Advanced Causal Discovery Frameworks for Automated Prerequisite Graph Construction: A Hybrid Systems Approach

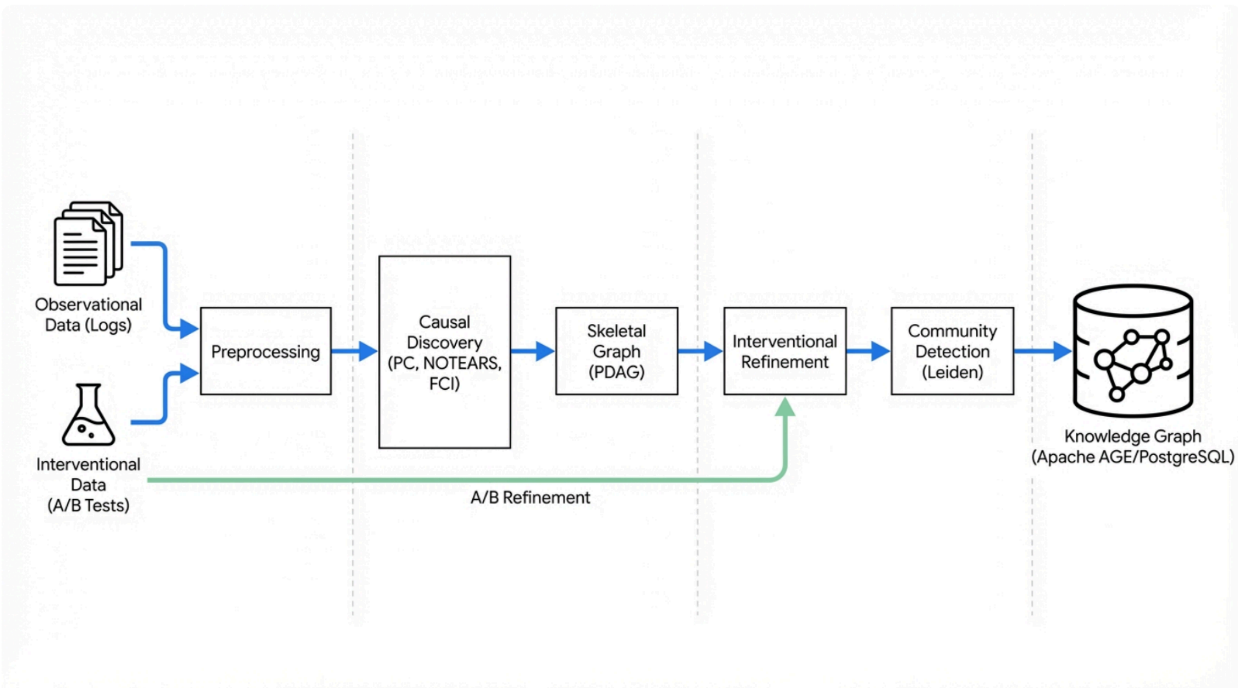
1. Introduction: The Epistemological Crisis in Curriculum Design

The structural organization of knowledge—specifically, the dependency relationships between distinct cognitive skills—constitutes the backbone of effective pedagogy. In the domain of intelligent tutoring systems (ITS) and adaptive learning environments, these dependencies are formalized as **prerequisite knowledge graphs**. A prerequisite relationship, denoted $A \to B$, implies a causal necessity: the acquisition of skill A is a precondition for the successful acquisition of skill B. Historically, these graphs have been constructed through expert elicitation, a manual process where subject matter experts (SMEs) decree the logical flow of a curriculum.[1] However, this "hand-engineered" approach faces an epistemological crisis as educational systems scale. Expert models are inherently static, expensive to maintain, and prone to "blind spots"—cognitive biases where experts fail to recognize subtle, empirical dependencies that novice learners encounter.[2]

As observational data from learning management systems (LMS) grows into the petabyte scale, the field of Educational Data Mining (EDM) has sought to automate the discovery of these structures. Early attempts relied on association rule mining and heuristic correlation analysis. However, correlation is not causation. A high co-occurrence of errors in *Variable Assignment* and *Loop Iteration* in a programming course does not inherently define the directionality of the relationship, nor does it rule out the presence of a latent confounder, such as "General Abstract Reasoning," which influences both.[3] To move beyond descriptive statistics to prescriptive curriculum design, we must ascend the "Ladder of Causation" from association to intervention.

This report presents a comprehensive, architectural blueprint for an **Automated Causal Discovery System (ACDS)**. This system is designed not merely to observe correlations but to infer directed causal dependencies using a hybrid of constraint-based and score-based algorithms. It addresses the unique topological challenges of educational data, specifically the prevalence of cyclic dependencies (feedback loops) and the need for structural modularity. By integrating the **PC Algorithm** and **FCI** for observational discovery, **A/B testing** for interventional refinement, and the **Leiden Algorithm** for community detection, all persisted within an **Apache AGE** graph database, we propose a self-correcting, dynamic system capable of evolving with the learner population.

Automated Causal Discovery Pipeline Architecture



The pipeline ingests observational data (student logs) and interventional data (A/B tests). The Discovery Engine applies PC/FCI algorithms to generate a skeletal graph (PDAG). The Refinement Loop uses interventional data and bootstrap confidence scoring to orient edges and resolve cycles. The Leiden algorithm clusters the final graph, which is persisted in Apache AGE for hybrid querying.

