Press ECCS to Doubt (Your Causal Graph)

Markos Markakis, Ziyu Zhang, Rana Shahout, Trinity Gao, Chunwei Liu, Ibrahim Sabek, Michael Cafarella

MIT CSAIL

June 14, 2024

Causal reasoning has seen wide applicability

Causal reasoning helps scientists pose, discuss and test hypotheses in a principled manner

Several non-CS domains make heavy use of it

Economics and social sciences, e.g. for policy making

Biology and medicine to design and evaluate treatments

Etc.

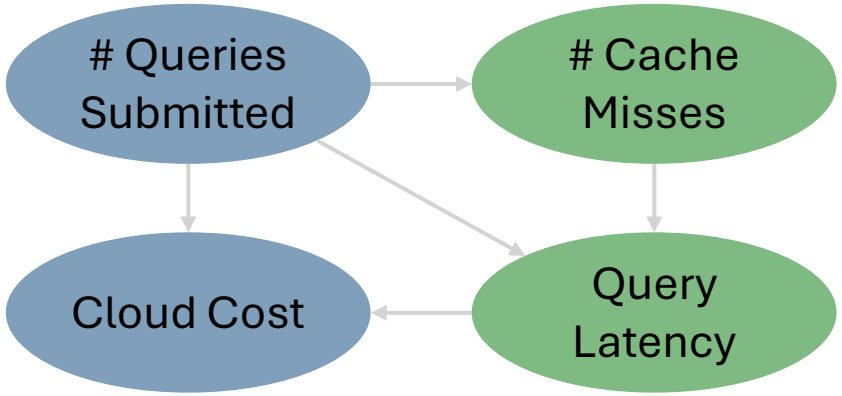
In recent years, many CS domains also leverage these concepts

ML: Causal representation learning, causal reinforcement learning etc.

Systems: network policy selection, large system debugging etc.

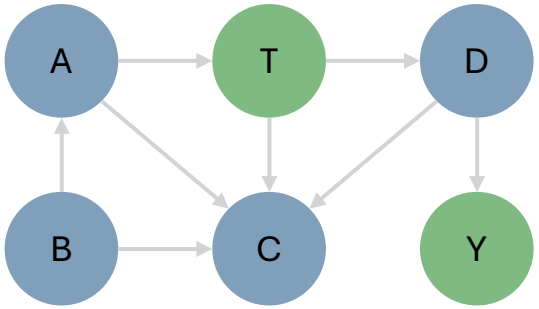
Leveraging causality helps avoid biases

- Causal relationships are often expressed as a **causal graph**:
  - A node is a variable
  - A directed edge is a causal relationship.
- Our goal is to compute correct **Average Treatment Effects**:
  - Involves controlling for certain variables…
  - …but NOT controlling for some others.
  - We call this the **adjustment set**.
- Graphical criteria exist to find a correct adjustment set easily.

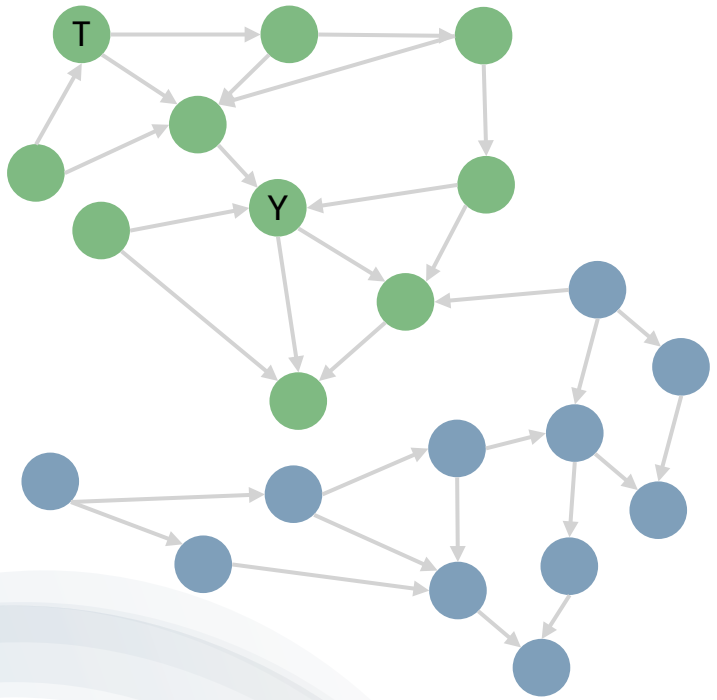


High-quality causal graphs are hard to find

- Two ways to obtain causal graphs:
  - Manual curation
  - Automatic causal discovery from data
- In large problem instances, manual curation is impractical.
- But discovered graphs may contain errors!
  - Causal discovery algorithms require assumptions (about missing variables, FDs etc.) that may not be satisfied in the data.
  - Graph should still be **verified manually.**



Can we only verify what matters?



