2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION
Paper ID #53563

Problem Generation to Personalized Tutoring: Leveraging Generative AI and Knowledge Components in Engineering Education

Mr. Luciano Bermudez, University of California, Riverside

Problem Generation to Personalized Tutoring: Leveraging Generative AI and Knowledge Components in Engineering Education

1. Introduction

Engineering education places heavy emphasis on problem solving as the primary mechanism for learning. Master of engineering concepts requires repeated exposure to a diverse range of problems to develop conceptual understanding and procedural fluency. However producing large volumes of high-quality practice problems that are both correct and pedagogically aligned is a time-consuming task for instructors [1]. This challenge is compounded at higher levels in engineering education, where problem solving involves multi-step reasoning and mathematical proficiency [2].

1.1 Adaptive Problems and Their Limitations

Adaptive problems, in which numerical values or problem context change across attempts have been proposed as a partial solution [2]. Such problems reduce solution memorization and discourage copying while enabling students' repeated practice [2]. However adaptation at the surface level alone, such as parameter variation, does not address deeper learning challenges. Students frequently struggle not because of insufficient exposure to problems, but because they lack one or more pre-requisite concepts or mathematical skills required to successfully solve the problem [2].

This issue is pronounced in upper division mechanical engineering courses, such as Transport Phenomenon, where topics integrate concepts from fluid mechanics, thermodynamics, and heat transfer. These courses rely on a dense network of pre-requisite knowledge including both conceptual understanding (e.g. conservation principles) and procedural competence (e.g. vector calculus and differential equations). When pre-requisite gaps are present, additional practice alone may reinforce confusion rather than promote learning [2]. Thus scalable practice systems must account for prerequisite knowledge and learning progression.

1.2 Knowledge Components and Prerequisite Modeling

Prior work has sought to formalize prerequisite structure through the use of Knowledge Components (KC) defined as atomic units of knowledge,skills, concepts, or procedures that a learner must master in order to solve a problem or perform a task within a given domain [2]. Decomposing complex tasks into KCs enables instructional systems to understand what needs to be learned first, track student mastery, and provide targeted feedback [3].

Despite their theoretical effectiveness, the construction of KC models has traditionally relied on extensive manual effort from domain experts. This process is labor-intensive, highly domain-specific, and difficult to maintain as course content evolves, particularly in advanced engineering disciplines where prerequisite structures are dense and tightly coupled [3]. As a result, KC-based systems are rarely deployed at scale in upper-division engineering courses [3].

1.3 Large Language Models for Knowledge Component Extraction

Recent work has begun to explore the use of large language models (LLMs) to mitigate the cost and scalability limitations of manual KC construction. In particular, prior studies have demonstrated the feasibility of using LLMs to automatically identify and assign knowledge components to educational tasks, significantly reducing the need for manual KC authoring [3]. These results suggest that LLMs can play a meaningful role in scalable KC discovery and annotation. In parallel, Microsoft has introduced the GraphRAG indexing framework, which provides a data pipeline and transformation suite designed to extract structured, semantically meaningful representations from unstructured text using LLMs [4]. While originally proposed to support graph-enhanced retrieval, this framework also enables the construction of knowledge graphs, graph-structured models used to represent concepts and their relationships [4].

1.4 Entities, Relationships, and Communities in Knowledge Graphs

A knowledge graph represents structured knowledge using entities, relationships and communities. Entities are represented as nodes in a knowledge graph [5], they represent distinct concepts, objects or ideas explicitly present in the source material, such as concepts, laws, equations or skills. A relationship defines how entities are connected, representing a semantic link between two entities, such as dependency, usage or conceptual association and captures how knowledge components relate to one another [5]. A community is a higher level grouping of entities identified through graph analysis. Unlike entities and relationships, communities are not directly extracted from the source material. Instead, they emerge from patterns of connectivity within the graph and represent clusters of closely related entities, often corresponding to broader thematic or topical regions.