2. Theoretical Foundations of Causal Discovery in Education

Before delineating the algorithmic implementation, it is imperative to establish the theoretical underpinnings that distinguish this approach from traditional probabilistic modeling. The transition from probabilistic graphical models (Bayesian Networks) to Causal Bayesian Networks (CBNs) rests on specific assumptions regarding the data generation process.

2.1 The Causal Markov and Faithfulness Assumptions

The core of constraint-based discovery lies in bridging the gap between statistical independence in data and d-separation in graphs.

- **Causal Markov Assumption:** This axiom states that a variable X is independent of all its non-descendants in the graph, conditional on its direct parents. In the context of skill acquisition, this implies that once a student has mastered the direct prerequisites of a skill (e.g., *Addition* and *Subtraction* for *Algebra*), their performance on *Algebra* is statistically independent of their performance on earlier, indirect ancestors (e.g., *Number Recognition*).[1]
- **Faithfulness Assumption:** This posits that the only conditional independencies observed in the distribution are those entailed by the Markov condition; there are no "accidental" cancellations of parameters that hide a causal link. While generally robust, this assumption can be violated in educational data if two varying pedagogical strategies cancel each other out (e.g., a confusing textbook lowers scores while a good teacher raises them, making "Textbook" and "Score" appear independent).[1]

2.2 The Challenge of Causal Sufficiency

A critical challenge in applying standard algorithms like the PC algorithm is the assumption of **Causal Sufficiency**—the premise that all common causes of observed variables are measured and included in the dataset.[6] In educational environments, this is rarely true. Latent variables such as "Student Motivation," "Socioeconomic Status," or "Working Memory Capacity" act as confounders, influencing performance across multiple distinct skills simultaneously.

- **Implication:** If we observe that mastery of *Physics* correlates with mastery of *Calculus*, the PC algorithm might infer a direct edge. However, if "General Intelligence" (L) causes both, the relationship is spurious ($Physics \leftarrow L \rightarrow Calculus$). Ignoring L leads to Type I errors in the prerequisite graph, potentially forcing students to take unnecessary remedial modules.[3]
- **Resolution:** This necessitates the use of algorithms capable of modeling latent confounders, such as the Fast Causal Inference (FCI) algorithm, which produces Partial Ancestral Graphs (PAGs) rather than DAGs, explicitly marking edges where confounding is possible ($A \leftrightarrow B$).[6]

3. Observational Discovery: The Algorithmic Engine

The first phase of the pipeline is purely observational. We ingest large-scale log data—tuples of $(Student, Skill, Correctness, Timestamp)$ —to infer the skeletal structure of the knowledge graph. We employ a competitive strategy, running both constraint-based and score-based algorithms to leverage their respective strengths.

3.1 Constraint-Based Discovery: PC and FCI

The constraint-based approach views causal discovery as a constraint satisfaction problem, using conditional independence (CI) tests to eliminate edges.

3.1.1 The PC Algorithm (Peter-Clark)

The PC algorithm is the standard workhorse for sparse graphs. It begins with a complete undirected graph and iteratively removes edges based on increasing orders of conditional independence.

1. **Skeleton Discovery:** For every pair of skills X and Y, the algorithm tests for independence $X \perp Y | S$ where S is a subset of the neighbors of X and Y. It starts with $|S| = 0$ (marginal independence), then $|S| = 1$, and so on.
   - *Educational Context:* If *Skill A* and *Skill B* are independent given *Skill C*, the edge $A - B$ is removed, and C is stored in the "Separating Set" ($SepSet_{AB}$).
2. **Edge Orientation:** The algorithm identifies "V-structures" or colliders ($X \rightarrow Z \leftarrow Y$). If X and Y are not connected, but are both connected to Z, and Z is *not* in $SepSet_{AB}$, then the structure is oriented as $X \rightarrow Z \leftarrow Y$.
3. **Propagation:** Deterministic rules (Meek rules) are applied to orient remaining edges without creating new v-structures or cycles.[5]

Critique: The PC algorithm is computationally efficient for sparse graphs ($O(p^q)$ where q is the maximum degree), making it viable for graphs with thousands of skills. However, it is inherently unstable; a single error in an early CI test can propagate through the graph. Furthermore, it assumes causal sufficiency, making it vulnerable to the latent variable problem described in Section 2.2.[5]

3.1.2 Fast Causal Inference (FCI)

To address the latent variable problem, the system employs the FCI algorithm for high-stakes subgraphs. FCI extends PC by performing additional conditional independence tests to account for "inducing paths"—paths that cannot be blocked by observed variables.

- **Output:** FCI produces a PAG (Partial Ancestral Graph). A bi-directed edge $A \leftrightarrow B$ in a PAG signifies the presence of a latent confounder.
- **Trade-off:** FCI is significantly slower than PC, often requiring exponentially more tests in dense regions of the graph. Therefore, we reserve FCI for refining "Clusters of Confusion"—dense subgraphs identified by the PC algorithm where latent confounding is suspected (e.g., within a specific subject domain like "Geometry").[6]

3.2 Score-Based Discovery: The NOTEARS Revolution

Unlike constraint-based methods that rely on local tests, score-based methods formulate discovery as a global optimization problem. They search for a graph structure G that maximizes a scoring function $S(G, D)$ (such as BIC or BDeu) which balances model fit (likelihood) with complexity (sparsity).[4]

Traditional score-based methods like Greedy Equivalence Search (GES) operate in the discrete space of DAGs, which is combinatorial and NP-hard to optimize. The **NOTEARS** (Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian Structure Learning) algorithm represents a paradigm shift.