- Causal graphs are not created in a void – there is a specific ATE question that the user is trying to answer.
- Only some parts of the graph can affect the *adjustment set*, which determines answer to each given question.
- Instead of the user having to verify *everything*, we would like to only ask them to verify the *parts that matter for their question*.
- We do this with **ECCS: E**xposing **C**ritical **C**ausal **S**tructures.

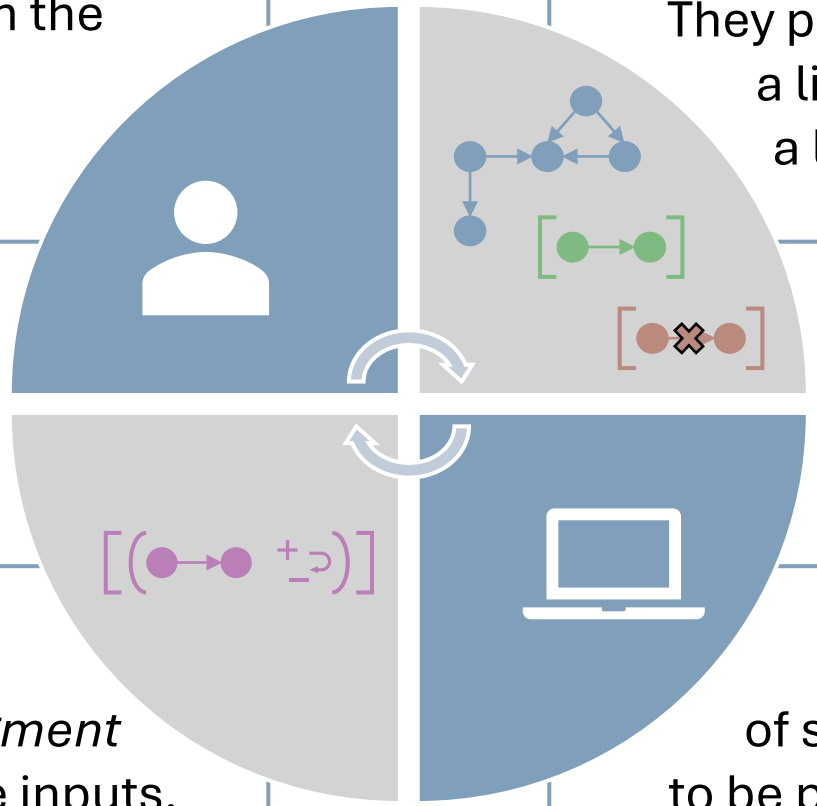
We include a human in the loop

The user fixes:

- a dataset D
- a treatment variable T
- an outcome variable Y

The user interacts with the system in a series of interaction rounds.

They provide a causal graph, a list of FIXED edges and a list of BANNED edges.



ECCS computes a set of suggested graph edits, to be presented one-by-one.

For each suggestion, the user *makes a judgment* and revises their three inputs.

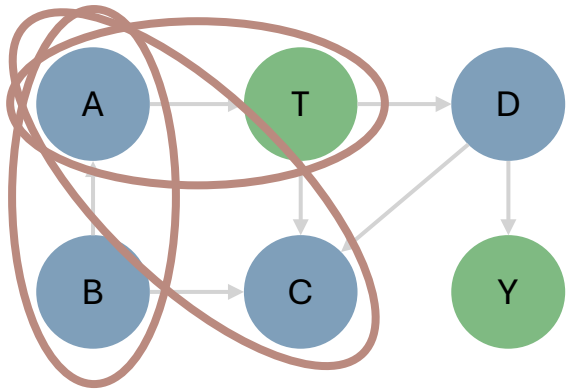
ECCS should make “good” suggestions

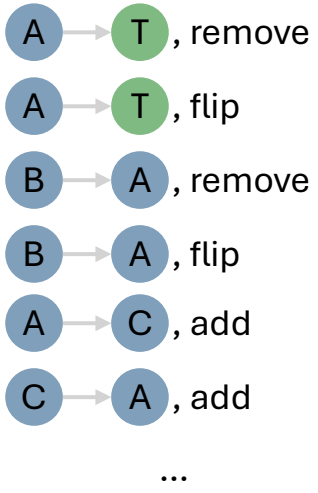
- Our goal is to **converge to the correct ATE** while minimizing user interactions.
  - Put another way, over a series of judgements, we want **ATE error to decrease fast**.
- Each interaction should be actually interactive – i.e. we want **low latency**.
- Problem: We **don’t know the correct ATE** to which we are trying to converge, so we cannot optimize directly.
- We implement 3 strategies to address this.