1.5 Approach

This work introduces a structured, learning-oriented approach to knowledge extraction that explicitly models educational structure and prerequisite relationships. We present an LLM-driven pipeline for constructing educational knowledge graphs from lecture materials, in which core concepts, governing laws, equations, mathematical skills, prerequisites, and knowledge components are represented as first-class entities and connected through learning-relevant relationships.

We apply this approach to Transport Phenomena, an upper-division mechanical engineering course characterized by dense and interdependent prerequisite knowledge. Through this case study, we demonstrate how large language models can be leveraged to generate educational content and to construct a learning pathway required to understand the engineering material.

2.0 Methods

2.1 Overview of Pipeline

The proposed pipeline operates on lecture materials from an upper-division Transport Phenomena course. The source content consists of scanned, handwritten lecture notes that primarily emphasize mathematical derivations, governing equations, and formal definitions. These notes reflect the typical instructional style of the course. Figure 1 illustrates a representative subset of the lecture images used in this study.

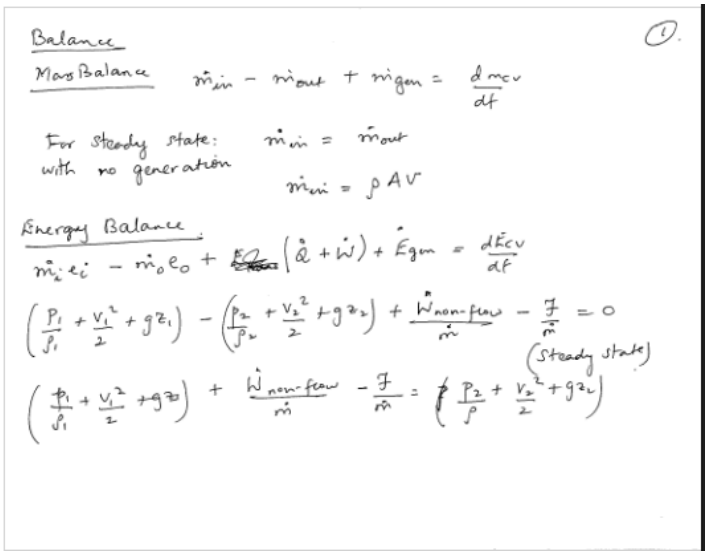

Figure 1: Transport Phenomena Lecture Snippet

To transform the raw lecture notes into structured, interpretable representations, we use a set of LLM-based extraction chains designed to produce structure outputs. Prompts are tuned based on the goal of the extraction, and structure output allows us to pass additional constraints to the large language model enabling responses to be returned in predefined schemas rather than unstructured text [6].

2.2 Data Sources

The primary data source for this study consists of handwritten lecture notes from an upper-division Transport Phenomena course. A total of 28 lectures were processed, with a mean page count of 6. The notes are provided as scanned images and reflect a derivation-heavy instructional format, with an emphasis on mathematical development, governing equations, and formal definitions.

2.3 Lecture Content Preprocessing

Four specialized extraction chains are constructed to capture different aspects of the instructional content.

- **Lecture Analysis Chain:** Extracts pedagogical metadata, including summaries, core concepts, learning objectives, and assumed prerequisites.
- **Derivation Extraction Chain:** Tailored to the derivation-heavy nature of the course, this chain extracts mathematical derivations from lecture notes and formats symbolic expressions using LaTeX. Page-level references are preserved to maintain traceability to the source material.
- **Question Generation Chain:** Generates conceptual questions grounded in the lecture material.
- **Question Extraction Chain:** Identifies worked examples or explicit questions embedded within the lecture notes and extracts them in structured form when present.