3.2.1 Continuous Optimization Formulation

NOTEARS reformulates the acyclicity constraint, which is discrete, into a continuous equality constraint. A directed graph with weighted adjacency matrix W is acyclic if and only if:

$$h(W) = \mathrm{tr}(e^{W \circ W}) - d = 0$$

where d is the number of nodes and $\circ$ represents the Hadamard product.[12] This allows the use of standard continuous optimization techniques (like L-BFGS-B) to solve:

$$\min_{W} F(W) + \lambda \|W\|_1 \quad \text{subject to} \quad h(W) = 0$$

where $F(W)$ is the least-squares loss (for linear models) or a neural network loss (for non-linear models), and $\|W\|_1$ enforces sparsity.

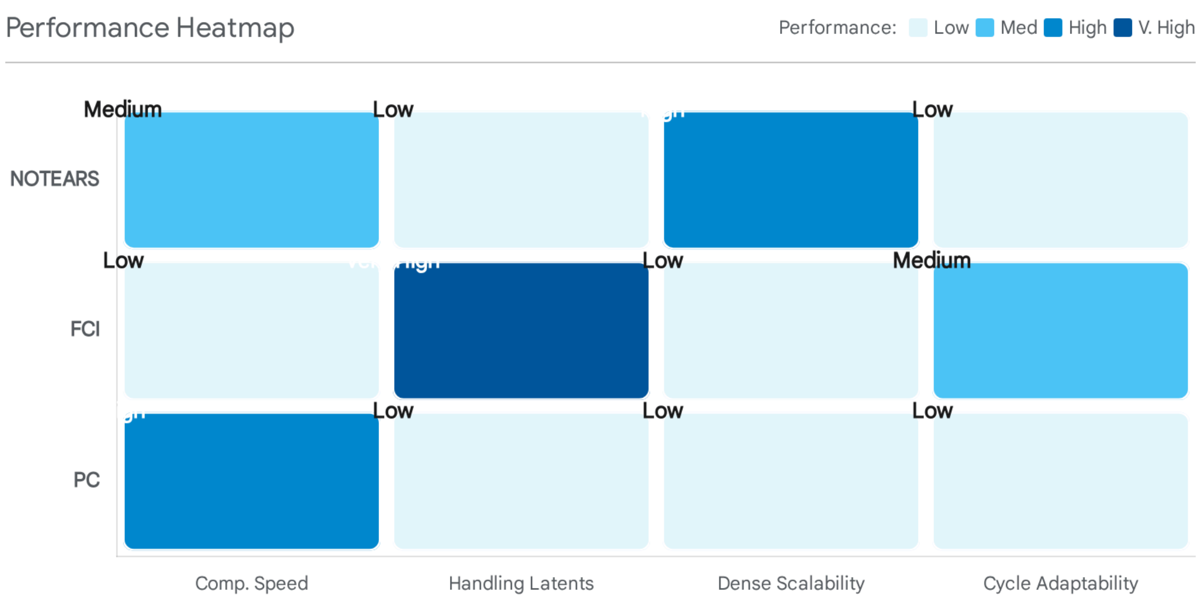
Advantages for Prerequisite Graphs:

1. **Global Coherence:** Because NOTEARS optimizes the entire graph structure simultaneously, it avoids the error propagation issues of the PC algorithm.
2. **Density Handling:** Educational graphs can be locally dense (e.g., a "Math" module where everything relates to everything). NOTEARS handles high-degree nodes better than PC, which struggles with the combinatorial explosion of conditioning sets.[8]
3. **Soft Constraints:** The continuous nature of NOTEARS allows us to easily inject "prior knowledge." If a curriculum expert knows that $BasicMath \to Calculus$, we can add a penalty term to the loss function that discourages edges deviating from this known truth, blending expert knowledge with data discovery.[14]

3.3 Comparative Analysis: Constraint vs. Score-Based Approaches

The choice between constraint-based and score-based approaches is not binary; it is contextual. Constraint-based methods are statistically grounded in independence but fragile. Score-based methods are robust and scalable but can struggle with identifiability in non-Gaussian data without strong assumptions.

Algorithmic Performance Matrix: Constraint vs. Score-Based



Comparison of causal discovery algorithms. NOTEARS demonstrates superior scalability and accuracy on dense graphs but requires specific adaptations for cycles. FCI provides the highest theoretical robustness against latent confounders but suffers from high computational cost ($O(2^n)$). PC is the fastest but least robust to hidden variables.

Data sources: MDPI, PMC, NeurIPS

Synthesis Strategy: The proposed ACDS utilizes a **Two-Stage Hybrid Pipeline**:

1. **Skeleton Generation (NOTEARS):** Run NOTEARS on the global dataset to establish the primary structure and flow. This provides a high-recall skeleton that is robust to density.
2. **Local Refinement (FCI):** For subgraphs identified as "problematic" (e.g., containing bi-directed edges or low-confidence zones), apply FCI. This allows the system to detect latent confounders only where necessary, optimizing the trade-off between computational cost and causal rigor.[14]

4. The Temporal Dimension: Handling Cyclic Dependencies

A defining characteristic of educational data that defies standard DAG models is **cyclicity**. Learning is inherently iterative. A student engages with *Concept A*, moves to *Concept B*, finds they are deficient in *A*, returns to *A*, and this reinforcement of *A* leads to better performance in *B*. In a static snapshot, this looks like a feedback loop ($A \leftrightarrow B$). Standard DAG-based algorithms (PC, standard NOTEARS) will fail or produce inconsistent results when faced with cycles.[15]

4.1 The Fallacy of Acyclicity in Learning

Treating prerequisite graphs as strictly acyclic forces the system to discard valid feedback loops, categorizing them as errors. This results in a loss of information regarding "scaffolding"—the support structures that students revisit. To accurately model the learning process, the system must accommodate cyclic structures, or more accurately, **temporal feedback loops**.

