1Cademy: Progressive Disclosure for Collaborative Authoring and Local Inspection of Large Prerequisite Graphs

Anonymous Author(s)

Abstract

Collaborative prerequisite-graph authoring creates a local-versus-global interaction problem: contributors must place new nodes in a shared dependency structure without being overwhelmed by the full graph. We present 1Cademy, a deployed system that addresses this problem with persistent personal views, mandatory structural placement of each new contribution, and *Path-Preserving Virtual Edges (PPVE)* that bridge visible endpoints whose connecting paths pass entirely through hidden intermediates. In a semester-long deployment (937 nodes, 48 active users), the median contributor's personal view contained only 2% of the shared graph despite a mandatory linking rule that forced traversal before every new contribution. A deployment analysis shows a roughly 18× reduction in rendered edges (real plus virtual) under progressive disclosure compared to a dense view, and interviews with 14 individuals reveal anchor-then-explore navigation strategies that manage information overload. As a secondary formative result, a two-model LLM audit shows that AI-generated prerequisite edges are dominated by structural false positives (over 80% for both models), further motivating bounded local inspection. Together, these results support a single claim: in collaborative prerequisite-graph authoring, persistent personal views, mandatory structural placement, and PPVE can keep local authoring neighborhoods compact without severing dependency context, even at near-thousand-node scale.

CCS Concepts

• **Human-centered computing** → **Interactive systems and tools**; *HCI design and evaluation methods*; • **Applied computing** → *Education*.

Keywords

progressive disclosure, collaborative knowledge graphs, prerequisite graphs, graph authoring, graph visualization, deployment study

1 Introduction

A contributor to a large collaborative knowledge base needs to add a new node about ridge regression. To place it correctly, she must locate the existing node on regularization—but the shared graph has grown to 937 concepts connected by nearly a thousand prerequisite edges. Displaying the full graph at once would overwhelm the search for the right placement site; showing nothing would leave her without structural context. This tension between navigation scope and local focus is fundamental to collaborative prerequisite-graph authoring, where each directed edge ($A \rightarrow B$) asserts that concept A must be understood before concept B [15, 26, 27]. As these shared knowledge graphs grow dense and highly interconnected, displaying them in full obscures the local structure that matters for reading, authoring, and inspecting dependency claims [8, 9].

Progressive disclosure—beginning with a minimal view and revealing structure on demand—is a well-established strategy for managing visual complexity [7, 17, 21, 24]. However, most prior systems target single-user graph *exploration* of static datasets. Collaborative prerequisite-graph *authoring* introduces a different challenge: multiple contributors must independently navigate, extend, and verify local structure within a shared graph that grows over time, all while maintaining structural coherence without a central editor.

We present **1Cademy**, a deployed interactive system that applies progressive disclosure to support collaborative authoring of large prerequisite graphs. 1Cademy combines three mechanisms aimed at this authoring problem: persistent personal views that expose only the contributor's current neighborhood, mandatory structural placement that anchors each new node in existing graph structure, and *Path-Preserving Virtual Edges (PPVE)* that bridge visible endpoints whose connecting paths pass entirely through hidden nodes. Each node is therefore managed in one of three visibility states—**Hidden**, **Collapsed**, or **Expanded**—while the visible subgraph remains structurally legible without rendering the full graph. A natural concern with this coupling is that mandatory traversal will progressively expand personal views as contributors accumulate navigation history, eroding the complexity reduction that progressive disclosure is intended to provide. Our deployment evidence addresses this concern directly: personal views remained compact even under sustained authoring pressure.

We ground one central claim in two empirical studies and use a formative audit only as secondary design rationale: (1) a semester-long deployment analysis showing that contributors maintained compact personal views even as the shared graph scaled; (2) qualitative interviews detailing how users adopted anchor-then-explore navigation strategies to manage information overload; and (3) a formative two-model LLM audit that characterizes the prerequisite-edge prediction task, showing that AI-generated candidates are dominated by structural false positives and motivating bounded local inspection as a downstream use case. Across these studies, the paper supports a single claim: in collaborative prerequisite-graph authoring, persistent personal views, mandatory structural placement, and PPVE can keep local authoring neighborhoods compact without severing dependency context—as the deployment shows, the median contributor's view remained at 2% of a 937-node shared graph despite sustained traversal pressure.

In summary, this paper contributes: (1) a collaborative-authoring interaction model that combines persistent personal views, mandatory structural placement, and PPVE to keep local structure legible inside a shared evolving prerequisite graph; (2) deployment and interview evidence showing that bounded personal views persisted under real authoring pressure, including a median final view of 19 visible nodes, a 109-versus-19 non-trivial-compactness comparison, and anchor-then-explore navigation practices under the design; and (3) as a secondary formative result, a two-model LLM audit

with a quantified error taxonomy showing that AI-generated prerequisite edges are dominated by structural false positives (83.2% for Flash, 80.7% for Pro), motivating local structural inspection as an additional application context. This combination addresses a gap at the intersection of graph visualization and collaborative authoring: how to keep authoring interactions local in a shared graph that grows with each contribution.

Earlier prerequisite-map systems—QMaps [27] and prior 1Cademy work [26]—established the broader collaborative prerequisite-linking platform; this paper's new scientific contribution is specifically the local-view interaction model, PPVE as the hidden-intermediate bridging mechanism within those views, and deployment evidence that bounded personal views persist under sustained authoring pressure. Recent adjacent systems address different interaction problems: knowledge workspaces and LLM augmentation (Knoll [30], Garden of Papers [12]), embedding-guided exploration of existing graphs (KGScope [28]), or offline pairwise candidate review (ACE [1]). None studies collaborative authoring of a shared evolving prerequisite graph under bounded local views with deployment evidence that those views remain compact.

For context, although the behavioral analyses in this paper focus on one archived course deployment, 1Cademy has broader platform use beyond that single case. Across the full platform database, the archived platform aggregate records 1,786 learners and researchers from 215 institutions, alongside 59,146 nodes and 303,965 prerequisite links (see supplement for counting rules and archived query summary); we cite these totals only as deployment context, and all behavioral claims remain anchored to the archived 937-node course dataset. Because we do not include a randomized dense-view baseline, our empirical claims focus on how the interaction model supports deployed use in context.

2 Related Work

2.1 Graph Visualization and Progressive Disclosure

Displaying dense graphs degrades node-link readability [8, 9]. Progressive disclosure strategies—beginning with a minimal view and revealing structure on demand—avoid this by keeping visible complexity proportional to user attention. SpaceTree [17] applies this principle to trees, animating subtree expansion; van Ham and Perer's degree-of-interest model [24] extends it to general graphs with a "search, show context, expand on demand" interaction loop. Perer and van Ham [16] further argue that partial-graph views should integrate querying with browsing so users can validate local structure without rendering the full network. Elmqvist and Fekete [6] formalize hierarchical aggregation for managing visual clutter. Shneiderman's "overview first, zoom and filter, then details-on-demand" mantra [21] and Cockburn et al.'s comparison of focus+context paradigms [5] provide foundational design principles. Hogräfer et al. [10] recently combined degree-of-interest functions with progressive visualization, demonstrating continued interest in demand-driven graph disclosure.

1Cademy builds on this lineage but targets a different use case: *collaborative authoring* of a shared graph by many contributors over extended periods, rather than single-analyst exploration of a static dataset. This requires personal views atop a shared structure, mandatory structural placement of new content, and layout stability across editing sessions.