As mentioned previously, each chain contains its own prompt and structured schema. For instance the lecture analysis chain schema is shown in Table 2

Field	Type	Description
lecture_title	String	Concise title summarizing the main focus of the lecture
lecture_summary	String	High-level pedagogical summary of the lecture
core_topics	List[String]	Primary concepts covered (e.g., Bernoulli equation)
learning_objectives	List[String]	Action-oriented learning goals (e.g., derive, explain)
assumed_prerequisites	List[String] (optional)	Prior knowledge assumed by the lecture

lecture_type	Enum (optional)	Conceptual, derivation-heavy, computational, or mixed

Table 2: Structured Output of Lecture Analysis Chain

Field	**Type**	**Description**
derivation_title	String	Concise title describing the focus of the derivation
derivation_stub	String	Statement of the equation or relationship being derived
steps	List[String]	Ordered logical or mathematical derivation steps
reference	Start page, end page	Page range in the lecture material where the derivation appears

Table 3: Structured Output of Derivation Extraction Chain

Field	Type	Description
question	String	The conceptual question being asked
topics	List[String]	Key concepts addressed by the question
options	List[Option]	Multiple-choice answer options
reference	Start page, end page	Page range where the concept or question originates
explanation	String	Concise explanation of the correct answer

Table 4: Structured Output of Question Generation Chain

Field	**Type**	**Description**
question_text	String	Raw question text extracted

		from the source material
question_type	Enum	Detected question type (conceptual, computational, derivation, multiple-choice)
topics	List[String]	Detected topics, keywords, or concepts associated with the question
options	List[Options] (optional)	Answer options if the question is multiple-choice
reference	Start page, end page	Location in the lecture material where the question appears
solution	List[String]	Ordered solution steps or explanation corresponding to the question

Table 5: Structured Output of Question Extraction Chain

These extraction chains are executed in parallel and composed into a unified pipeline, enabling batch processing of multiple lectures within a single run. Figure 3 provides an overview of the multi-chain execution workflow. For classification and orchestration within the pipeline, we use GPT-5-mini.

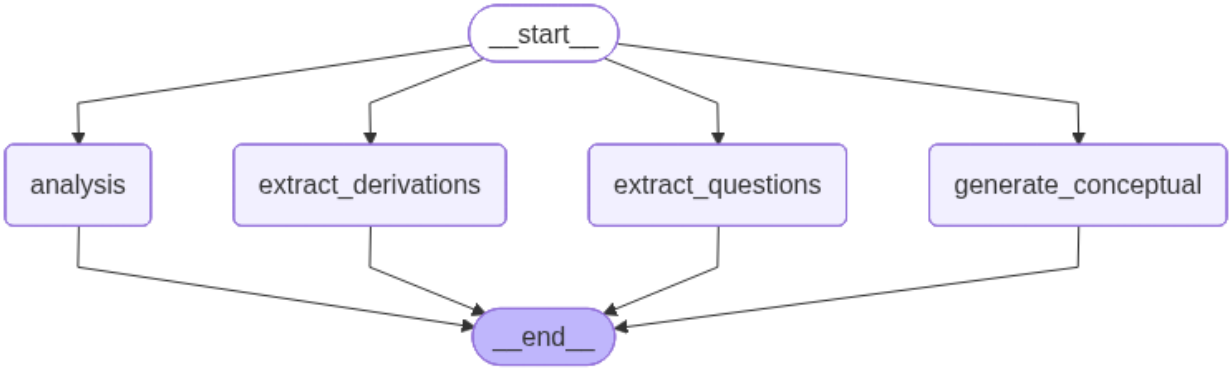

Figure 3: Lecture extraction pipeline

Across 28 lecture-processing runs, the system successfully generated structured outputs for all lectures. Due to the asynchronous and batched nature of the underlying API calls, token

usage and cost metrics were reliably captured for a subset of 16 lectures. Within this subset, the mean latency per lecture was 295.23 seconds, with an average token usage of 49,091 tokens and a mean cost of $0.0289 per lecture. These costs are incurred once per lecture as an offline preprocessing step. A full summary of the computational statistics is provided in Table 6.

Metric	**Mean**	**Min**	**Max**
Latency (s)	295.23	109.00	360.58
Tokens per lecture	49,091.13	11,160	72,957
Cost per lecture (USD)	0.028890	0.005945	0.043348

Table 6. Summary of Computational Performance Metrics for Lecture Processing

2.4 Knowledge Graph Construction

The outputs from each extraction chain (Tables 2–5) are aggregated into a single structured JSON payload, which is then flattened into text representations for analysis. These text artifacts form the input to the next stage of the pipeline, where Microsoft's GraphRAG framework is applied [4].