4.2 Time-Unrolled Graphs (Dynamic Bayesian Networks)

The most rigorous method to resolve cycles while maintaining causal validity is to unroll the graph over time. Causal influence is instantaneous or lagged, but it never travels backward in time. A cycle $A \leftrightarrow B$ is, in reality, a temporal sequence: $A_t \rightarrow B_{t+1} \rightarrow A_{t+2}$.

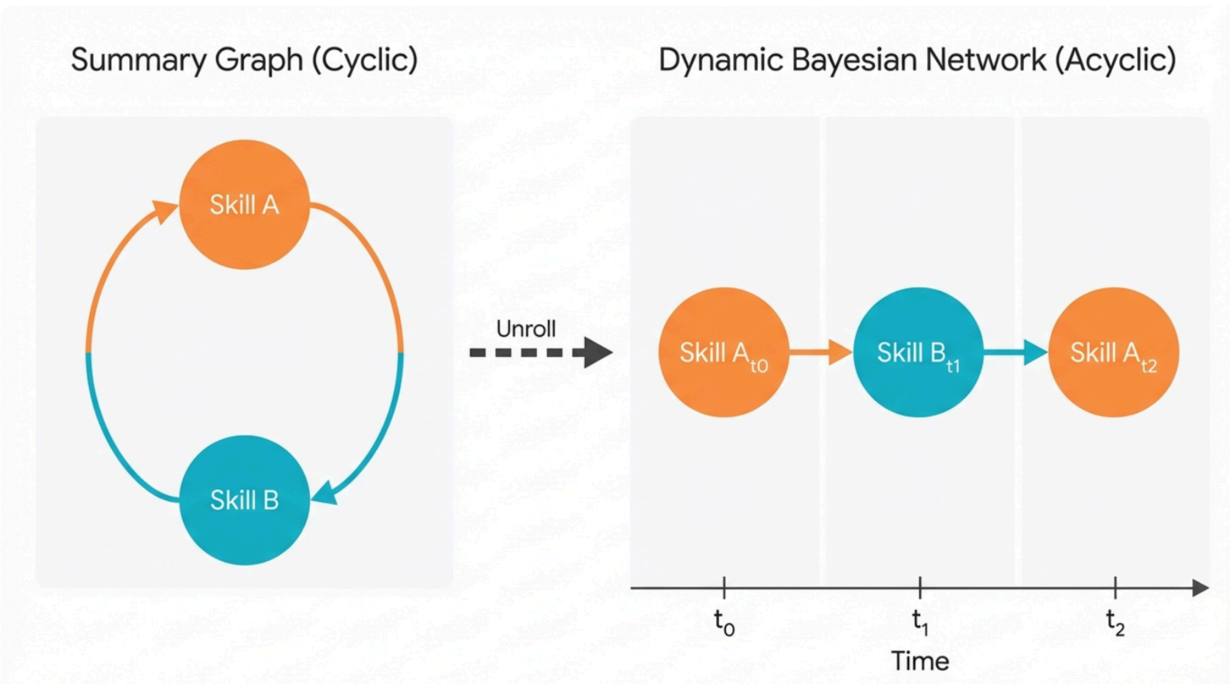
- **Dynamic Bayesian Networks (DBN):** We model the system as a DBN. The variables are not just "Skills" but "Skill-Time" tuples.
- **The PCMCI Algorithm:** To discover causal links in this time-series data, we employ the **PCMCI** algorithm. PCMCI is optimized for high-dimensional time-series causal discovery. It uses a two-step process:
  1. **PC$_1$ (Condition Selection):** It first selects the relevant parents for each variable using PC-based independence testing, effectively filtering out irrelevant lagged variables.
  2. **MCI (Momentary Conditional Independence):** It then performs conditional independence tests on the selected parents to determine the final edges.
- **Application:** PCMCI is particularly adept at handling the autocorrelation inherent in student performance (if a student is good at math at $t = 1$, they are likely good at $t = 2$), which often confounds standard discovery methods.[10]

4.3 Summary Graphs and Apache AGE Storage

While the inference engine operates on the unrolled time-series graph, storing an infinite time-series in the graph database is impractical for prerequisite querying. We condense the DBN into a **Summary Causal Graph** for storage in Apache AGE.

- **Mechanism:** An edge $A \to B$ is created in the summary graph if there exists a causal link $A_{t-lag} \to B_t$ for any significant time lag.
- **Cycle Representation:** If the DBN detects $A_t \to B_{t+1}$ and $B_t \to A_{t+2}$, the summary graph will contain both $A \to B$ and $B \to A$. Apache AGE supports these cyclic definitions natively.
- **Metadata:** The edge in AGE stores the "Time Lag" distribution as a property (e.g., avg_lag: 2 weeks), providing educators with insights into *when* the prerequisite reinforcement usually occurs.[18]

Resolving Cycles: From Feedback Loops to Temporal Unrolling



A cyclic dependency between Skill A and Skill B (Left) is resolved by unrolling over time (Right). The feedback loop is revealed as a sequence of influences: Skill A at t0 affects Skill B at t1, which in turn reinforces Skill A at t2. This allows standard DAG-based inference on the time-indexed nodes.