2.2 Collaborative Knowledge Platforms

Knowledge Forum [19] pioneered computer-supported knowledge building but uses threaded discussion rather than prerequisite structure. CmapTools [3] supports concept-map authoring with labeled links but presents the full graph without progressive disclosure and does not enforce prerequisite semantics. WebProtégé [23] is a highly used collaborative ontology editor that supports multi-user editing but typically displays the full ontology hierarchy rather than enforcing progressive disclosure. Wikum [29] uses staged collaborative summarization to compress discussions; 1Cademy grows by *extension* rather than compression, adding nodes and prerequisite links rather than condensing threads. Knoll [30] is a recent deployed knowledge-management system presented at UIST: it creates a knowledge ecosystem where users create, curate, and configure knowledge modules for LLM augmentation, with 200+ users. 1Cademy differs from Knoll in three ways: it manages *hierarchical prerequisite structures* rather than flat knowledge modules; it uses progressive disclosure rather than full-graph views for navigation; and its primary artifact is the collaboratively authored graph itself, not an LLM augmentation layer. Similarly, Garden of Papers [12] provides a visual, integrated workspace for organizing research papers. However, like many spatial workspaces, it emphasizes flexible, personal spatial arrangements of documents rather than enforcing a unified, shared prerequisite structure across a collaborative community. Prerequisite ordering imposes a directed dependency structure that constrains both navigation (users must traverse parents before children) and authoring (new contributions must be anchored to existing structure), creating interaction challenges absent from flat-module or freeform-spatial systems where items can be added and arranged independently.

Furthermore, while popular Personal Knowledge Management (PKM) and networked note-taking tools (e.g., Roam Research, Obsidian) utilize local graph views to navigate dense notes, they emphasize personal, bidirectional, and unstructured idiosyncratic links. 1Cademy, in contrast, enforces a shared prerequisite directed acyclic graph (DAG) with mandatory structural placement, adapting progressive disclosure for collaborative rather than purely personal curation.

2.3 Educational Prerequisite Modeling

Concept maps [15] establish prerequisite-oriented representations through human authoring. QMaps [27] provides a direct precedent for collaboratively authored prerequisite links between educational questions in a shared map. ACE [1], by contrast, studies AI-assisted construction of educational knowledge graphs by iteratively presenting candidate concept pairs for expert review. OpenDSA [20] is relevant at a different level: it is a community active-eBook project for collaboratively developing educational content rather than a system for collaborative prerequisite-graph authoring. Automated prerequisite inference via knowledge-graph mining [4, 13] relies on structured corpora, whereas 1Cademy supports the organic,

human-driven authoring of new structures without requiring pre-existing datasets. None of these systems studies persistent personal subgraph views for progressive-disclosure authoring in a live shared graph. LLMs promise to infer prerequisites from general parametric knowledge, but as our two-model audit shows (Section 6), this generality comes at the cost of structural precision. Recent work has begun exploring this direction: Reales et al. [18] use LLM-constructed knowledge graphs to identify core concepts in educational resources, and Wang et al. [25] propose a knowledge-graph-augmented LLM tutoring model that improves accuracy on complex questions. Both approaches construct or augment educational graphs with LLMs but do not address collaborative human authoring or progressive-disclosure inspection of the resulting structure.

Unlike QMaps, 1Cademy makes prerequisite placement a mandatory step inside a shared evolving graph and reveals only the local neighborhood on demand; crucially, QMaps did not evaluate progressive-disclosure navigation or personal-view compactness, which is the central empirical focus of the present paper. Unlike ACE, our focus is not reducing offline graph-construction effort in isolation, but supporting deployed contributors as they inspect and validate candidate prerequisite edges in interface context. Unlike OpenDSA, the primary artifact in 1Cademy is the collaboratively authored prerequisite graph itself rather than the surrounding course materials.

2.4 Interactive Systems for LLM-Generated Content

Recent HCI systems address structurally imprecise LLM outputs: Sensecape [22] supports multilevel exploration of LLM text hierarchies and Graphologue [11] enables interactive diagramming of LLM responses. KGScope [28], by contrast, is closer to the graph-exploration lineage: it uses embedding-guided local exploration of existing knowledge graphs rather than curating LLM-generated structure. Across these systems, the recurring interaction pattern is incremental human inspection of complex generated or retrieved structure rather than one-shot acceptance of a monolithic result. Unlike these systems, which focus on single-user sensemaking of generated text or exploration of existing graphs, 1Cademy applies bounded local inspection specifically to the collaborative authoring of a shared, evolving structure.

Taken together, these systems emphasize four concerns: knowledge workspaces and LLM augmentation; existing-graph exploration; shared prerequisite artifacts without bounded personal views; and offline candidate review. 1Cademy instead studies collaborative authoring of a shared evolving prerequisite graph under bounded local views with PPVE and deployment evidence of compact views.

Table 1 summarizes the resulting design space. The key gap is not progressive disclosure alone, but the combination of bounded local views with collaborative authoring and local inspection in a shared, evolving prerequisite graph.

3 System Design: 1Cademy

1Cademy is an interactive web-based system for collaborative authoring, browsing, and local inspection of large prerequisite graphs. Its design is organized around four principles: (1) progressive disclosure to manage graph density, revealing edges incrementally as users expand nodes [24]; (2) mental-map preservation through stable layout during state changes [14]; (3) hierarchical aggregation via a three-state node model [6]; and (4) mandatory structural placement, requiring every new node to be linked to at least one existing prerequisite parent so authoring begins from local context rather than a blank canvas.

3.1 Graph Structure and Visibility States

The shared graph is a directed acyclic graph (DAG) of *micro-topic* nodes—each constrained to a concise description rather than a full article—connected by prerequisite edges: $(A \rightarrow B)$ asserts that concept A must be understood before concept B. Nodes carry typed content (five types: Concept, Reference, Relation, Question, Code; distribution in the supplement) and metadata including authorship and timestamps.

Each node exists in one of three mutually exclusive visibility states per user: **Hidden** (default; absent from the personal view), **Collapsed** (visible as a titled card), or **Expanded** (full content visible, Hidden children promoted to Collapsed). Transitions follow four rules: (1) Hidden → Collapsed when a neighboring Expanded node reveals it or when the user searches by name; (2) Collapsed → Expanded on click; (3) Expanded → Collapsed on click; (4) Collapsed → Hidden on explicit dismissal. A typical session proceeds through seed selection, incremental expansion, search, and selective collapse.

3.2 Path-Preserving Virtual Edges

When users hide intermediate nodes, naive edge suppression fractures long-range connectivity: two visible concepts may appear unrelated even though they are connected through a chain of hidden prerequisites. Displaying all boundary edges to hidden nodes avoids this problem but inflates visual complexity by rendering stubs to invisible structure. 1Cademy resolves this tension through *Path-Preserving Virtual Edges (PPVE)*: for every pair of visible nodes connected by a directed path in the shared DAG whose intermediate nodes are all Hidden, the system renders a single virtual edge between the visible endpoints.

Formally, let $V_u \subseteq V$ be the set of nodes visible (Collapsed or Expanded) to user u. A virtual edge (a, b) is rendered if and only if $a, b \in V_u$ and there exists a directed path from a to b whose intermediate nodes are all Hidden for u. Operationally, PPVE adds a shortcut only for visible endpoint pairs connected by at least one all-Hidden intermediate path, restoring only the dependency signals that hiding would otherwise sever while leaving directly rendered visible-to-visible real edges unchanged.