For knowledge graph construction, we use the standard GraphRAG indexing pipeline, which relies on a large language model to perform all reasoning and extraction tasks. The model first extracts entities and relationships from the text. Entities are represented by a title, type, and description, while relationships are represented by a source entity, target entity, and a descriptive label [5]. After extraction, entity and relationship descriptions are consolidated into single concise summaries. Finally, community detection is performed using the Hierarchical Leiden algorithm, which clusters the graph into communities until a predefined community-size threshold is reached.

2.5 Domain-Specific Optimization of the GraphRAG Pipeline

While the standard GraphRAG pipeline is general-purpose, improved results are achieved by customizing the prompt to explicitly emphasize educational structure, prerequisite relationships, and mathematical competence. The customized prompt instructs the language model to identify and represent:

- **Conceptual Foundations:** Core concepts, laws, definitions, and organizing principles

- **Prerequisite Knowledge:** Concepts that must be understood prior to engaging with the material, including explicit dependency relationships.
- **Mathematical Skills:** Procedural knowledge required for problem solving, including algebraic manipulation, calculus operations, vector and scalar reasoning, dimensional analysis, and symbolic simplification.

Finally to support this educational focus, entity extraction is constrained to the following types: **CONCEPT**, **LAW**, **EQUATION**, **MATHEMATICAL_SKILL**, and **PREREQUISITE**. Restricting extraction to these entity types enables the resulting knowledge graph to explicitly encode both conceptual understanding and procedural readiness, which are critical for modeling learning dependencies in upper-division engineering courses.

3. Results

Across the full lecture set the knowledge graph construction process identified 2984 unique entities, 3185 relationship instances, and 639 communities. The presence of a large number of communities suggests meaningful thematic clustering across lecture content, reflecting distinct but interrelated topic areas within the course. An example of the extracted entities, relationships and communities are found in Table 7,8,9 respectively.

Title	Type	Description
FLUID MECHANICS	CONCEPT	The branch of engineering and physics that studies the behavior of fluids (liquids and gases) and the forces on them; provides the overall domain context for pipe flow, conservation laws, and transport phenomena
CYLINDRICAL COORDINATES	PREREQUISITE	Coordinate system (r, θ, x) appropriate for circular pipes; knowledge of differential operators (gradient, divergence, Laplacian) in cylindrical coordinates is required to write and simplify the Navier–Stokes/ momentum equations

Table 7: Example Educational Entities Extracted from Lecture Content

Source Entity	Target Entity	Description
Fluid Mechanics	Derivation-Based Problem Solving	Following These Derivations Trains Students In Mechanistic Thinking Central To Fluid Mechanics: Reducing Pdes With Justified Assumptions, Solving Resulting Equations, And Interpreting Physical Meaning—This Is The Intended Educational Outcome Of The Lecture
Cylindrical Coordinates	Axial Momentum Equation In Cylindrical Coordinates	Familiarity With Differential Operators In Cylindrical Coordinates Is Necessary To Write The Axial Momentum Equation In The Correct Form (Including The 1/R D/Dr(R Du/Dr) Operator); This Mathematical Framework Underpins The Derivation

Table 8: Example Educational Relationship Extracted From Lecture Content

Title	Summary
Velocity Triangle Relations, Geometry, And Trigonometric Prerequisites	This Community Groups Four Tightly Related Instructional/Technical Concepts: Geometric And Blade Dimensions (Including Blade Angle B), Velocity-Triangle Relations, The Absolute Velocity And Its Components, And The Trigonometric Relations Needed To Convert Between Components And Magnitudes. The Relationships Show A Clear Dependency Chain In Which Geometry (Blade Angle) Supports Applying Trigonometry, Which In Turn Enables The Velocity-Triangle Relations That Define V, V_n, And V_θ; Those Velocity Components Are The Quantities Used In Subsequent Work/Energy Calculations (E.G., Euler's Equation). The Data Records Explicitly Link These Concepts And

	Indicate Procedural And Conceptual Prerequisites Between Them [Data: Entities (542, 543, 544, 554); Relationships (562, 563, 565)].

