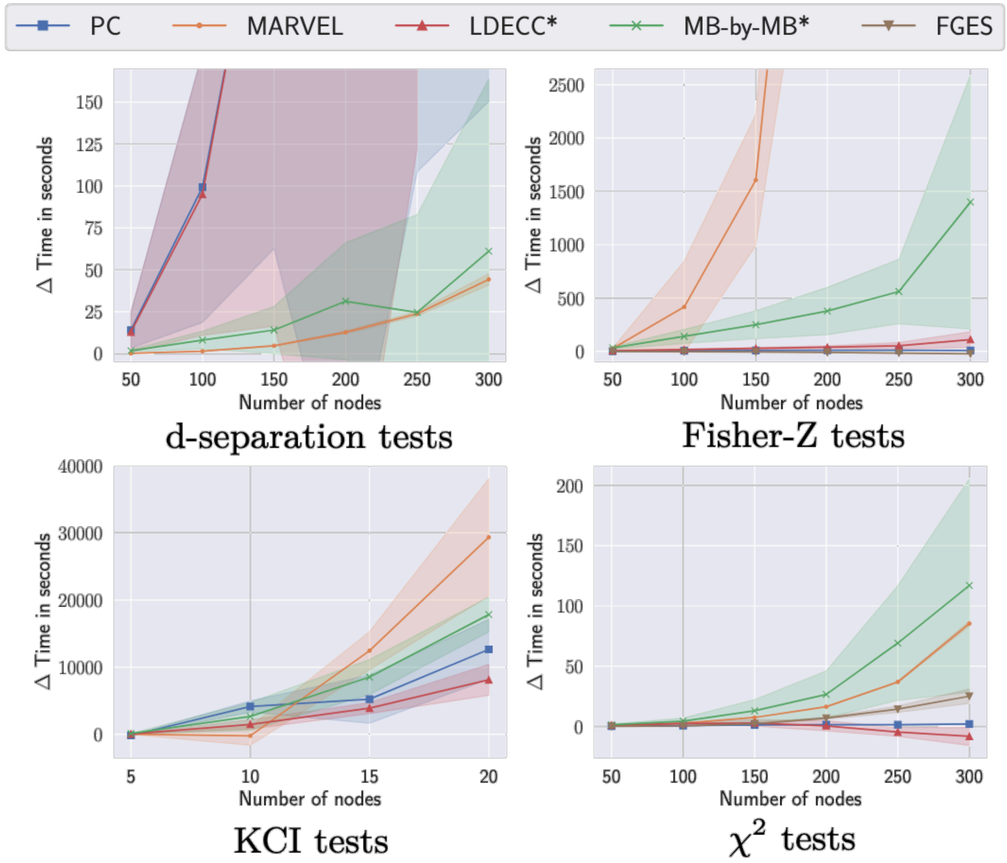
- SNAP(∞) is **one of the best methods across domains** in terms of number of CI tests and computation time, with a comparable intervention distance



4 targets, expected degree of 3, maximum degree of 10 and 1000 data points

Experiments (synthetic graphs)

- Pre-filtering with SNAP(0) improves the baselines in most settings



4 targets, expected degree of 3, maximum degree of 10 and 1000 data points

Experiments (synthetic graphs)

- Pre-filtering with SNAP($k > 0$) further reduces CI tests on denser graphs

d-separation CI tests, 50 nodes and expected degree of 5

Experiments (MAGIC-NIAB)

- Pre-filtering with SNAP(0) consistently improves most baselines

	CI tests	Time
PC	12807 (±2086)	79.3 (±24.7)
PC-SNAP(0)	**955 (±10)**	0.5 (±0.1)
MARVEL	8873 (±3056)	27.3 (±9.4)
MARVEL-SNAP(0)	960 (±5)	0.6 (±0.1)
LDECC*	18142 (±2608)	19.2 (±4.0)
LDECC*-SNAP(0)	981 (±23)	0.8 (±0.1)
MB-by-MB*	11464 (±1995)	25.7 (±4.7)
MB-by-MB*-SNAP(0)	972 (±17)	0.7 (±0.2)
FGES	-	0.7 (±0.1)
FGES-SNAP(0)	-	**0.4 (±0.1)**
SNAP(∞)	**955 (±10)**	0.6 (±0.2)

4 *identifiable* targets and 1000 linear Gaussian data points

Conclusions and future work

- We have shown how to estimate the causal effects between a few targets variables in large graph in a **computationally** and **statistically efficient way**
- The SNAP framework discovers the relevant portion of the CPDAG for estimating these effects
  - SNAP(k) can be used as a pre-processing step for other discovery algorithms, making them faster
  - SNAP(∞) is a stand-alone sound and complete discovery algorithm
- Future work:
  - Handle unobserved confounders
  - Add background knowledge, e.g. coming from an LLM, including when it's imperfect

Questions?