Figure 2 illustrates the mechanism on a small subgraph. When all four intermediate nodes between two visible concepts are Hidden, a single virtual edge replaces the four-hop chain, reducing five rendered edges to one without losing the dependency signal.

In the archived deployment snapshot, the effect is substantial: the canonical graph contains 929 real prerequisite edges, yet the median user's visible set (19 nodes) induces far fewer real edges within it. PPVE supplements these with virtual edges so that the visible subgraph remains a faithful summary of the global reachability

Table 1: Design-space comparison. The rightmost columns indicate whether each system family supports four properties that 1Cademy combines: progressive-disclosure view management (Prog. Disc.), a shared evolving graph artifact (Shared Graph), multi-user collaborative authoring (Collab. Auth.), and bounded local edge inspection (Local Insp.). ✓ = present; – = absent.

System family	Artifact	View strategy	Prog. Disc.	Shared Graph	Collab. Auth.	Local Insp.
Progressive graph exploration [10, 17, 24]	Static tree/graph	Expand on demand	✓	–	–	–
Collaborative knowledge platforms [3, 19, 23, 29]	Shared notes/maps	Full-map or staged	–	✓	✓	–
LLM curation interfaces [11, 22]	Generated text/graph	Multilevel exploration	✓	–	–	–
Guided KG exploration [28]	Existing KG	Embedding-guided local	✓	–	–	–
Knowledge workspaces [12, 30]†	Papers/modules	Flexible workspace	–	–	–	–
Prerequisite-link maps [27]	Shared question map	Shared map view	–	✓	✓	–
Collaborative active e-books [20]	Active e-book content	Book/chapter view	–	–	✓	–
AI-assisted prereq. construction [1]	Educational KG	Pairwise expert review	–	✓	–	✓
1Cademy [this paper]	Shared prereq. DAG	Personal notebook + 3-state	✓	✓	✓	✓

†Knoll [30] reports 200+ users creating knowledge modules; we mark Collab. Auth. as – because its modules are individually curated rather than co-authored nodes in a shared graph.

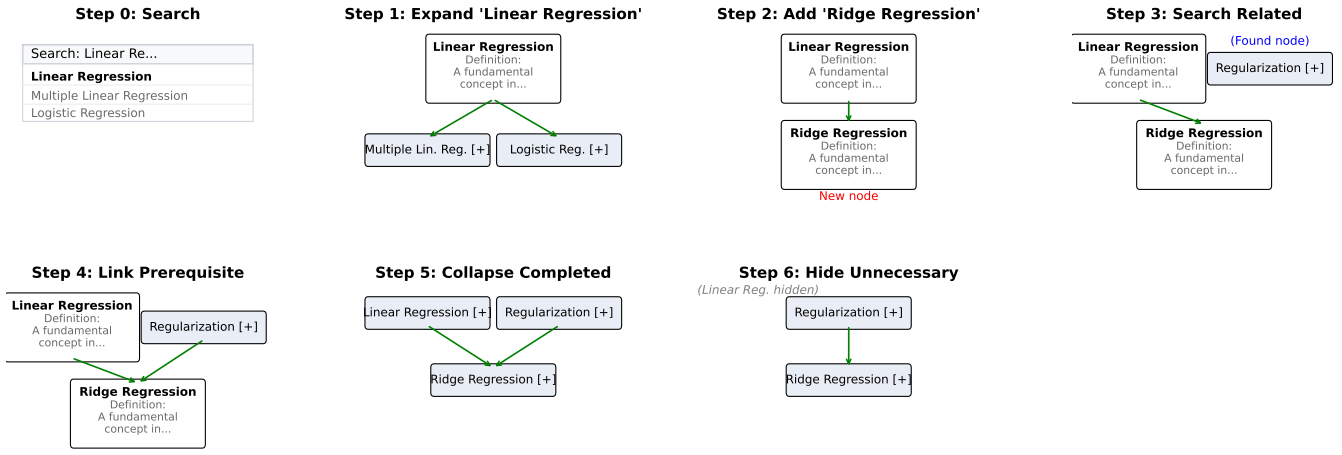


Figure 1: Scenario walkthrough of the three-state interaction model. A user adds a new concept (Ridge Regression) to a large prerequisite graph. By enforcing mandatory linking and progressive disclosure, the interface anchors the authoring task in local graph structure rather than exposing the dense full curriculum. Note that this figure shows a schematic view of these steps and is not the real UI of the system. The real system UI is available in Figure 3.

structure, not a disconnected fragment. Across users with active views, PPVE generates an average of 8.9 virtual edges per user to supplement their visible real edges. The scalability analysis in Table 2 shows an average of 44.1 real edges per user at the 937-node scale. Combining these yields roughly 53 total rendered edges per user on average—an 18× reduction compared to the 929 edges in the dense view, preserving structural coherence while suppressing visual complexity.

3.3 Personal Views

Each contributor maintains a personal view—a persistent subset of the shared graph that reflects their individual reading, authoring, and navigation history. The notebook records which nodes the user has made visible, which they have expanded, and which they have studied. Because each user's view is independent, two contributors can work on different parts of a large graph simultaneously, each seeing only the local neighborhood relevant to their current task. The shared graph is the single source of truth; personal views are lenses onto it.

3.4 Mandatory Prerequisite Linking

Every new node must be attached to at least one existing prerequisite parent before it can be proposed for inclusion in the shared graph. This constraint—the system's strongest structural requirement—serves two interaction-design functions: (1) it prevents orphan nodes from entering the authoring workflow by requiring attachment to existing graph structure; and (2) it forces the proposer to navigate the existing graph and locate the appropriate placement before writing new content, which promotes familiarity with the local neighborhood.

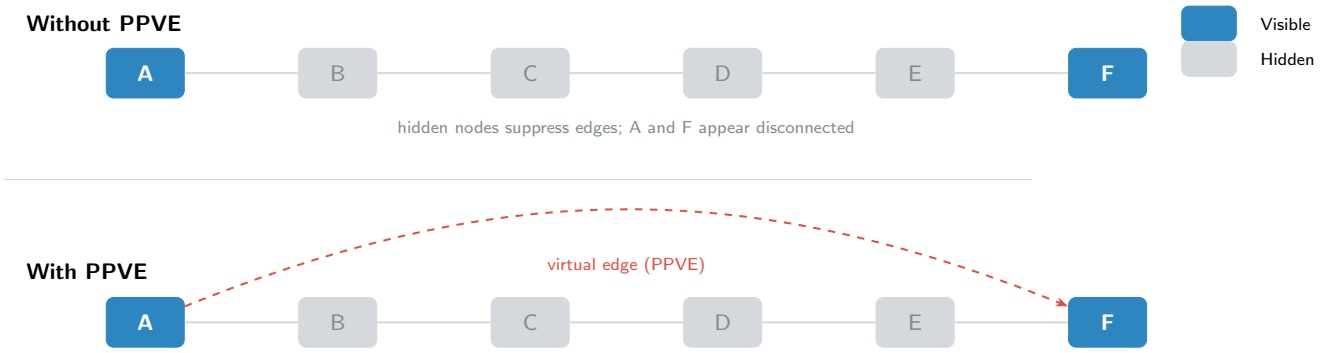


Figure 2: Path-Preserving Virtual Edges (PPVE). Top: without PPVE, hiding intermediate nodes B–E disconnects visible nodes A and F. Bottom: PPVE renders a single virtual edge from A to F, preserving reachability without displaying hidden structure.