Table 9: Example Educational Communities Extracted from Lecture Content

3.1 Lecture Content Extraction

The extraction pipeline converted the raw lecture material into structured Markdown representations which preserve both mathematical and explanatory content. Figure 4,5,6 presents a representative example showing a lecture snippet containing a mathematical derivation alongside the corresponding rendered Markdown output. The extracted result maintains the step-by-step structure of the derivation, including intermediate equations and supporting text.

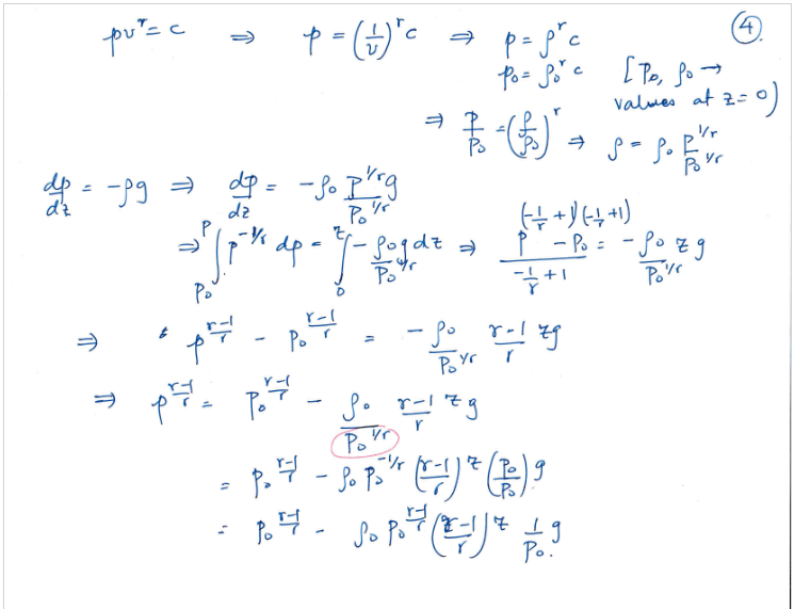

Figure 4: Lecture Derivation Part 1

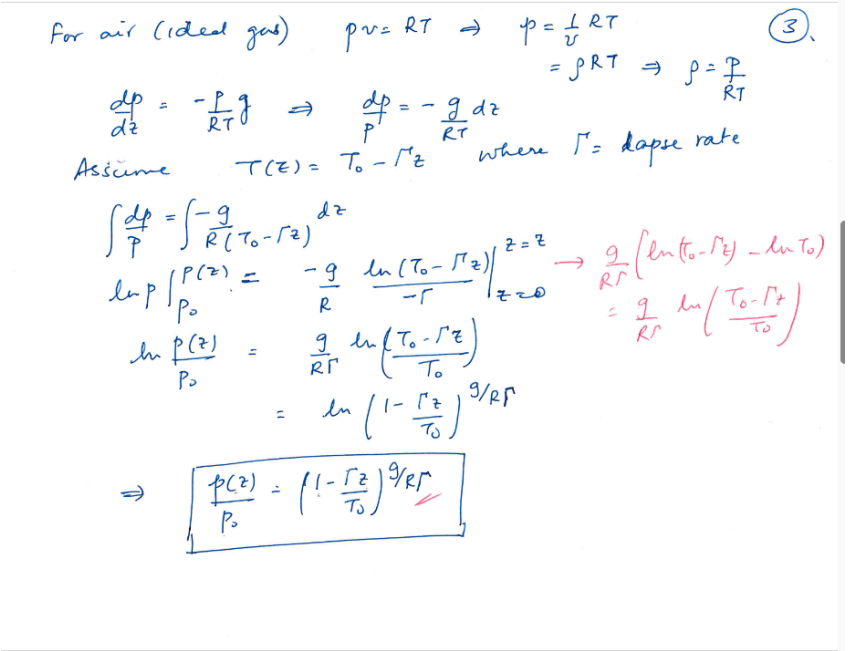

Figure 5: Lecture Derivation Part 2

Derivations

Pressure variation with altitude (barometric formula with linear lapse)