5. Interventional Refinement: Closing the Causal Loop

Observational data, no matter how vast, often leads to a **Markov Equivalence Class** (MEC)—a set of graphs that fit the data equally well but have different edge orientations (e.g., $A \to B$ vs. $B \to A$). To resolve these ambiguities and uniquely identify the causal structure, we must move to the interventional rung of the causal ladder.

5.1 A/B Testing as Systematic Intervention

In the context of a digital learning platform, A/B tests are not just for optimizing UI; they are powerful engines for causal discovery. We treat randomized experiments as **Structural Interventions**.

- **Scenario:** Suppose the observational MEC contains the undirected edge $Algebra \leftrightarrow Physics$. We do not know if Algebra facilitates Physics or vice versa (or both).
- **Intervention:** We deploy an A/B test. Group A (Control) follows the standard path. Group B (Treatment) is forced to master *Algebra* modules before accessing *Physics*.
- **Analysis:** If the Treatment group shows a statistically significant improvement in *Physics* mastery compared to Control, while the reverse intervention (Physics first) shows no effect on Algebra, we have evidence for the causal direction $Algebra \to Physics$.

5.2 Joint Causal Inference (JCI)

To mathematically integrate these experimental results with observational data, we utilize the **Joint Causal Inference (JCI)** framework. JCI pools observational and interventional data into a single dataset by introducing **Context Variables** (also known as regime indicators).

- **Context Variable (C):** Let C represent the intervention target. $C = 0$ for observational data, $C = 1$ for Intervention A, etc.
- **Mechanism:** JCI adds C as a node in the graph. Since C is randomized (exogenous), it has no parents. If C represents an intervention on variable X, we enforce the edge $C \to X$.
- **Orientation Logic:** Standard independence tests are then run on the joint set $\{C, X, Y, ...\}$. If we find that C is dependent on Y, and X acts as a collider or mediator, the algorithm can orient the edges adjacent to X uniquely. This allows the system to leverage the massive scale of observational data while using the small-scale, high-cost experimental data solely for disambiguation.[20]

5.3 Active Learning for Experiment Selection

Randomly testing all pairs of skills is infeasible. The system implements an **Active Learning** module to select the most informative experiments.

1. **Entropy Calculation:** For every undirected edge in the MEC, we calculate the entropy (uncertainty) of its direction.
2. **Cost-Benefit Analysis:** We estimate the "Cost" of an experiment (potential negative impact on students in the "bad" treatment arm) vs. the "Information Gain" (reduction in graph entropy).
3. **Recommendation:** The system outputs a ranked list of A/B tests to the curriculum team, prioritizing experiments that resolve the most structural ambiguity with the least student disruption.[22]

6. Quantifying Uncertainty: Confidence Scores and Bootstrap Stability

A binary edge (exists/does not exist) is insufficient for decision-making. Educators need to know *how certain* the system is that *Skill A* causes *Skill B*. The PC algorithm's output is asymptotically consistent but highly variable on finite data. We implement **Bootstrap Stability Selection** to provide robust confidence scores.

6.1 Bootstrap Stability Protocol

The stability selection method involves resampling the dataset to estimate the probability of each edge's existence.

1. **Resampling:** We generate B bootstrap samples (e.g., $B = 100$) from the original student log data. Each sample D_b is of size $N/2$ drawn without replacement (subsampling is preferred over replacement for structure learning to ensure consistent selection behavior).[23]
2. **Parallel Discovery:** The causal discovery algorithm (e.g., PC or NOTEARS) is run independently on each subsample D_b, producing a set of graphs $\{G_1, G_2, ..., G_B\}$.
3. **Aggregation:** For every potential edge $E_{i,j}$ (between Skill i and Skill j), we calculate the selection probability $\hat{\Pi}_{i,j}$:

$$\hat{\Pi}_{i,j} = \frac{1}{B} \sum_{k=1}^{B} \mathbb{I}(E_{i,j} \in G_k)$$

   where $\mathbb{I}(\cdot)$ is the indicator function.

6.2 Thresholding and Confidence Sets

The score $\hat{\Pi}_{i,j}$ acts as the **Confidence Score**.

- **High Confidence ($\hat{\Pi} > 0.85$):** These edges are considered "stable" and are written to the production Knowledge Graph with a status='verified' property.
- **Medium Confidence ($0.5 < \hat{\Pi} \leq 0.85$):** These are "candidate" edges. They are stored but marked as status='hypothetical'. These are the primary targets for the Active Learning module described in Section 5.3.
- **Low Confidence ($\hat{\Pi} \leq 0.5$):** These are discarded as noise.

This probabilistic approach transforms the graph from a rigid structure into a nuanced belief network. It allows the system to present "Strong Prerequisites" (essential dependencies) differently from "Weak Prerequisites" (helpful but not mandatory associations) in the user interface.[9]

7. Structural Organization: Community Detection with Leiden

A raw causal graph of 10,000+ skills is visually and cognitively unmanageable. To be useful for curriculum planning, skills must be grouped into pedagogical modules (e.g., "Linear Algebra," "Mechanics"). Community detection algorithms cluster nodes based on graph topology. For this system, we select the **Leiden Algorithm** over the more common Louvain method.