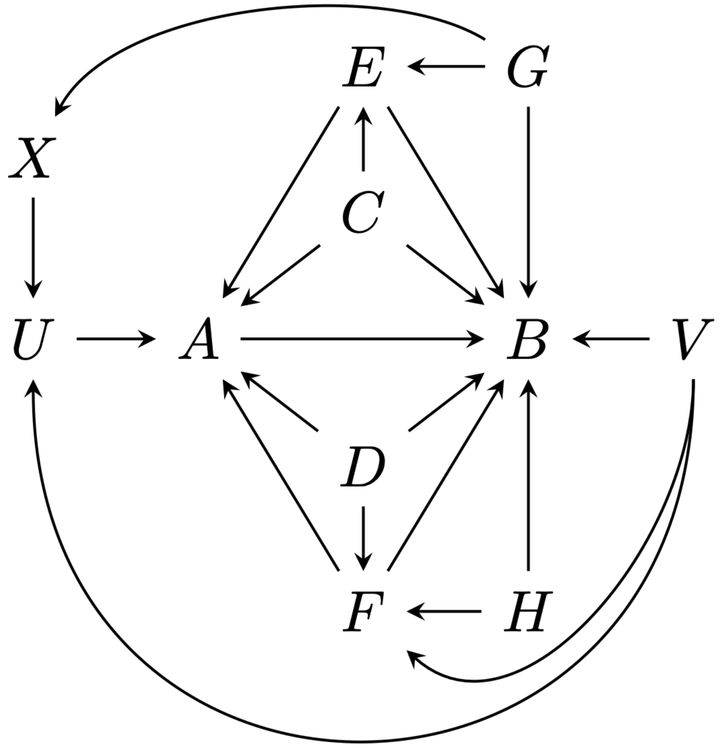
Paper: arxiv.org/abs/2502.07857
Project page: matyasch.github.io/snap

References

- Leonard Henckel, Emilija Perković, and Marloes H Maathuis. Graphical criteria for efficient total effect estimation via adjustment in causal linear models. Journal of the Royal Statistical Society Series B: Statistical Methodology, 2022.
- Changzhang Wang, You Zhou, Qiang Zhao, and Zhi Geng. Discovering and orienting the edges connected to a target variable in a dag via a sequential local learning approach. *Computational statistics & data analysis*, 2014.
- Shantanu Gupta, David Childers, and Zachary Chase Lipton. Local causal discovery for estimating causal effects. CLeaR, 2023.
- Jacqueline R. M. A. Maasch, Weishen Pan, Shantanu Gupta, Volodymyr Kuleshov, Kyra Gan, and Fei Wang. Local discovery by partitioning: Polynomial-time causal discovery around exposure-outcome pairs. UAI, 2024
- David S Watson and Ricardo Silva. Causal discovery under a confounder blanket. UAI, 2022
- F Richard Guo, Emilija Perković, and Andrea Rotnitzky. Variable elimination, graph reduction and the efficient g-formula. Biometrika, 2023.
- Marcel Wienöbst and Maciej Liskiewicz. Recovering causal structures from low-order conditional independencies. AAAI, 2020

Where standard v-structure orientations fail

- At order 3, we get $A \leftrightarrow B$, even though $A \rightarrow B$ in the graph



RFCI v-structure orientation rules

Algorithm 4 OrientVstructRFCI: Orienting v-structures in the RFCI algorithm [Colombo et al., 2012]