ECCS computes a set of suggested graph edits, to be presented one-by-one.

Strategy 1: SingleEdit

Strategy 1: Edit the “most impactful edge”

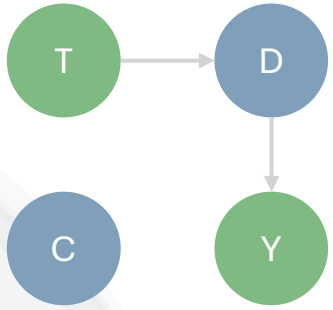


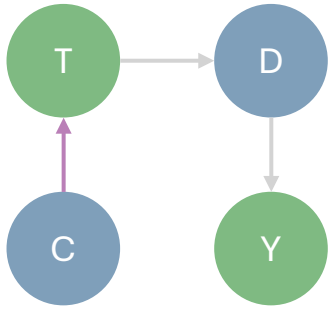


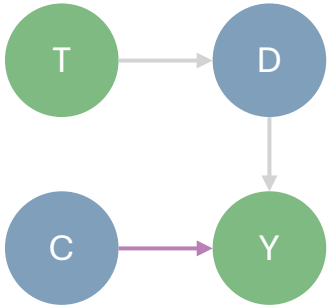
- Find all possible single-edge edits from the current graph:
  - For a non-FIXED edge, remove it or flip it.
  - For non-BANNED edge, add it in.
- Suggest the edit that maximizes the impact on the ATE of interest:
  - Try to take a maximally large “step”.
  - If accepted, moved in the right direction fast.
  - If rejected, narrowed the space of possible ATEs.

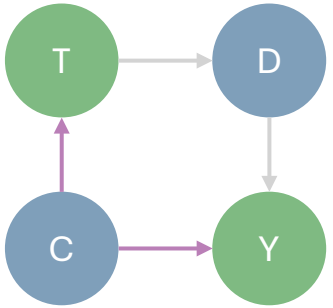
Strategy 1 cannot escape some local minima

- There are cases where a single-edge edit will not affect the ATE of interest at all, but several edits together would.
- Here, no single edge addition affects the ATE.
- However, adding two edges would add C to the adjustment set.