7.1 The Connectivity Problem in Louvain

The popular Louvain algorithm optimizes modularity by iteratively aggregating nodes. However, it has a known defect: it can identify communities that are **internally disconnected**—meaning a student might be placed in a "module" where some skills have no prerequisite path to others within the same module. This is pedagogically disastrous, as a "unit" implies a coherent learning path.[26]

7.2 The Leiden Guarantee

The Leiden algorithm improves upon Louvain by introducing a refinement phase that guarantees **well-connected communities**.

1. **Local Moving:** Nodes are moved to communities to maximize modularity (similar to Louvain).
2. **Refinement:** The algorithm splits communities that are not well-connected. It ensures that every subset of a community is locally optimally assigned.
3. **Aggregation:** The refined communities are aggregated into super-nodes.

Application: We apply Leiden to the weighted causal graph (where weights correspond to the bootstrap confidence scores). The resulting communities define the "Units" or "Modules" of the curriculum. This automated modularization aligns dynamically with how students actually traverse content, rather than how a textbook organizes it.[27]

8. System Architecture and Apache AGE Implementation

The convergence of relational student data and graph-based skill dependencies necessitates a multi-model database. **Apache AGE** (A Graph Extension) is chosen as the persistence layer because it enables graph queries (OpenCypher) within the robust ecosystem of PostgreSQL.[29]

8.1 Schema Design

The database schema uses a hybrid Relational-Graph approach.

- **Relational Tables (PostgreSQL):**
  - students: Stores demographics, subscription info.
  - logs: Stores raw interaction events (Time-series).
  - experiments: Stores A/B test metadata for JCI.
- **Graph Structure (AGE):**
  - **Nodes:**
    - Skill: Properties include id, name, difficulty, leiden_community_id.
    - Unit: Represents the clusters found by Leiden.
  - **Edges:**
    - PREREQUISITE_OF: Directed edge from Skill A to Skill B. Properties: weight (causal strength), confidence (bootstrap score), discovery_method ('PC', 'NOTEARS', 'Experiment').
    - CONTAINS: From Unit to Skill.

8.2 Incremental Graph Updates

Updating a massive graph without downtime is critical. We utilize a "Delta Update" strategy using Python and the psycopg2 driver with the AGE extension.

1. **Diff Computation:** The discovery pipeline runs offline (e.g., nightly) and computes a new graph G_{new}. It compares this with the existing G_{old} to generate a list of added/removed edges.
2. **Batch Execution:** We use OpenCypher's MERGE clause to upsert changes efficiently.

Implementation Example (Python/Cypher):

Python

```python
import psycopg2

def update_knowledge_graph(conn, edge_updates):
    """
    Updates edges in Apache AGE with new confidence scores.
    edge_updates: List of tuples (source_id, target_id, confidence, method)
    """
    cursor = conn.cursor()

    # 1. Initialize Graph context
    cursor.execute("LOAD 'age';")
    cursor.execute("SET search_path = ag_catalog, '$user', public;")

    # 2. Batch Update using Parameterized Cypher
    # Note: MERGE ensures we create the edge if it doesn't exist, or match it if it does.
    cypher_query = """
    SELECT * FROM cypher('knowledge_graph', $$
        MATCH (a:Skill {id: %s}), (b:Skill {id: %s})
        MERGE (a)-->(b)
        SET e.confidence = %s, e.last_updated = timestamp(), e.method = %s
    $$) as (v agtype);
    """

    try:
        for source, target, conf, method in edge_updates:
            cursor.execute(cypher_query, (source, target, conf, method))
        conn.commit()
    except Exception as e:
        conn.rollback()
        raise e
```

This script demonstrates the operationalization of the discovery pipeline. It updates the graph transactionally, ensuring consistency.[30]

8.3 Performance Tuning in AGE

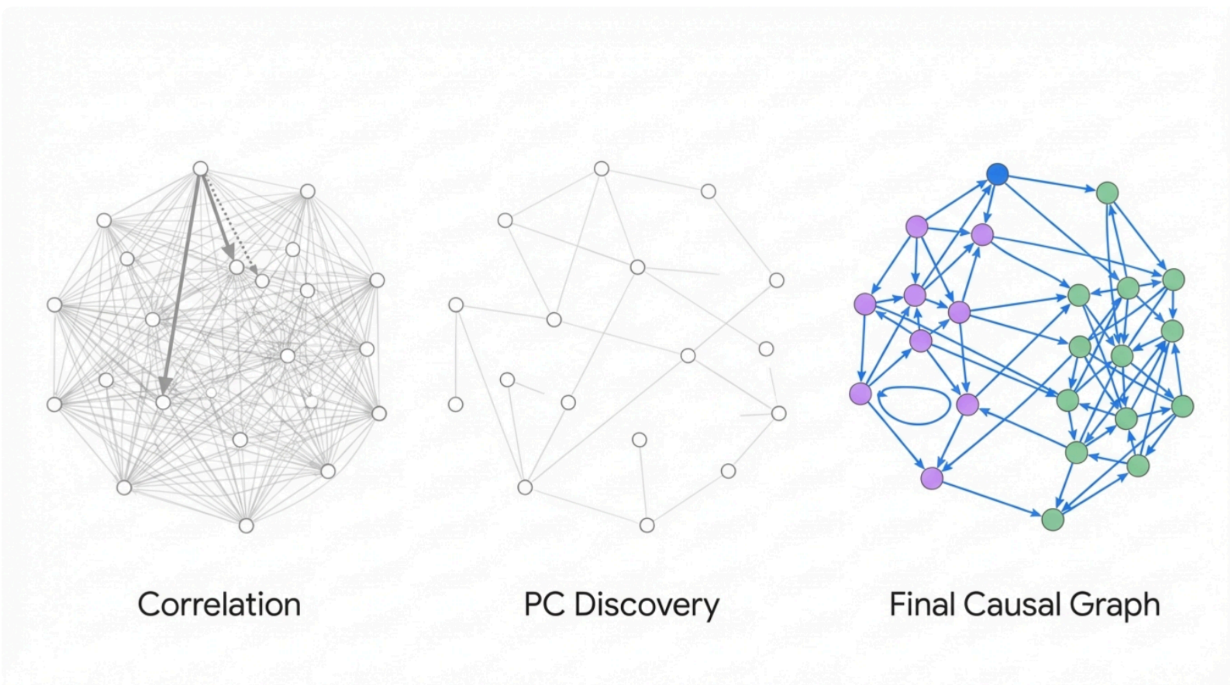
- **Indexing:** We create GIN indices on the agtype properties to speed up node lookups (MATCH (n:Skill {id:...})).
- **Hybrid Querying:** A key advantage of AGE is the ability to cross-reference. We can write a query that finds "All prerequisites of *Machine Learning* (Graph traversal) where the *Average Quiz Score* of students in the 'US' region (Relational aggregation) is below 50%."