3.5 Proposal Workflow

Contributing new content to the shared graph follows a proposal workflow that keeps authoring tied to a local neighborhood:

(1) The contributor navigates the graph (using expand and search) to locate the appropriate parent node.
(2) They draft a new micro-topic node and attach it to at least one existing prerequisite parent (mandatory linking).
(3) The proposal enters a shared queue before appearing in the common graph.
(4) Processed proposals that enter the shared structure later appear during neighborhood expansion.

This workflow applies to both new-node proposals and improvement proposals (edits to existing nodes). The distinction between new-node and improvement proposals matters for progressive disclosure: new nodes extend the graph structure that later appears during expansion, while improvements refine existing content without changing the topology.

3.6 Node Types

The graph supports five node types—Concept (69.4% of the deployment graph), Reference (17.9%), Relation (11.7%), Question (0.3%), and Code (0.6%)—so that the prerequisite structure captures not only ordering but also the pedagogical resources (references, practice items, implementations) at each node. Definitions and examples for each type are in the supplement.

3.7 Implementation

1Cademy is a web application backed by a cloud database. The client renders the graph in a top-to-bottom hierarchical layout with animated state transitions that preserve the user's mental map [14], and a search bar supports semantic search across all node titles, backed by a graph-based retrieval-augmented generation (RAG) pipeline that surfaces contextually relevant nodes. Implementation details (layout parameters, prototype artifact) are in the supplement.

4 Study 1: Deployment Analysis

We analyze interaction logs from a semester-long deployment to understand how users work with the progressive-disclosure design in practice.

4.1 Setting

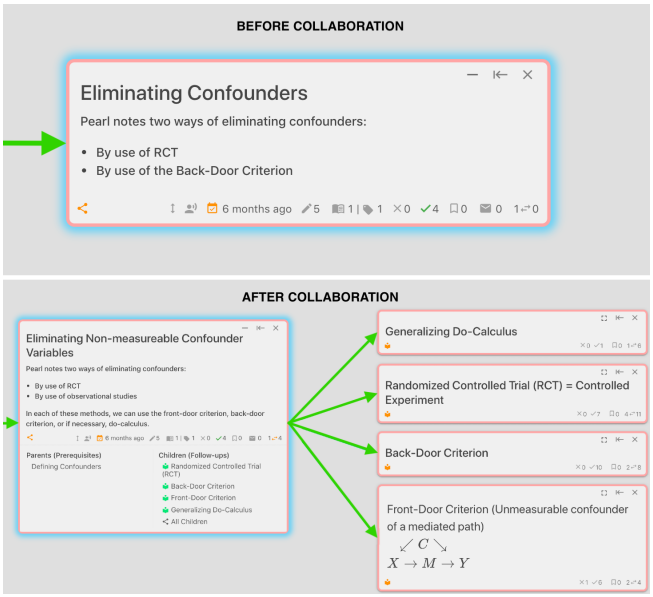
The deployment took place in a data science and machine learning course at a large public university. We define one canonical archived graph for all graph-derived statistics: 937 concept nodes and 929 deduplicated parent→child prerequisite links whose endpoints both appear in the archived node table. The raw export records 1,035 parent links overall, of which 106 point to nodes outside the archived snapshot; we use the parents field as the canonical edge source and document the full reconciliation in the supplement. Under this definition, the archived internal graph is a valid DAG (verified: 0 cycles) with a longest directed path of 16 hops, graph density 0.0011, and 75 weakly connected components.

For view and interaction analyses, we filter the per-user state table and the timestamped interaction log to rows whose node appears in the archived 937-node snapshot. This yields 48 active users and 24,337 interaction events on the archived graph. We report proposal counts separately at the semester-archive level because snapshot-filtering the proposal log undercounts new-node proposals: the full raw archive contains 2,846 proposals (353 new-node proposals and 2,493 improvement proposals) from 38 proposers. All reported numbers are traceable to frozen outputs in the artifact directory documented in the supplement. We intentionally restrict the present analysis to this one archived course export so that every reported user-state statistic is traceable to a single frozen snapshot, even though the broader platform has been used in larger multi-topic settings documented separately in the supplement.

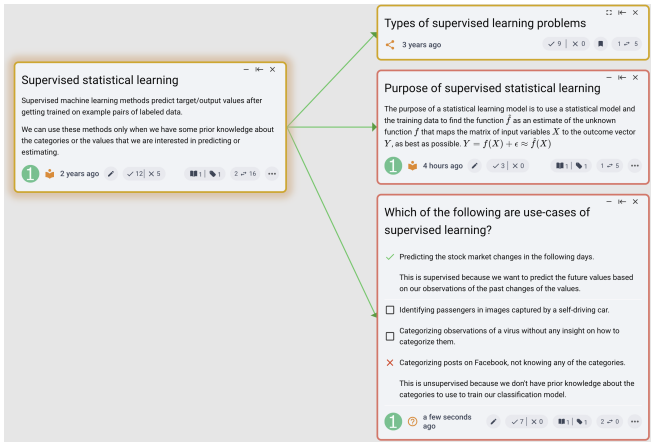
4.2 Progressive Disclosure in Practice

The central question is whether users actually work with small subsets of the graph, or whether they expand everything and defeat the progressive-disclosure design.

Personal view sizes. The median active user had only 19.0 visible nodes at the end of the semester—**2.0% of the 937-node graph** (IQR = 69.0, SD = 55.6). Figure 4 (left) shows the full distribution



(a) A 1Cademy node before and after collaborative expansion. A single concept node (*Eliminating Confounders*) gains four child nodes through contributor proposals, each adding a more specific sub-topic as a new micro-topic node. Green edges denote prerequisite links.



(b) A knowledge graph neighborhood in 1Cademy showing three node types: a Concept node (*Supervised Statistical Learning*), a child Concept node with a mathematical formula, and a Question node with answer feedback. Each node displays its content, lightweight metadata, and type indicator.

Figure 3: The 1Cademy interface. Nodes are displayed as titled cards with typed content. Prerequisite edges (green arrows) flow top-to-bottom. Each node can be independently Expanded (content visible, edges rendered) or Collapsed (title only). The personal view shows only the user's currently relevant neighborhood, not the full graph. These screenshots reflect the node view as it appeared during the study period; the 1Cademy interface has since been iteratively redesigned, and the current production UI differs in visual styling and layout from the version shown here.

after restricting the state table to the 937 archived node IDs. The mean was higher (41.9) due to a right-skewed distribution: a small number of power users maintained views of 131–230 nodes, but most active users worked with fewer than 50. Personal view sizes (visible or studied nodes) were also modest: median 32.0 nodes (3.4% of the graph). As a corroborative temporal check, we parsed the timestamped interaction log, reconstructed each user's visible set after every logged interaction, and summarized the resulting per-user working sets. This log-based reconstruction yielded a median of 15.25 visible nodes across users' within-user medians (supplement), but because 9 of 48 accounts do not reconcile exactly with the archived final-state table, we treat the final snapshot estimate as primary. Restricting to the 34 users who had at least one visible node in the final snapshot, the median rises to 41.0 visible nodes (4.4% of the graph, mean = 59.2), and the 38 active proposers represent users with confirmed ongoing engagement. The compact-view finding therefore holds even when zero-visible-node accounts—which may reflect disengagement rather than design effects—are excluded.