Stub: Derive p(z)/p0 = \left(1 - \Gamma z / T_0\right)^{g/(R\Gamma)} from hydrostatic balance and an assumed linear temperature lapse (complete)

Steps:

1. Start from a vertical force balance on a small rectangular parcel (hydrostatic equilibrium). Summing forces in z:

$$-p(x,y,z+\tfrac{\Delta z}{2})\,\Delta x\Delta y + p(x,y,z-\tfrac{\Delta z}{2})\,\Delta x\Delta y - \rho\,\Delta x\Delta y\,\Delta z\,g = 0.$$

2. Taylor-expand top and bottom pressures about z:

$$p\left(z+\tfrac{\Delta z}{2}\right) \approx p(z) + \frac{\partial p}{\partial z}\frac{\Delta z}{2}, \qquad p\left(z-\tfrac{\Delta z}{2}\right) \approx p(z) - \frac{\partial p}{\partial z}\frac{\Delta z}{2}.$$

3. Substitute expansions into force balance, cancel common terms, divide by volume and take the limit (\Delta z\to0) to obtain the differential hydrostatic equation:

$$\frac{\partial p}{\partial z} = -\rho g.$$

4. Use the ideal gas relation for air:

$$p = \rho R T \quad \Rightarrow \quad \rho = \frac{p}{RT}.$$

Substitute into hydrostatic equation to get

$$\frac{1}{p}\frac{dp}{dz} = -\frac{g}{RT(z)}.$$

5. Assume a linear temperature profile (constant lapse rate):

$$T(z) = T_0 - \Gamma z \quad (\Gamma = \text{lapse rate}).$$

Integrate from z=0 (p=p_0) to z:

$$\int_{p_0}^{p(z)} \frac{1}{p}\,dp = -\int_0^z \frac{g}{R\,(T_0 - \Gamma z')}\,dz'.$$

6. Carry out the integral:

$$\ln\frac{p(z)}{p_0} = \frac{g}{R\Gamma}\ln\frac{T_0 - \Gamma z}{T_0}.$$

7. Exponentiate to obtain the barometric formula with linear lapse:

$$\boxed{\frac{p(z)}{p_0} = \left(1 - \frac{\Gamma z}{T_0}\right)^{g/(R\Gamma)}}.$$

8. Comment on domain/assumption: this expression requires (T_0-\Gamma z>0) (small enough z) and assumes constant lapse rate (\Gamma) and ideal-gas behaviour. The derivation as shown on the slides is complete under these assumptions.

Figure 6: Rendered Markdown

3.2 Visualization

Gephi was used to visualize the constructed knowledge graph. Due to the size and density of the full graph, only a focused subgraph is presented. Figure 7 shows a representative subgraph centered on the vapor compression cycle, highlighting related concepts and their connections. This view emphasizes the local structure of the graph and illustrates the prerequisite relationships between conceptual and procedural entities within a single thermodynamic topic.