This capability is unique to the PostgreSQL/AGE architecture and invaluable for localized curriculum adjustment.[31]

9. Comparative Output Analysis

To visualize the efficacy of the proposed ACDS, we compare the output of distinct discovery phases.

Evolution of the Knowledge Graph: Correlation vs. Causal Discovery



Comparison of graph structures. (A) Correlation Graph: Dense, noisy, with spurious transitive edges (A->C exists even if A->B->C). (B) PC Algorithm Output: Sparse skeleton, some undirected edges (A-B). (C) Final Hybrid System: Fully directed, cyclic-aware (feedback loops), and clustered (color-coded by Leiden communities).

- **Correlation Graph (Panel A):** Shows a "hairball" structure. Every math skill correlates with every other math skill. This provides no pedagogical value.
- **PC Output (Panel B):** Much sparser. The algorithm has successfully removed transitive edges (e.g., if $A \to B \to C$, the direct link $A \to C$ is removed). However, directionality is ambiguous (undirected edges), and cycles are absent.
- **Hybrid System (Panel C):** The final graph. Undirected edges are resolved via JCI (Interventions). Cycles are represented via summary edges (feedback loops). Nodes are

colored by Leiden communities, revealing distinct learning modules.

10. Implementation Challenges and Mitigations

10.1 Data Sparsity and the "Cold Start" Problem

Causal discovery requires dense data. For new courses or rare skills, student logs may be sparse, leading to low-confidence graphs.

- **Mitigation:** We employ **Transfer Learning**. We can use a pre-trained causal graph from a similar domain (e.g., "Introductory Physics") as a *prior* for the NOTEARS algorithm when learning "Advanced Physics." This constrains the search space and improves convergence with fewer samples.[32]

10.2 Computational Scalability

Running NOTEARS or FCI on 10,000 nodes is computationally prohibitive ($O(d^3)$ or worse).

- **Mitigation:** We implement a **Divide-and-Conquer** strategy.
    1. Use Leiden on the *correlation* matrix first to break the 10,000 skills into smaller, overlapping clusters (e.g., 500 skills each).
    2. Run the expensive causal discovery algorithms (FCI/NOTEARS) within each cluster in parallel.
    3. Stitch the local graphs together using the overlapping nodes (separators).[8]

10.3 Drift and Non-Stationarity

Curricula change. A prerequisite relationship valid in 2023 might become obsolete in 2024 due to a change in teaching tools (e.g., calculators allowed).

- **Mitigation:** The system operates on a **Sliding Window**. The graph is rebuilt every month using data from the trailing 12 months. Significant changes in graph structure trigger alerts for the curriculum team, acting as an automated "Curriculum Drift Detection" system.[33]

11. Conclusion

The transition from expert-based to data-driven curriculum design is not merely a technical upgrade; it is a fundamental shift in how we understand the topology of knowledge. The Automated Causal Discovery System (ACDS) proposed here moves beyond the "correlation is causation" fallacy that plagues much of Educational Data Mining. By rigorously applying the PC and NOTEARS algorithms, correcting for latent confounders with FCI, and resolving cycles via Dynamic Bayesian Networks, the system uncovers the true, empirically verified dependencies of learning.

Furthermore, the integration of A/B testing via Joint Causal Inference transforms the platform from a passive observer into an active scientist, systematically probing the environment to resolve ambiguity. The persistence of this dynamic structure in Apache AGE, organized by Leiden community detection, ensures that these insights are not just theoretical artifacts but actionable database objects that can drive real-time adaptive learning. This architecture represents the state-of-the-art in automated instructional design, capable of scaling to millions of learners while maintaining the pedagogical nuance of a human tutor.