Expand and collapse actions. Of 24,337 interaction events from the deployment, 11,827 (48.6%) recorded a state change. Users performed 11,425 expand actions and 402 collapse actions, yielding an expand-to-collapse ratio of 28.4:1. This asymmetry is descriptive rather than causal: the archived logs record many more expansions than explicit collapses, with previously opened nodes often remaining visible in the saved state. The pattern is consistent with users carrying local context forward during navigation, but the logs do not reveal why a given node was left open.

Compactness is non-trivial. A skeptic might counter that compact views are trivially guaranteed by the default-Hidden design: since every node starts invisible, the system mechanically limits exposure. Two observations push against this reading. First, the mandatory linking constraint creates sustained counter-pressure toward expansion: every new-node proposal requires the author to navigate the existing graph and locate a placement site, forcing traversal and node reveals. Over a full semester of authoring (2,846 proposals from 38 proposers), this traversal pressure is substantial. If no user ever collapsed or hid a node, each traversal would monotonically grow the personal view. A log-based reconstruction confirms this pressure: if every node revealed to a user during the deployment remained visible, the median active user's view would have grown to 109.0 nodes. Instead, actual median view size remained sharply bounded at 19.0 nodes. Second, the 28.4:1 expand-to-collapse ratio shows that users rarely explicitly pruned their views—compactness persisted not because users actively cleaned up, but because they navigated selectively even when the system placed no ceiling on expansion. Together, these observations suggest that the bounded-view pattern reflects selective navigation

Table 2: Scalability of progressive disclosure under a deterministic breadth-first ordering of nodes. "Dense" shows the induced edge count in each prefix; "PD" shows the mean visible edges among users who had at least one visible node in that prefix. The reduction factor ranges from 15× to 30×.

Subgraph N	Dense (edges)	PD (mean vis.)	Reduction
50	49	1.6	29.9×
100	99	5.0	19.6×
200	214	13.2	16.2×
500	551	36.2	15.2×
937	929	44.1	21.1×

behavior rather than a mechanical artifact of the default-Hidden starting state.

4.3 Scalability of Progressive Disclosure

To assess how progressive disclosure scales with graph size, we construct a deterministic node order over the canonical graph. The intuition is to simulate a growing graph by adding nodes in breadth-first order that mirrors how a real prerequisite graph might expand outward from its most-connected concepts. We sort weakly connected components by size, start a breadth-first traversal in each component at its highest weak-degree node (ties broken by archived row order), concatenate those traversals, and evaluate prefixes of size 50, 100, 200, 500, and 937 nodes. The 50–500 rows therefore remain within the largest weakly connected component, whereas the 937-row is the full archived internal graph. For each prefix, "Dense" counts all induced edges and "PD" reports the mean visible edges among users who had at least one visible node in that prefix.

Table 2 reports the results. At all scales, progressive disclosure reduces the number of visible edges by an order of magnitude. At the full 937-node graph, the mean number of edges visible to an overlapping user under progressive disclosure is 44.1, compared to 929 edges in the dense view—a **21.1× reduction**. Across all sampled sizes, progressive disclosure reduces visible edges by roughly 15–30×, indicating that the interaction keeps local inspection substantially sparser than a dense all-visible rendering.

4.4 Contribution Patterns

The proposal archive from the full semester provides additional context on the scale and variety of collaborative authoring activity.

Proposal volume and types. The 38 active proposers submitted 2,846 proposals over the semester: 353 new-node proposals (12.4%) and 2,493 improvement proposals (87.6%). The 7:1 ratio shows that, within the raw semester archive, contributors recorded many more edits to existing nodes than new-node additions. We treat this mix as workflow context rather than as evidence that progressive disclosure itself caused the proposal pattern.

Contributor distribution. Proposal activity was right-skewed: the median proposer submitted 39.5 proposals (SD = 71.6), while the most active contributor submitted 251. The proposal archive includes 38 proposers overall, while the deployment logs include 48 active users, indicating that about one in five users engaged through browsing or inspection rather than submitting proposals.

Graph structure. Under the canonical parents-based graph definition, the archived graph has density 0.0011, with most nodes having exactly one prerequisite parent (median in-degree = 1.0) and 61.8% of nodes being leaves (out-degree = 0). The longest directed path is 16 hops, and the graph contains 75 weakly connected components. These components reflect both distinct topic clusters (e.g., causal inference, regression, clustering, Bayesian networks) and the fact that 106 raw parent links point to nodes outside the archived snapshot. In the archived final graph, nodes usually carry only a few prerequisite links rather than forming a densely cross-linked curriculum. That sparse structure is compatible with the micro-topic constraint and local-placement rules, but the deployment logs alone do not identify which mechanism produced it.

5 Study 2: Interview Analysis

To understand how users experience the progressive-disclosure design, we conducted semi-structured interviews with 14 individuals across 13 coded interview units under institutional human-subjects approval: 12 student interviews and one joint instructor interview involving two instructors. Each session lasted approximately 30 minutes and covered navigation strategies, prerequisite linking, review workflows, and comparison with other tools. Four researchers independently coded each transcript, iteratively refined the code definitions through consensus discussion, and then reconciled the final theme labels. We report only themes that at least two coders independently marked in the pre-consensus coding round. Because the coding followed an iterative-consensus process with discussion rounds [2] rather than fixed parallel coding, we do not report a formal inter-rater reliability coefficient; the consensus process is detailed in the supplement (Section 2.3). Participants are cited by pseudonym below.

5.1 Navigation Strategies

Participants described two primary navigation strategies enabled by the three-state model:

Anchor-then-explore. Users locate a familiar node and use it as a starting point for expansion. Miles described this pattern explicitly: *"Once you find that one note that you're familiar with, it's like having an anchor and then it helps you to explore from that."* This mirrors the "search, show context, expand on demand" loop identified by van Ham and Perer [24], but in a collaborative authoring context where the user is both reader and contributor.

Parent-first traversal. To locate the appropriate placement for a new node, users navigate upward to a broad parent concept and then drill down. Rajsi described this: *"You have to go to the parent node, like a bigger concept, and then manually look for whatever the note you were trying to create."* This strategy is consistent with mandatory prerequisite linking: because new nodes must attach to an existing parent, the interface directs contributors to inspect the local neighborhood before contributing.

5.2 Managing Visual Complexity

Participants reported that the micro-topic structure combined with progressive disclosure reduces the overwhelm associated with dense information displays:

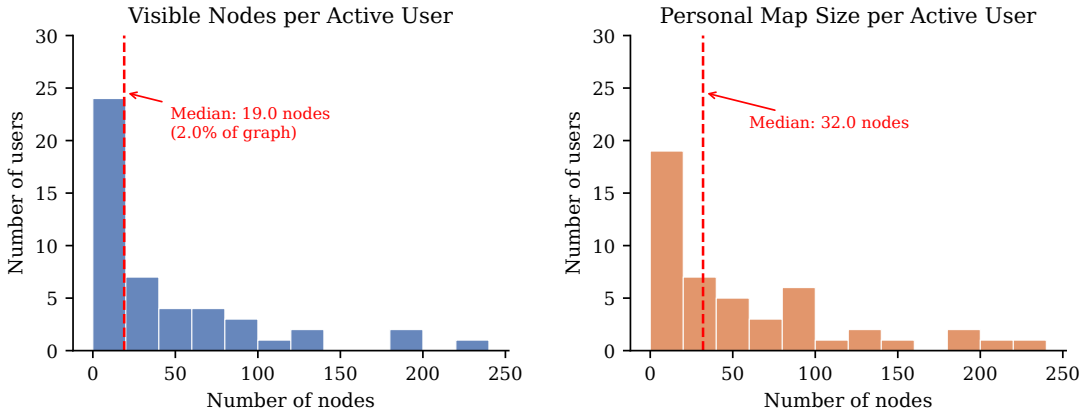


Figure 4: Distribution of personal view sizes across 48 active users after restricting the state table to the archived 937-node graph. Left: visible nodes at the end of the semester (median = 19.0, 2.0% of the graph). Right: personal view size (visible or studied nodes; median = 32.0). Most users end the archived deployment with small neighborhoods despite the graph's size.