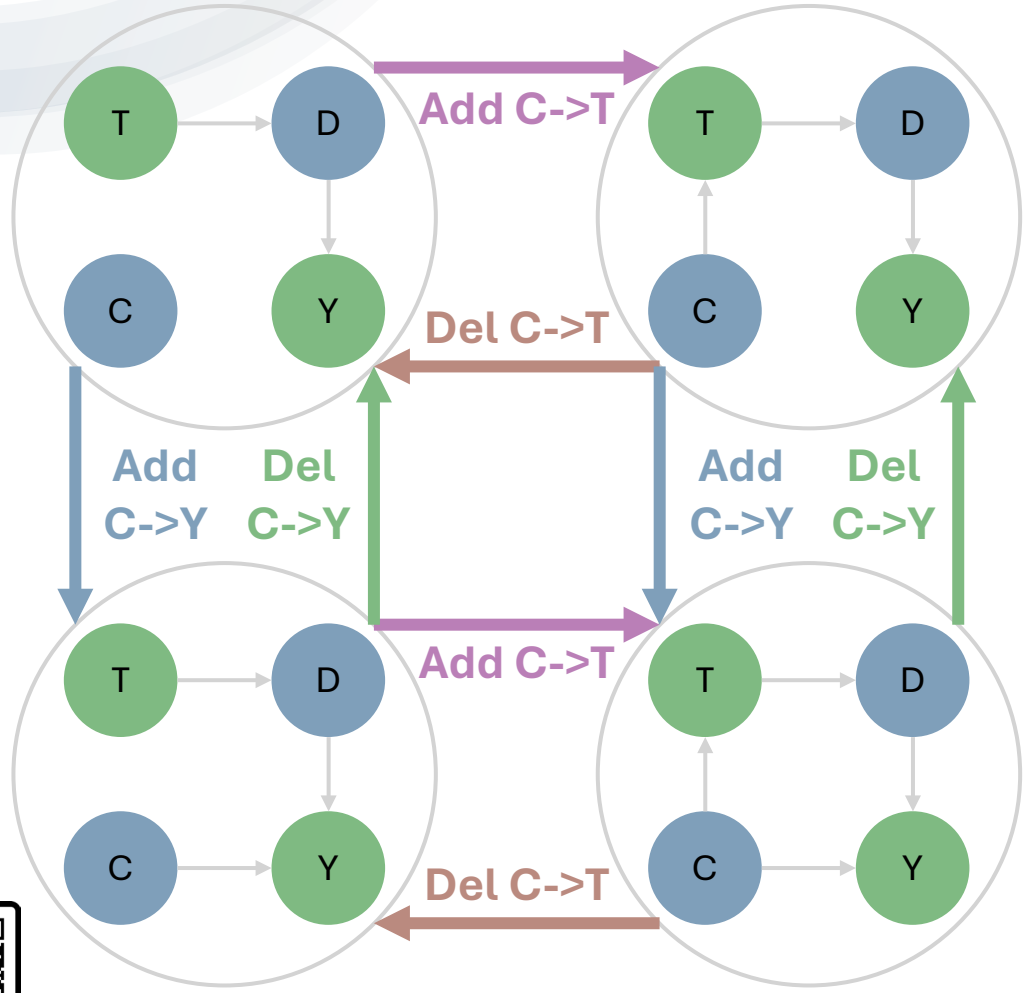






Strategy 2: HeuristicEdit

Strategy 2: Search further into the future



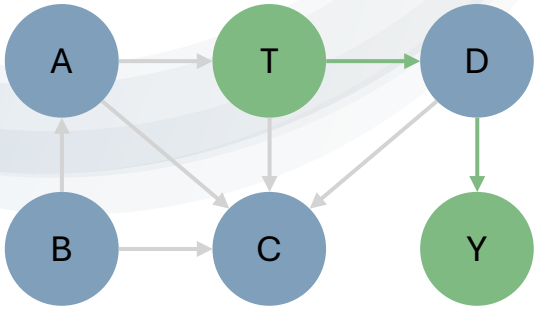
- One way to escape such local minima is to look “further down the line”:
- Explore *sequences* of graph edits and see what ATE difference they achieve.
- Define a *search graph:*
  - A node is a causal graph
  - An edge is a causal graph edge edit.
- Edges have types based the edit.
- Explore using A* and find most common edge types that lead to “good” graphs.

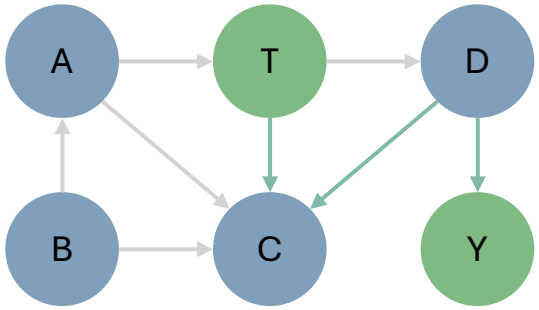
Strategy 2 still essentially processes edges

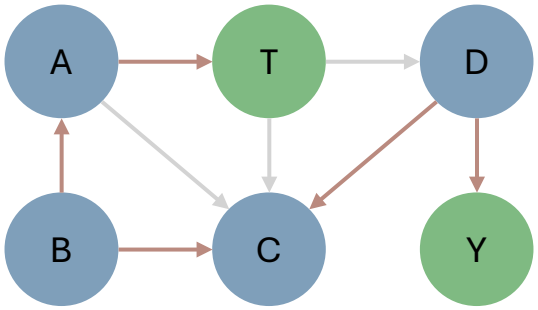
- Although strategy 2 may help uncover edits that are “worth it in the long run”, it still considers individual edge edits when building the search graph.
- This means that the size of the search graph is exponential in the number of causal graph edges.
  - As we will later see, this leads to a very long running time.
  - Will this be worth it in terms of results?