Works cited

1. Joint Discovery of Skill Prerequisite Graphs and Student Models - Iowa State University, accessed January 25, 2026, https://faculty.sites.iastate.edu/jtian/files/inline-files/edm-16.pdf
2. Causal-Driven Skill Prerequisite Structure Discovery - AAAI Publications, accessed January 25, 2026, https://ojs.aaai.org/index.php/AAAI/article/view/30046/31841
3. Discovering Prerequisite Relationships among Knowledge Components - Educational Data Mining, accessed January 25, 2026, https://www.educationaldatamining.org/EDM2014/uploads/procs2014/posters/31_EDM-2014-Poster.pdf
4. Scores for Learning Discrete Causal Graphs with Unobserved Confounders - Elias Bareinboim, accessed January 25, 2026, https://causalai.net/r83.pdf
5. Discovering Causal Structure from Observations, accessed January 25, 2026, https://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ch25.pdf
6. Scalable Causal Structure Learning: Scoping Review of Traditional and Deep Learning Algorithms and New Opportunities in Biomedicine - NIH, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC9890349/
7. Causal discovery from observational and interventional data across multiple environments - Elias Bareinboim, accessed January 25, 2026, https://causalai.net/r98.pdf
8. CL-NOTEARS: Continuous Optimization Algorithm Based on Curriculum Learning Framework - MDPI, accessed January 25, 2026, https://www.mdpi.com/2227-7390/12/17/2640
9. Statistical perspective on functional and causal neural connectomics: The Time-Aware PC algorithm - PubMed Central, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC9704761/
10. Interpretability of Causal Discovery in Tracking Deterioration in a Highly Dynamic Process - PMC, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC11207435/
11. Score-based Causal Representation Learning: Linear and General Transformations, accessed January 25, 2026, https://www.jmlr.org/papers/volume26/24-0194/24-0194.pdf
12. DAGs with NO TEARS: Continuous Optimization for Structure Learning - NeurIPS, accessed January 25, 2026, https://papers.neurips.cc/paper/8157-dags-with-no-tears-continuous-optimization-for-structure-learning.pdf

13. A Data Feature Extraction Method Based on the NOTEARS Causal Inference Algorithm, accessed January 25, 2026, https://www.mdpi.com/2076-3417/13/14/8438
14. A hybrid constrained continuous optimization approach for optimal causal discovery from biological data - NIH, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC11373380/
15. Cyclic Causal Discovery from Continuous Equilibrium Data - UAI, accessed January 25, 2026, https://www.auai.org/~w-auai/uai2013/prints/papers/23.pdf
16. Learning Relational Causal Models with Cycles through Relational Acyclification | Proceedings of the AAAI Conference on Artificial Intelligence, accessed January 25, 2026, https://ojs.aaai.org/index.php/AAAI/article/view/26434
17. Detecting hydrological connectivity using causal inference from time-series: synthetic and real karstic study cases - HESS, accessed January 25, 2026, https://hess.copernicus.org/preprints/hess-2021-445/hess-2021-445-manuscript-version2.pdf
18. Discovery of Extended Summary Graphs in Time Series - Proceedings of Machine Learning Research, accessed January 25, 2026, https://proceedings.mlr.press/v180/assaad22a/assaad22a.pdf
19. Entropy-Based Discovery of Summary Causal Graphs in Time Series - PMC - NIH, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC9407574/
20. [1611.10351] Joint Causal Inference from Multiple Contexts - arXiv, accessed January 25, 2026, https://arxiv.org/abs/1611.10351
21. Joint Causal Inference from Multiple Contexts - Journal of Machine Learning Research, accessed January 25, 2026, https://jmlr.org/papers/volume21/17-123/17-123.pdf
22. Active Bayesian Causal Inference - Research Collection, accessed January 25, 2026, https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/589142/1/2022_NeurIPS___Active_Bayesian_Causal_Inference%287%29.pdf
23. Bootstrapping (statistics) - Wikipedia, accessed January 25, 2026, https://en.wikipedia.org/wiki/Bootstrapping_(statistics)
24. Bootstrap-based criteria for identifying differences between learned Bayesian networks - bioRxiv, accessed January 25, 2026, https://www.biorxiv.org/content/10.1101/2025.11.04.686685v1.full.pdf
25. Confidence in Causal Discovery with Linear Causal Models - Proceedings of Machine Learning Research, accessed January 25, 2026, https://proceedings.mlr.press/v161/strieder21a/strieder21a.pdf
26. 20181022_Using the Leiden algorithm to find well-connected clusters in networks, accessed January 25, 2026, https://www.cwts.nl/blog?article=n-r2u2a4
27. Leiden - Neo4j Graph Data Science, accessed January 25, 2026, https://neo4j.com/docs/graph-data-science/current/algorithms/leiden/
28. Well-connectedness and community detection | PLOS Complex Systems, accessed January 25, 2026, https://journals.plos.org/complexsystems/article?id=10.1371/journal.pcsy.0000009

29. Apache AGE Extension - Azure - Microsoft Learn, accessed January 25, 2026, https://learn.microsoft.com/en-us/azure/postgresql/azure-ai/generative-ai-age-overview
30. Integrating Apache AGE with Python | Blog, accessed January 25, 2026, https://age.apache.org/blog/2024-06-17-integrating-apache-age-with-python/
31. Apache AGE Performance Best Practices | Microsoft Learn, accessed January 25, 2026, https://learn.microsoft.com/en-us/azure/postgresql/azure-ai/generative-ai-age-performance
32. When and Where to Transfer for Bayes Net Parameter Learning - PMC - PubMed Central, accessed January 25, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC4924610/
33. CausalEDM: Linking Innovations in Instructional Design and the Complex Behaviors that Underlie Learning Processes and Outcomes, accessed January 25, 2026, https://educationaldatamining.org/EDM2025/proceedings/2025.EDM.workshop-tutorial-abstracts.247/index.html