Chunking reduces overload. Allison noted: *"Smaller doses make the information flow a lot easier."* Rajsi explicitly contrasted this with monolithic notes: *"If I just read an entire chapter of the textbook and made my own big notes, it would be a lot more overwhelming than just reading it slowly and making little nodes."*

Spatial layout aids comprehension. Emmett valued the spatial organization: *"The ability to place text on more of a map structure has been really helpful."* Columbus described the dual benefit of overview and detail: *"You see the whole and the constituent parts. It goes into detail but you get a broader scope of what the material is."*

Pain points. Users also reported navigation difficulties. Natalie described viewport disorientation in large graph regions: *"I would lose my place… the knowledge graph is so large and you can zoom really in or zoom really out. I would sometimes lose my place… that probably took up some of my time where I should have been just working."* Miles noted progress uncertainty: *"It's hard to tell… when you read a textbook, you know when it will end… with nodes it's kind of hard to know."* These pain points suggest design improvements such as minimap overlays and progress indicators.

5.3 Comparison with Alternative Tools

Several participants spontaneously contrasted 1Cademy with conventional document-based tools:

Allison distinguished spatial connectivity from linear stacking: *"On a Google Doc, information gets stacked rather than connected."* Natalie contrasted 1Cademy with simultaneous editing in document tools: *"Google Docs can get very overwhelming because everyone's just adding stuff."* An instructor participant (Chelsey) described the hide/expand mechanism as enabling customization: *"You can hide what you don't want to use, but you can expand if you want to take a different approach."*

5.4 Prerequisite Linking in Practice

A recurring theme was how the mandatory prerequisite-linking constraint shapes the contribution experience. Participants described the linking step as both a cognitive aid and a source of friction.

Linking as a learning mechanism. Rose described the linking process as forcing deeper engagement: *"You have to actually understand the material to know where it should go. You can't just dump information."* Maria noted that the process made her read existing nodes she would otherwise skip: *"I would look at the parent node and the other children to make sure I'm not duplicating what someone else already wrote."*

Linking friction. Jac described difficulty finding the right parent in dense subgraphs: *"Sometimes I know what I want to write about but I don't know where to put it. I have to search around and expand a bunch of nodes to find the right spot."* This friction is a known trade-off of mandatory structural placement: it increases the cost of contribution but reduces the risk of orphaned or misplaced content. The progressive-disclosure design mitigates this friction by allowing the contributor to expand only the relevant neighborhood rather than scrolling through the full graph.

5.5 Summary of Interview Findings

Several of these themes—anchor-then-explore navigation, chunking, and spatial comprehension benefits—are well-documented in information visualization and cognitive psychology. Their emergence here provides deployment validation that these patterns transfer to collaborative prerequisite-graph authoring, a context not previously studied, rather than claiming these as novel findings. Table 3 summarizes the major themes identified across the 13 coded interview units.

6 Formative LLM Audit

We use an LLM edge audit as a stress test for the interaction model. When an LLM proposes many plausible-but-noisy prerequisite edges, a reviewer must inspect structure locally rather than absorb a dense candidate graph at once. This section quantifies that noise through a formative audit and then analyzes how the three-state model affords a bounded local inspection workflow.

Table 3: Interview themes identified by at least two independent coders across 13 coded interview units. *n* indicates the number of transcripts in which the theme was raised; the joint instructor interview counts as one unit.

Theme	*n* (of 13 units)
Anchor-then-explore navigation	8
Chunking reduces overload	7
Spatial layout aids comprehension	6
Viewport disorientation (pain point)	5
Linking as learning mechanism	5
Progress uncertainty (pain point)	4

6.1 Formative Audit

We evaluated LLM prerequisite inference against the archived 1Cademy snapshot (937 concepts; 929 canonical internal prerequisite edges defined from deduplicated internal parent links). We used a pairwise evaluation design: for every pair of nodes whose embedding similarity (OpenAI `text-embedding-3-large`) exceeded a cosine similarity of 0.55—chosen to capture semantically related concepts while excluding clearly unrelated pairs—we asked the LLM whether concept A is a direct prerequisite for concept B. Both directions were tested for each pair (i.e., "Is A a prerequisite for B?" and "Is B a prerequisite for A?"), yielding 4,756 directional prompts covering 2,373 unique node pairs. The raw export records 1,035 parent links overall, 106 of which point outside the archived snapshot.

We evaluated two models—Gemini 3 Flash Preview and Gemini 3.1 Pro Preview—both with temperature = 0.0 and thinking level set to High. Both Flash and Pro were evaluated on all 4,756 pairs. The exact prompt is reproduced in the supplementary material.

We treat the audit as a formative stress test—intended to characterize error modes and inform interface design—rather than a benchmark leaderboard. Flash achieved $P = 0.168$, $R = 0.460$, $F_1 = 0.246$ on 426 testable ground-truth edges, proposing 1,169 VALID edges (24.6%). Pro achieved $P = 0.193$, $R = 0.603$, $F_1 = 0.292$ on 426 testable ground-truth edges, proposing 1,332 VALID edges (28.0On the 4,699 shared indices, the two models agreed on 90.8% of labels (Cohenś $\kappa = 0.76$).

6.2 Quantified Error Taxonomy

We classified each model's VALID predictions against the full 937-node curriculum graph. Classification details and frozen outputs are in the supplement.

Flash (1,169 VALID predictions):

- **True positives**: 196 edges (16.8%). The LLM correctly identified prerequisite links present in the human-authored graph.
- **False positives**: 973 edges (83.2%). The dominant failure mode is conflating topical relatedness with instructional dependency (e.g., predicting an edge from *Things to Watch Out For* to *Preventing Berkson Bias*).

Flash missed 230 of 426 testable ground-truth edges (54.0% miss rate).

Pro (1,332 VALID predictions):

- **True positives**: 257 edges (19.3%).
- **False positives**: 1,075 edges (80.7

Pro missed 169 of 426 testable ground-truth edges (39.7% miss rate).

Both models exhibit the same dominant error pattern: over 80% of predicted edges are false positives. Pro achieves higher recall than Flash (0.603 vs. 0.460) at the cost of a slightly higher VALID rate (28.0% vs. 24.6%), but the false-positive-dominant pattern is consistent across both models. This two-model consistency strengthens the motivation for human inspection of candidate edges.

6.3 Progressive Disclosure for Local Inspection

The same inspection mechanics that deployment participants used during collaborative authoring—expand a node, inspect its neighborhood, collapse when done—are directly applicable to candidate-edge review. Rather than displaying all 1,169 predicted edges simultaneously, the system can surface candidates incrementally as the reviewer expands nodes, keeping the inspection surface sparse. The scalability analysis (Section 4.3) quantifies this sparsity: at the 937-node scale, progressive disclosure yields a 21.1× reduction in real edges compared to a dense view. This workflow is design rationale motivated by the audit results, not an empirically tested feature: deployment participants did not review LLM-generated edges, and the load-based comparison does not by itself establish faster or more accurate human review.