Require: Skeleton $\hat{U}$, separating sets *sepset*

$M \leftarrow \{(X, Z, Y)$ such that $X - Z - Y$ in $\hat{\mathbf{E}}$ and $X \notin Adj_{\hat{U}}(Y)\}$
$L \leftarrow \{\}$
repeat
    $X, Z, Y \leftarrow$ choose an unshielded triple from M
    if $Z \notin sepset(X, Y)$ **then**
        if $X \not\perp\!\!\!\perp Z | sepset(X, Y)$ and $Z \not\perp\!\!\!\perp Y | sepset(X, Y)$ **then**
            $L \leftarrow L \cup \{(X, Y, Z)\}$ ▷ Add to legitimate v-structures
        else
            for $V \in \{X, Y\}$ **do**
                if $V \perp\!\!\!\perp Z | sepset(X, Y)$ **then**
                    $\mathbf{S} \leftarrow sepset(X, Y)$ ▷ Find minimal separating set
                    done $\leftarrow$ False
                    while not done **do**
                        done $\leftarrow$ True
                        for $S \in \mathbf{S}$ **do**
                            if $V \perp\!\!\!\perp Z | \mathbf{S} \setminus \{S\}$ **then**
                                $\mathbf{S} \leftarrow \mathbf{S} \setminus \{S\}$
                                done $\leftarrow$ False
                                break
                    $sepset(V, Z) \leftarrow sepset(Z, V) \leftarrow \mathbf{S}$
                    $M \leftarrow M \cup$ all triangles $(\min(V, Z), \cdot, \max(V, Z))$ in $\hat{U}$ ▷ Update new unshielded triples.
                    $M \leftarrow M \setminus$ all triples in M of the form $(V, Z, \cdot), (Z, V, \cdot), (\cdot, V, Z)$ and $(\cdot, Z, V)$
                    $L \leftarrow L \setminus$ all triples in L of the form $(V, Z, \cdot), (Z, V, \cdot), (\cdot, V, Z)$ and $(\cdot, Z, V)$
                    Delete the edge $V - Z$ from $\hat{U}$
until M is empty
$\hat{G} \leftarrow \hat{U}$
for $(X, Z, Y) \in L$ **do**
    Orient $X \ast\!\!\ast Z \ast\!\!\ast Y$ as $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast Y$ in $\hat{G}$
return $\hat{G}$, *sepset*

Computational complexity

We did not find a formal complexity analysis of the RFCI orientation rules in the literature, so we show in Lemma 3.2 that its worst-case complexity in terms of CI tests is at most $\mathcal{O}(|\mathbf{V}|^4)$.

Lemma 3.2. *The worst-case complexity of the RFCI orientation rules (Algorithm 4) in terms of CI tests performed is at most $\mathcal{O}(|\mathbf{V}|^4)$ CI tests, where $|\mathbf{V}|$ is the number of nodes.*

From Lemma 3.2 and the classical result on PC-style skeleton search by Spirtes et al. [2000], it follows that the worst-case computational complexity of SNAP(∞) is at most

$$\mathcal{O}(|\mathbf{V}|^{d_{\max}+2} + |\mathbf{V}|^4),$$

Corollary 3.2. *For graphs with maximum degree $d_{max} \geq 2$, the worst-case computational complexity of SNAP(∞) in terms of CI tests is $\mathcal{O}(|\mathbf{V}|^{d_{\max}+2})$, which matches the complexity of PC.*

Example where SNAP is slower

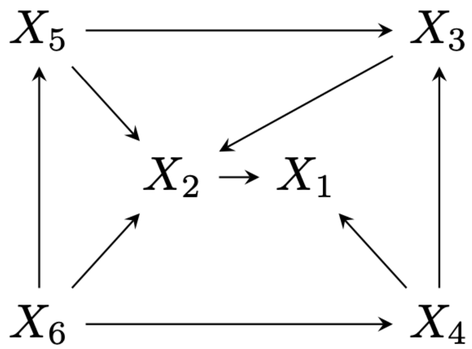


Figure 6: An example graph with targets $\mathbf{T} = \{X_1, X_2\}$ on which SNAP(∞) performs more CI tests than PC.

Rough approximation of expected ancestors

We get the expected rank of the highest ranking target by taking the expectation over all possible highest ranks $i = |\mathbf{T}|..|\mathbf{V}|$ as follows:

$$\hat{M} = \frac{1}{\binom{|\mathbf{V}|}{|\mathbf{T}|}} \sum_{i=|\mathbf{T}|}^{|\mathbf{V}|} i \binom{i-1}{|\mathbf{T}|-1} \tag{1}$$

We can now overestimate the size of the possible ancestors of $\mathbf{T}$ by simply considering it to be M.

Nodes	50	100	150	200	250	300
$\bar{M}$	19.64($\pm$7.16)	23.95($\pm$10.85)	26.88($\pm$13.60)	29.49($\pm$18.66)	28.18($\pm$15.50)	33.78($\pm$17.97)
$\hat{M}$	40.8	80.8	120.8	160.8	200.8	240.8

Table 2: Estimates for the expected number of possible ancestors empirically ($\bar{M}$) over 100 seeds with various numbers of nodes, $n_{\mathbf{T}} = 4, \overline{d} = 3$ and $d_{\max} = 10$ in the first row, and by using the Equation 1 in the second row ($\hat{M}$).