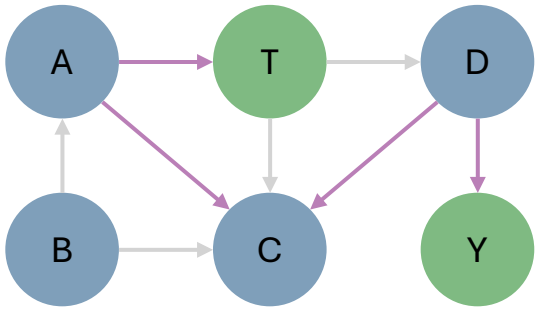
Strategy 3: AdjSetEdit

A closer look at the adjustment set



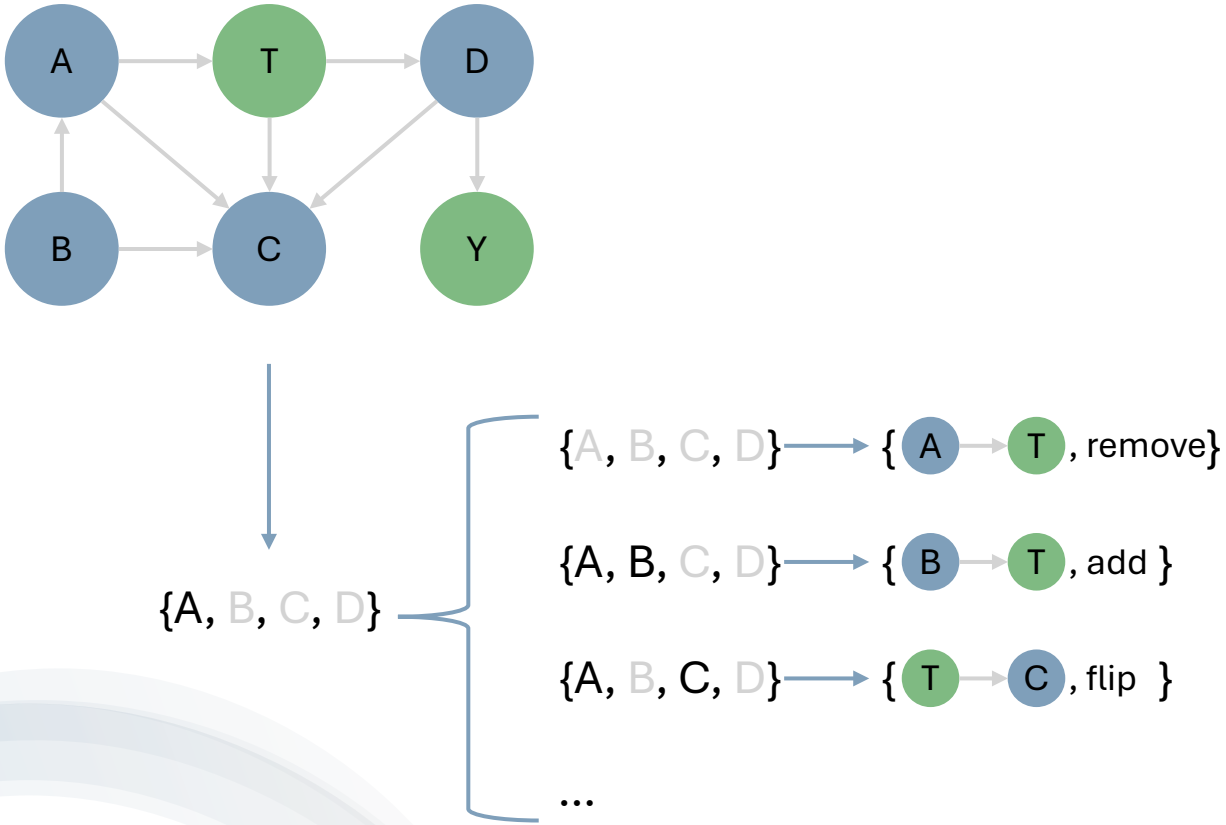






- Causal paths “transmit” the genuine influence of the treatment on the outcome.
- But non-causal paths also exist.
- A sufficient adjustment set **blocks the backdoor** paths.
  - **Blocks:** includes or excludes nodes based on local structure.
  - **Backdoor paths:** They start from T with a “backward” edge.

Strategy 3: Edit “most important adjustment”



- Start from an initial adjustment set based on the starting graph.
- For each variable, toggle its adjustment set membership.
- Translate to edge edits and calculate resulting ATE.
- Return the set of edits that maximizes impact on the ATE of interest.

Evaluating our strategies

What is our measure of success

Evaluation Metrics

- Absolute Relative Error (ARE) in ATE (**ARE_ATE**):

$$ARE\_ATE = \left| \frac{ATE_{current} - ATE_{ground\_truth}}{ATE_{ground\_truth}} \right|$$

- **Latency** per interaction round

Baseline Strategy

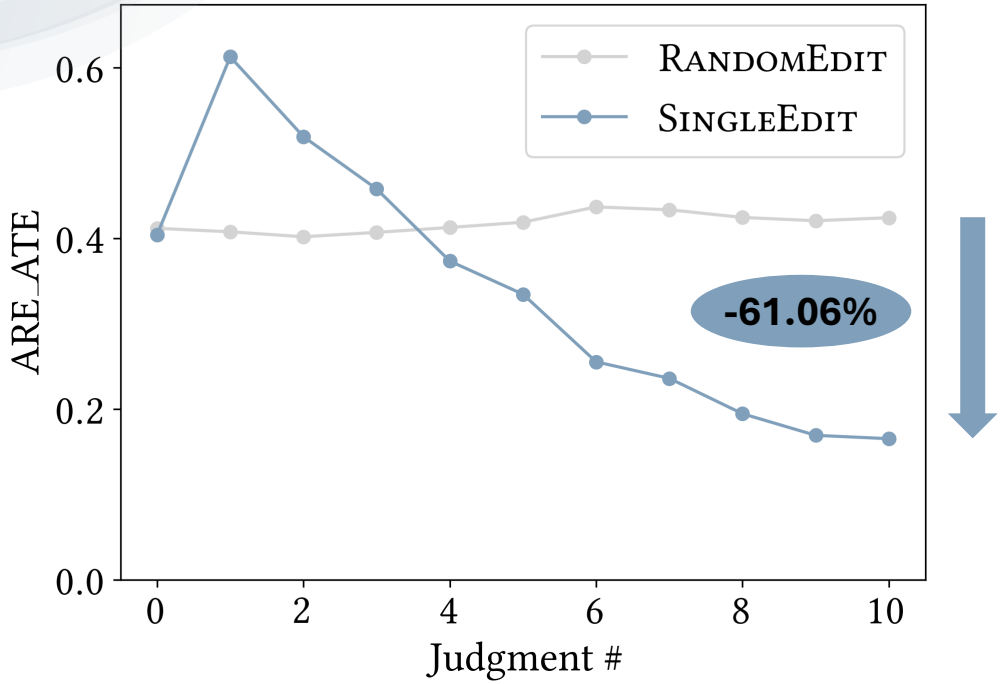
- RandomEdit:
  - Pick single-edge edit uniformly at random.
  - If acceptable, suggest it.
  - Else, sample another one.
  - If none work, suggest nothing.