7 Discussion

7.1 Design Lessons

The central surprise of the deployment is that personal views remained remarkably compact despite the mandatory linking constraint continuously forcing contributors to traverse the graph. The median active user's view contained only 19.0 visible nodes—2.0% of the 937-node graph—and a corroborative log-based reconstruction reported in the supplement yielded a similarly small median within-user working set (15.25 visible nodes). One might expect mandatory linking to push views toward expansion, because every new node requires the author to locate a placement site in the shared structure. Instead, the 21.1× real-edge reduction factor (Table 2) and the interview evidence that users valued selective hiding of unneeded content indicate that the interface effectively allows users to prune context and maintain focus even under this authoring pressure. This tension—traversal pressure that does not produce persistent view growth—anchors the three design lessons below.

Personal views keep graphs manageable. The compact-view finding is not merely a side effect of progressive disclosure; it persists despite the authoring rule that creates counter-pressure to expand. To our knowledge, no directly comparable metric for personal-view compactness exists in prior collaborative knowledge-platform studies; CmapTools and WebProtégé do not report per-user visible-subset fractions. The 2% figure is therefore a first data point rather than a demonstrated advantage over alternatives; its value lies in showing that mandatory linking pressure did not erode view compactness in this deployment.

As detailed in the non-trivial-compactness analysis (Section 4), this result is not a mechanical artifact of the default-Hidden design: the mandatory linking constraint creates sustained counter-pressure toward expansion, yet views remained small through selective navigation rather than active pruning.

Mandatory structural placement helps anchor authoring in local context. The parent-first traversal strategy reported in interviews aligns with the prerequisite-linking constraint: contributors described navigating to a broader parent before drilling down to a placement site. The system therefore asks contributors to start from an existing neighborhood rather than an unbounded canvas.

Navigation pain points suggest concrete improvements. Viewport disorientation (Natalie) and progress uncertainty (Miles) are known challenges in large graph interfaces [5]. Minimap overlays, breadcrumb trails, and path-length indicators could address these without abandoning the progressive-disclosure model.

7.2 Design Implications

The deployment evidence and interview findings suggest four actionable design implications for systems that support collaborative authoring of large structured artifacts. These implications extend beyond educational prerequisite graphs to other shared structures in which contributors must inspect and extend local dependencies without rendering the full artifact at once, including collaborative ontology editors, shared argument maps, prerequisite planners, research concept workspaces, and review interfaces for machine-suggested graph edges.

DI1: Personal views are a strong default once a shared graph reaches this scale. The fact that the median active user retained only 2.0% of the graph as visible at the end of the semester, together with the corroborative temporal reconstruction reported in the supplement, suggests that personal views should be treated as a first-class feature for collaborative graph authoring at this scale. Systems that default to showing the full graph and expect users to manually filter may overwhelm contributors as the artifact grows. The three-state model provides a principled mechanism for this: every node starts Hidden and must be explicitly revealed, ensuring that the default experience is manageable.

DI2: Mandatory structural placement can ground contribution in local context. The interface requires a parent link at submission time, and interviews describe authoring that begins from nearby structure rather than from a blank canvas. The proposal archive indicates that this workflow was exercised at semester scale, but not its fine-grained navigation mechanics; the constraint operates at contribution time and is intended to keep the interaction focused on a bounded neighborhood. This is relevant for any collaborative system where structural coherence matters (e.g., collaborative ontology construction, shared argument maps).

DI3: Asymmetric expand/collapse ratios are consistent with users carrying passive context forward. The 28.4:1 expand-to-collapse ratio is consistent with users sometimes leaving previously expanded nodes visible as passive context rather than explicitly pruning the view after each step. The logs do not identify whether that pattern was deliberate, habitual, or simply a byproduct of the saved state, but it still motivates testing auto-collapse or aging mechanisms that gradually reduce stale visual clutter without requiring explicit user action.

DI4: Interfaces for LLM-generated graph structure should foreground local structural inspection. The two-model error taxonomy shows that the dominant LLM failure mode is structural (83.2% false positives for Flash, 80.7% for Pro) rather than factual, with substantial cross-model agreement ($\kappa = 0.76$). Inspection interfaces for LLM-generated graph structure should therefore emphasize path-level context (showing intermediate nodes between endpoints) rather than only node-level content inspection. In 1Cademy, progressive disclosure offers one way to surface that path context by revealing the local neighborhood when a candidate edge is inspected.

7.3 Future Work

The clearest next step is a controlled within-subjects comparison between the progressive-disclosure interface and a dense-graph baseline on concept finding, candidate-edge review, and node addition, with time, error, and workload measures. Additional robustness work could compare multiple LLMs on the same evaluation subset and examine how personal views change over a semester as contributors gain experience.

7.4 Limitations

Deployment scope. The empirical analyses come from a single archived course deployment. While the broader 1Cademy platform has been used in additional multi-topic settings (documented in the supplement), we do not generalize the bounded-view or interview findings beyond the archived dataset. At 937 nodes, the deployment graph is orders of magnitude smaller than industrial knowledge graphs; the BFS-prefix scalability analysis (Table 2) provides encouraging extrapolation trends within the observed range, but whether the bounded-view pattern persists at substantially larger scales remains an open question.

No controlled comparison. The deployment analysis shows that users work with small personal views, but it does not include a controlled comparison against a dense-graph baseline with random assignment. Users in the deployment always had progressive disclosure available; we cannot directly attribute the observed navigation strategies to the three-state model versus other factors (e.g., motivation, task design). A controlled within-subjects experiment comparing progressive disclosure against a dense-graph view on standardized graph tasks would provide stronger causal evidence.

Baseline scope. Our current dense-view comparison is load-based rather than task-based: Table 2 contrasts visible-edge counts against a dense all-visible rendering, but we do not report a controlled human comparison for concept finding, candidate-edge review, or node addition.

LLM audit scope. The formative audit evaluates 2,373 semantically similar node pairs (4,756 directional prompts) covering the full 937-node graph. Both models have complete coverage on all 4,756 pairs. The pairwise evaluation with a similarity threshold cannot discover prerequisite edges between dissimilar nodes, and the results may not generalize to other LLMs or longer prompt contexts. Both models exhibit the same false-positive-dominant

pattern, providing two-model consistency for the structural error characterization.

Interview transferability. The 13 coded interview units were conducted with users of a specific system in a specific educational context. The navigation strategies and complexity-management themes we report may not transfer to users with different expertise levels or to prerequisite graphs in other domains.

8 Conclusion

We presented 1Cademy as a bounded-local-view interaction model for collaborative prerequisite-graph authoring: persistent personal views, mandatory structural placement, and PPVE keep local neighborhoods legible without severing dependency context. The central finding is that this mechanism combination did not erode view compactness: despite sustained traversal pressure, the median contributor ended the semester with only 19 visible nodes, far below the 109-node no-hiding reconstruction. For collaborative prerequisite graphs at roughly this scale, contributors need not see the full structure by default; bounded personal views can keep authoring and inspection local while preserving the dependency meaning needed for editing. The next step is to test whether the same bounded-view pattern persists at larger scales through a controlled comparison against a dense-view baseline on concept finding, candidate-edge review, and node addition.

Availability

The archived course exports are not redistributable; the submission reports derived aggregates, and the supplement describes the analysis scripts and frozen artifacts.