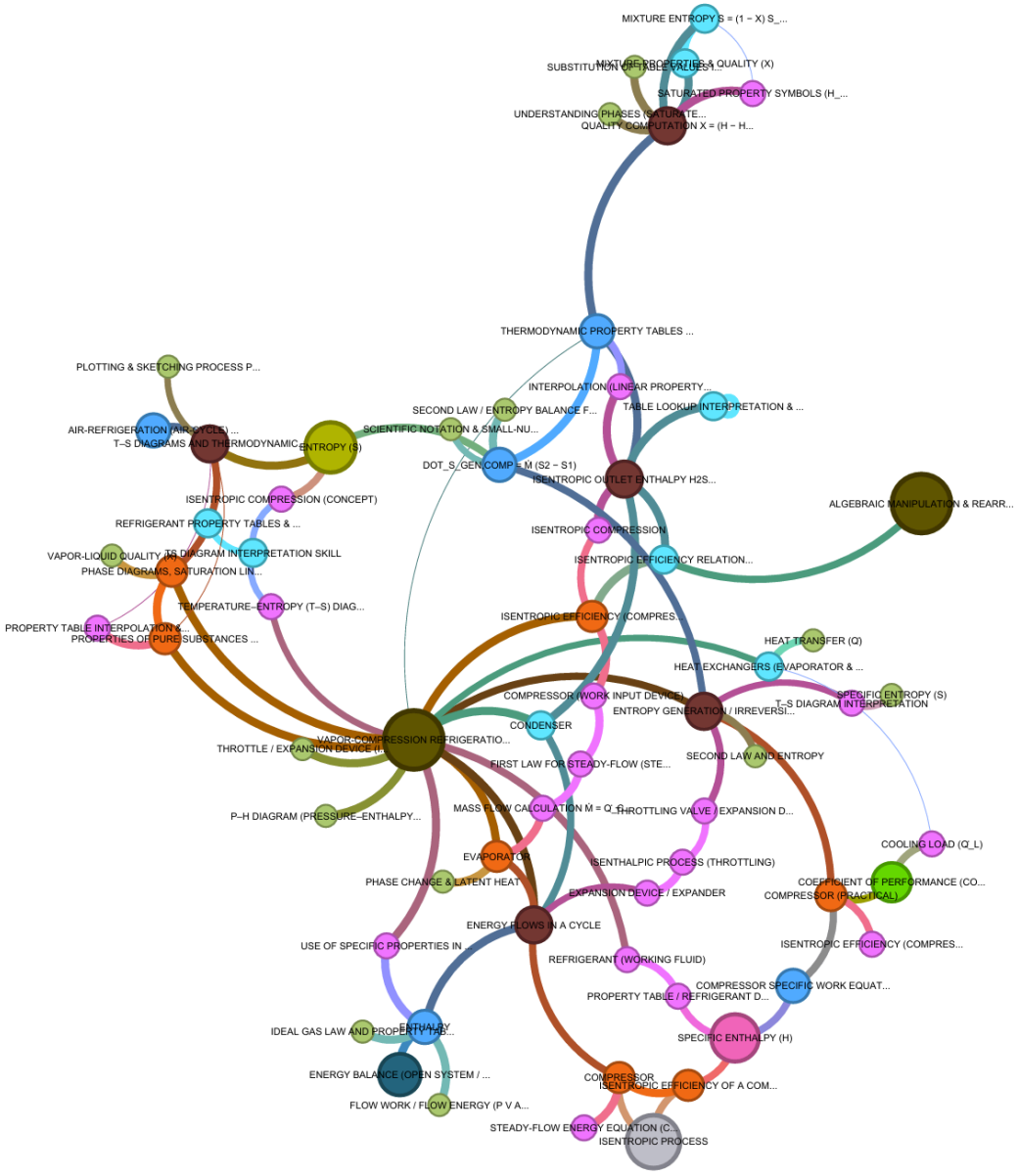

Figure 7: Vapor Compression Cycle Knowledge Subgraph

4. Limitation and Future Work

One limitation of the proposed extraction pipeline is the presence of redundant or near-duplicate entities that differ only in naming or phrasing. Although many of these redundancies are mitigated during graph construction through clustering and community formation, they nonetheless increase graph size and introduce noise. Future work will focus on improving entity normalization and deduplication earlier in the pipeline to consolidate semantically equivalent entities before graph construction, resulting in more compact and interpretable knowledge graphs.

A second limitation arises from the extraction process itself. Preliminary grading indicated that for longer lectures, typically exceeding six pages, the quality and consistency of extracted content begin to degrade, resulting in deviations in structure and coverage. This suggests that the current extraction approach is sensitive to input length and content density. Additionally, during batch processing, fine-grained metrics such as per-lecture token usage and cost could not be reliably captured. Future work will focus on implementing explicit token and cost tracking to obtain more accurate computational metrics. In parallel, hierarchical and chunk-aware extraction strategies will be explored to improve robustness when processing longer or more complex lecture materials.