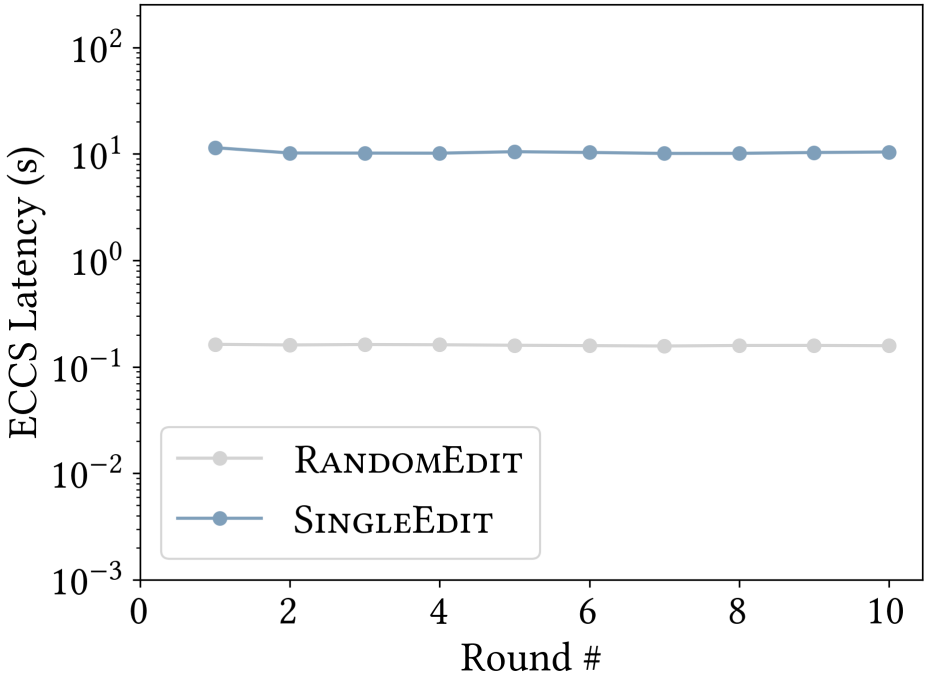
Constructing our experimental instances

Ground Truth Graphs	GenDAG(10, 0.5, −10, 10, 0.001, 2)	10 runs
Datasets	GenData($\mathcal{G}$, 1000, −10, 10)	3 runs/graph
Starting Graphs	GenDAG(10, 0.5, −10, 10, 0.001, 2)	10 runs
Choices of T/Y	Combinations($\mathcal{G}.nodes$, 2)	$\binom{10}{2} = 45$
Strategies	RandomEdit	3 runs
	SingleEdit	1 run
	HeuristicEdit	1 run
	AdjSetEdit	1 run
Total	$\mathbf{10 \times 3 \times 10 \times 45 \times (1 + 1 + 1 + 3) = 81{,}000}$	

- Create ground truth graphs with randomly drawn edges, edge weights and edge noises.
- Generate datasets based on each ground truth graph.
- Create a collection of randomized “starting causal graphs”.
- Iterate over all possible pairs of treatment and outcome variables.
- Observe each strategy over 10 judgments; repeat the randomized baseline 3 times.