References

[1] Mehmet Cem Aytekin and Yücel Saygın. 2024. ACE: AI-Assisted Construction of Educational Knowledge Graphs with Prerequisite Relations. *Journal of Educational Data Mining* 16, 2 (2024), 85–114. doi:10.5281/zenodo.14250896

[2] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[3] Alberto J. Cañas, Greg Hill, Roger Carff, Niranjan Suri, James Lott, Tom Eskridge, Gloria Gómez, Mario Arroyo, and Rodrigo Carvajal. 2004. CmapTools: A Knowledge Modeling and Sharing Environment. In *Concept Maps: Theory, Methodology, Technology, Proceedings of the First International Conference on Concept Mapping*. Universidad Pública de Navarra, Pamplona, Spain, 125–133.

[4] Penghe Chen, Yu Lu, Vincent Wenchen Zheng, Xiyang Chen, and Boda Yang. 2018. KnowEdu: A System to Construct Knowledge Graph for Education. *IEEE Access* 6 (2018), 31553–31563. doi:10.1109/ACCESS.2018.2839607

[5] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. 2009. A Review of Overview+Detail, Zooming, and Focus+Context Interfaces. *Comput. Surveys* 41, 1 (2009), 1–31. doi:10.1145/1456650.1456652

[6] Niklas Elmqvist and Jean-Daniel Fekete. 2010. Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines. *IEEE Transactions on Visualization and Computer Graphics* 16, 3 (2010), 439–454. doi:10.1109/TVCG.2009.84

[7] George W. Furnas. 1986. Generalized Fisheye Views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 16–23. doi:10.1145/22627.22342

[8] Mohammad Ghoniem, Jean-Daniel Fekete, and Philippe Castagliola. 2004. A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations. In *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, Piscataway, NJ, USA, 17–24. doi:10.1109/INFVIS.2004.1

[9] Ivan Herman, Guy Melançon, and M. Scott Marshall. 2000. Graph Visualization and Navigation in Information Visualization: A Survey. *IEEE Transactions on Visualization and Computer Graphics* 6, 1 (2000), 24–43. doi:10.1109/2945.841119

[10] Marius Hogräfer, Dominik Moritz, Adam Perer, and Hans-Jörg Schulz. 2023. Combining Degree of Interest Functions and Progressive Visualization. In *2023 IEEE Visualization and Visual Analytics (VIS)*. IEEE, Melbourne, Australia, 251–255. doi:10.1109/VIS54172.2023.00059

[11] Peiling Jiang, Jude Rayan, Steven P. Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 1–20. doi:10.1145/3586183.3606737

[12] Donghyeok Ma, Hanbee Jang, Joon Hyub Lee, and Seok-Hyung Bae. 2025. Garden of Papers: Finding, Reading, and Organizing Research Papers in a Visual, Integrated, and Flexible Workspace. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology* (Busan, Republic of Korea) *(UIST '25)*. Association for Computing Machinery, New York, NY, USA, Article 89, 15 pages. doi:10.1145/3746059.3747637

[13] Ruben Manrique, Bernardo Pereira, and Olga Mariño. 2019. Exploring Knowledge Graphs for the Identification of Concept Prerequisites. *Smart Learning Environments* 6, 1 (2019), 1–18. doi:10.1186/s40561-019-0104-3

[14] Kazuo Misue, Peter Eades, Wei Lai, and Kozo Sugiyama. 1995. Layout Adjustment and the Mental Map. *Journal of Visual Languages & Computing* 6, 2 (1995), 183–210.

[15] Joseph D. Novak and Alberto J. Cañas. 2008. *The Theory Underlying Concept Maps and How to Construct and Use Them.* Technical Report IHMC CmapTools 2006-01 Rev 01-2008. Florida Institute for Human and Machine Cognition.

[16] Adam Perer and Frank van Ham. 2011. *Integrating Querying and Browsing in Partial Graph Visualizations.* Technical Report. IBM Research.

[17] Catherine Plaisant, Jesse Grosjean, and Benjamin B. Bederson. 2002. SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. In *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, Piscataway, NJ, USA, 57–64. doi:10.1109/INFVIS.2002.1173148

[18] Daniel Reales, Ruben Manrique, and Christian Grévisse. 2024. Core Concept Identification in Educational Resources via Knowledge Graphs and Large Language Models. *SN Computer Science* 5, 8 (2024), 1029. doi:10.1007/s42979-024-03341-y

[19] Marlene Scardamalia and Carl Bereiter. 2006. Knowledge Building: Theory, Pedagogy, and Technology. In *The Cambridge Handbook of the Learning Sciences*, R. Keith Sawyer (Ed.). Cambridge University Press, Cambridge, UK, 97–118.

[20] Clifford A. Shaffer, Ville Karavirta, Ari Korhonen, and Thomas L. Naps. 2011. OpenDSA: Beginning a Community Active-eBook Project. In *Proceedings of the 11th Koli Calling International Conference on Computing Education Research*. ACM, New York, NY, USA, 112–117. doi:10.1145/2094131.2094154

[21] Ben Shneiderman. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*. IEEE, Piscataway, NJ, USA, 336–343. doi:10.1109/VL.1996.545307

[22] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 1–18. doi:10.1145/3586183.3606756

[23] Tania Tudorache, Csongor Nyulas, Natasha F Noy, and Mark A Musen. 2013. WebProtégé: A collaborative ontology editor and knowledge acquisition tool for the web. *Semantic web* 4, 1 (2013), 89–99.

[24] Frank van Ham and Adam Perer. 2009. "Search, Show Context, Expand on Demand": Supporting Large Graph Exploration with Degree-of-Interest. *IEEE Transactions on Visualization and Computer Graphics* 15, 6 (2009), 953–960. doi:10.1109/TVCG.2009.108

[25] Yurong Wang, Fei Zhang, Ruijun Wang, Yuanbo Yuan, and Wenbin Li. 2025. Synergizing Knowledge Graphs and LLMs: An Intelligent Tutoring Model for Self-Directed Learning. *Education Sciences* 15, 9 (2025), 1102. doi:10.3390/educsci15091102

[26] Iman Yeckehzaare. 2021. *Harnessing Micro-Topics Arranged in Learning Pathways for Spaced Retrieval, Reading, and Collaborative Note-taking.* Ph. D. Dissertation. University of Michigan.

[27] Iman Yeckehzaare, Tirdad Barghi, and Paul Resnick. 2020. QMaps: Engaging Students in Voluntary Question Generation and Linking. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 1–14. doi:10.1145/3313831.3376882

[28] Chao-Wen Hsuan Yuan, Tzu-Wei Yu, Jia-Yu Pan, and Wen-Chieh Lin. 2024. KGScope: Interactive Visual Exploration of Knowledge Graphs with Embedding-Based Guidance. *IEEE Transactions on Visualization and Computer Graphics* 30, 12 (2024), 7702–7716. doi:10.1109/TVCG.2024.3360690

[29] Amy X. Zhang, Lea Verou, and David R. Karger. 2017. Wikum: Bridging Discussion Forums and Wikis Using Recursive Summarization. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, New York, NY, USA, 2082–2096. doi:10.1145/2998181.2998235

[30] Dora Zhao, Diyi Yang, and Michael S. Bernstein. 2025. Knoll: Creating a Knowledge Ecosystem for Large Language Models. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology* (Busan, Republic of Korea) *(UIST '25)*. Association for Computing Machinery, New York, NY, USA, Article 140, 23 pages. doi:10.1145/3746059.3747711