.

While this study emphasizes knowledge graph construction and structural analysis, it does not directly evaluate educational effectiveness through student interaction. Preliminary work has begun on integrating the knowledge graph with retrieval-augmented generation (RAG) methods, with the goal of pairing graph-based retrieval with a tutoring interface for students in transport phenomena courses. This system would allow students to query concepts, explore prerequisite relationships, and receive explanations grounded in the extracted lecture material. Future studies will collect student feedback to assess the effectiveness of retrieval quality and explanation clarity.

Finally, evaluating the quality of extracted and generated content remains an open challenge. Although preliminary metrics have been developed to assess extraction fidelity and completeness, these measures are still under refinement. Future work will focus on formalizing and validating evaluation metrics for both entity extraction and instructional quality, enabling more systematic assessment of pipeline performance across lectures and courses.

5. Conclusion

This work successfully validates an LLM-driven pipeline for automatically extracting structured, learning-oriented representations from raw lecture materials and organizing them into a comprehensive knowledge graph that captures both conceptual content and prerequisite

structure. The system demonstrates robustness and efficiency, extracting 2,984 unique entities and 3,185 relationship instances across the full lecture set at a low mean processing cost of $0.0289 per lecture. The pipeline preserves complex mathematical derivations and instructional relationships, enabling visualization, structural analysis, and a foundation for downstream retrieval-based tutoring applications.

As future work, we aim to deploy this knowledge-graph-driven system within a live classroom setting, integrating it into a chat-based tutoring interface to evaluate its impact on student learning outcomes. This includes controlled studies comparing students with and without access to the structured prerequisite knowledge system, using longitudinal assessment data to measure improvements in conceptual understanding and performance. Ongoing technical work focuses on improving entity normalization and extraction robustness for longer lectures.

Collectively, these results position knowledge-graph-driven representations as a scalable foundation for next-generation educational tools and personalized tutoring systems in engineering disciplines.

Acknowledgments

This work was supported by the California Education Learning Lab under the grant titled *"Across-the-curriculum AI-based personalized assistants to support student success in engineering."*

References

[1] P. Jackson, “Generating automated problem sets for rapid content delivery and adaptive learning modules,” in *Proceedings of the 2018 ASEE Annual Conference & Exposition*, 2018, doi: 10.18260/1-2--30557

[2] Y. Tang, H. Bai, and R. Catrambone, “Developing deliberate practice for learning engineering dynamics by analyzing students’ mental models,” in *Proceedings of the 2022 ASEE Annual Conference & Exposition*, 2022, doi: 10.18260/1-2--42084..

[3] S. Moore, R. Schmucker, T. Mitchell, and J. Stamper, “Automated generation and tagging of knowledge components from multiple-choice questions,” in *Proceedings of the 11th ACM Conference on Learning @ Scale*, Jul. 2024, pp. 122–133, doi: 10.1145/3657604.3662030.

[4] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. Osazuwa Ness, and J. Larson, “From local to global: A GraphRAG approach to query-focused summarization,” *arXiv preprint arXiv:2404.16130*, Feb. 2025. [Online]. Available: https://arxiv.org/abs/2404.16130

[5] C. Dong, Y. Yuan, K. Chen, S. Cheng, and C. Wen, “How to build an adaptive AI tutor for any course using knowledge graph-enhanced retrieval-augmented generation (KG-RAG),” *arXiv preprint arXiv:2311.17696*, Jan. 2026. [Online]. Available: https://arxiv.org/abs/2311.17696

[6] OpenAI, “Structured model outputs | OpenAI API,” Jan. 22, 2026. [Online]. Available: https://platform.openai.com/docs/guides/structured-outputs

[7] Y. Yan, “Create multi-part problems with random parameterization on Blackboard and Canvas similar to *Mastering* and *Connect*,” in *Proceedings of the 2023 ASEE Annual Conference & Exposition*, 2023, doi: 10.18260/1-2--42782.