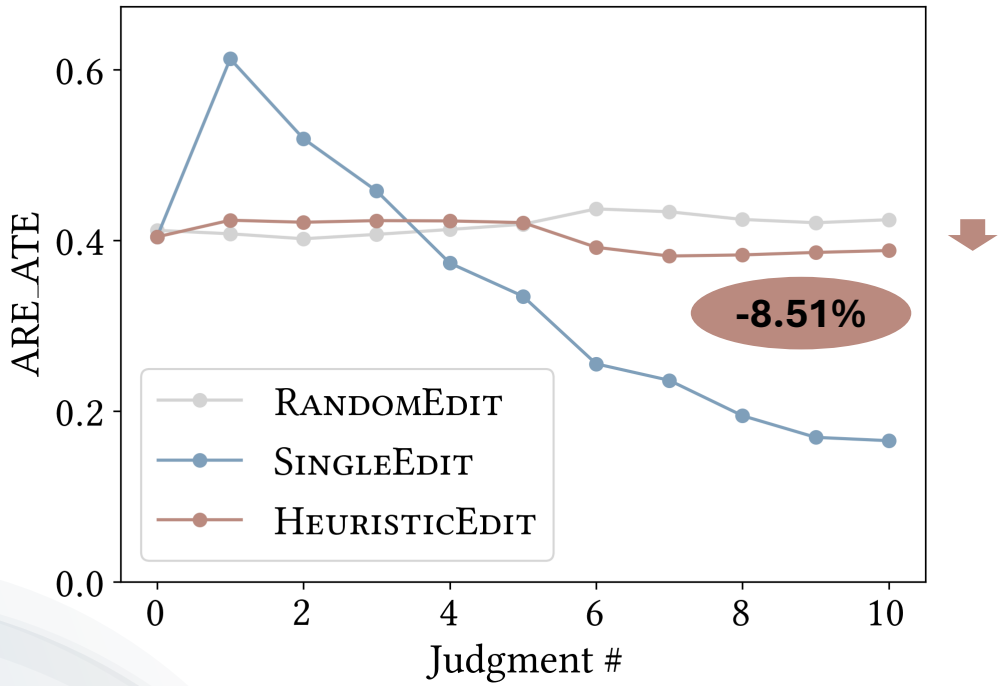
Strategy 1 improves ARE_ATE but is slow

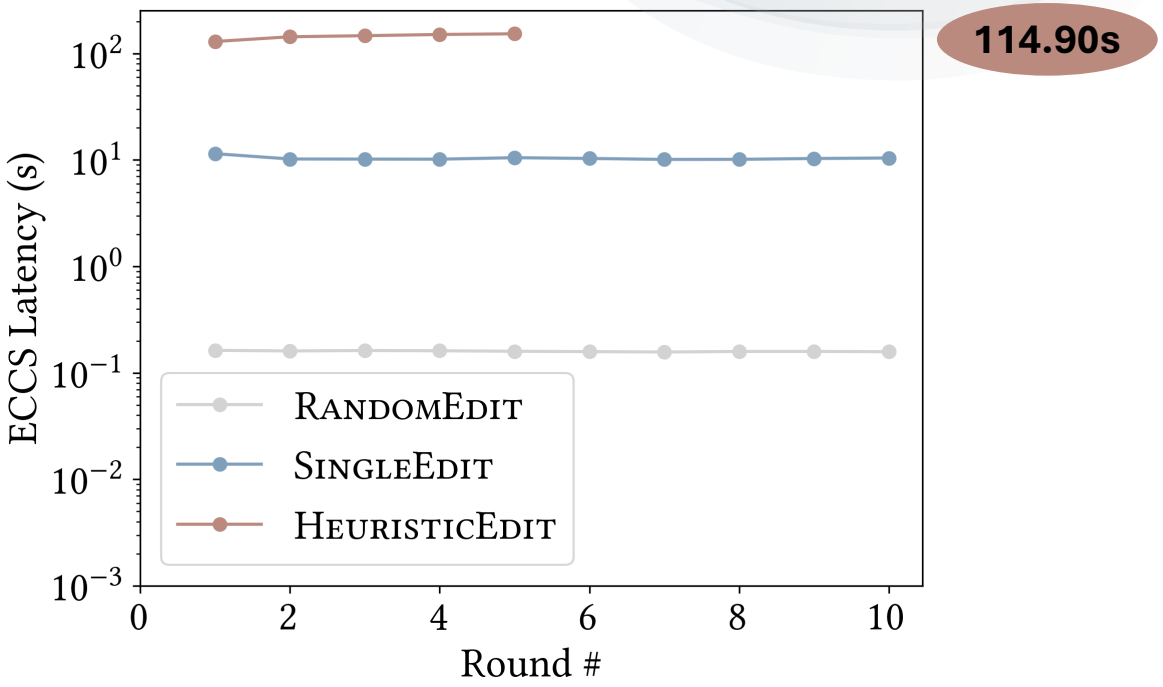




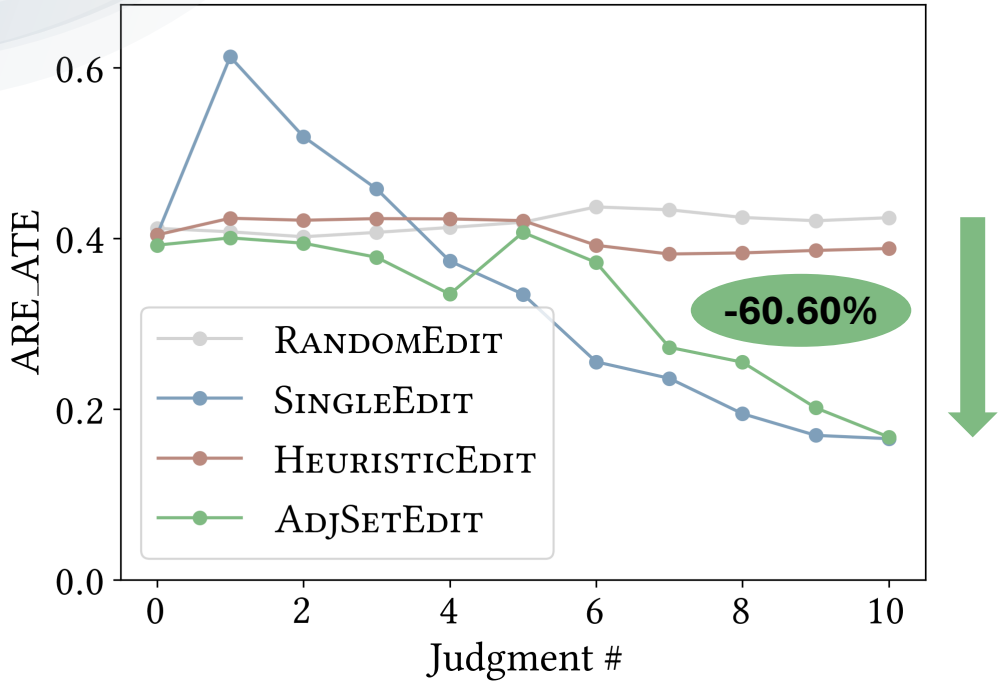
10.39s

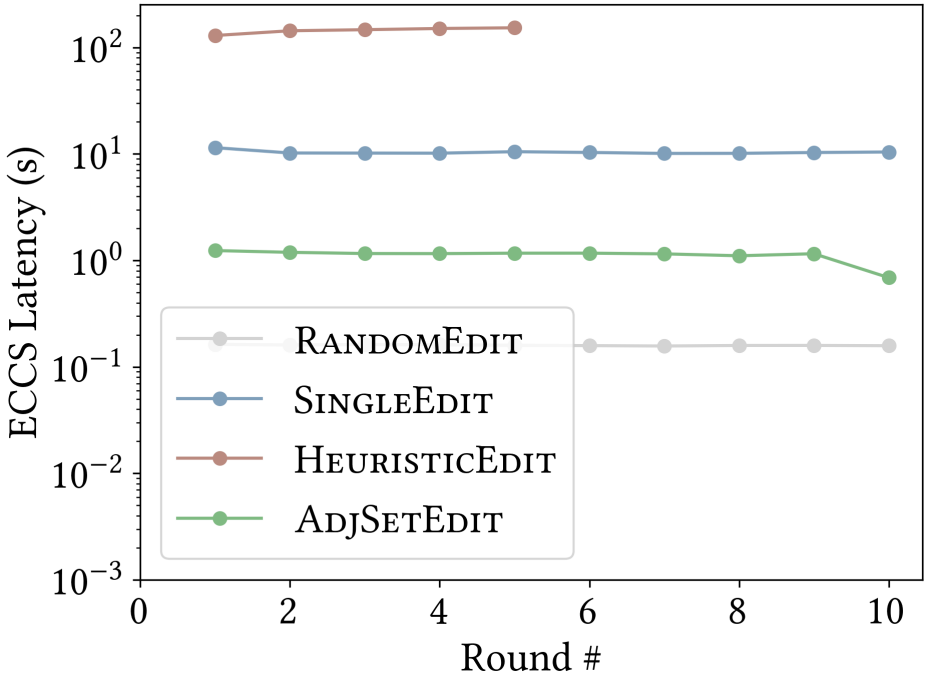
Strategy 2 does not really improve things





Strategy 3 is comparably good, but also fast





Future Directions

- **Prune considered edge edits further**
  - Scaling our current strategies to large #s of variables may be challenging.
  - Could tap recent theoretical results around “local” causal discovery.
- **Tune implementation for performance**
  - Improve latency through more parallelism/partial result caching.
- **Extend experimental evaluation**
  - Sensitivity to causal graph scale and complexity.
  - Use starting graphs generated by causal discovery algorithms.
  - Evaluate on real-world datasets.

Thank you!