Generating Schedule Knowledge Graphs for Structural Activities by Resource Linking Using Retrieval-Augmented Generation

Raynard Vincent Elsantio[1] **and Jacob J. Lin**[1]

[1]Department of Civil Engineering, National Taiwan University, Taipei City, Taiwan

r13521256@ntu.edu.tw, jacoblin@ntu.edu.tw

Abstract -

Construction schedules serve as one of the primary documents for encoding project intent in the planning phase, yet their unstructured, inconsistent, and ambiguous term renders them inaccessible to processing systems. These instances present a critical barrier to automation for construction actions, as processing require explicit, machine-readable representations of tasks to reason about physical interactions and execute actions properly. To address this challenge, we propose a framework that leverages Large Language Models and domain-specific knowledge to transform raw schedule into machine-interpretable knowledge graphs. Our approach first decomposes compound tasks into granular primitives comprising an object, a sub-task, and an action. Each primitive is then grounded in a construction resource knowledge base derived from the official construction resources standard, which provides canonical definitions of manpower, material, and equipment interactions. Using few-shot and chain-of-thought prompting, the system extracts contextual attributes, while a retrieval-augmented generation module, powered by sentence transformers, ensures semantic fidelity through robust matching against the knowledge base. The resulting knowledge graphs explicitly model both Human–Object and Object–Object Interactions with set edges, enabling construction action understanding for automation applications. Validated by seven construction experts across three countries, our outputs demonstrate high constructability and logical sequencing.

Keywords -

Construction Schedule; Large Language Model; Knowledge Graph Generation

1 Introduction

Recent advancements in technology have enabled autonomous systems to operate for extended periods in human-inhabited environments [1, 2], including construction sites. However, processing perception in such dynamic and unstructured settings remains partial and unreliable, which complicates robust decision-making and task execution. This limitation is particularly pronounced in construction, where environmental complexity and task variability demand contextual awareness [3].

Construction schedules serve as the primary planning artifact in this domain, encoding agreed-upon sequences, duration estimates, resource allocations, and logical dependencies among activities [4]. Despite their central role, these schedules are predominantly textual and often exhibit inconsistent terminology, omitted context, and lack of explicit semantic grounding [5, 6]. As a result, they remain inaccessible to machine reasoning systems that require structured, relational representations to support downstream applications of automation planning.

As an automation example, for robots to collaborate effectively with human workers, they must move beyond passive execution and develop task-awareness—the ability to infer intent from high-level descriptions and anticipate required actions [3]. This capability necessitates a predictive understanding of physical interactions, including how manpower, material, and equipment relate during task execution on-site. Without a formal representation of these relationships, robots cannot reason [7]

A critical gap exists between the unstructured, ambiguous, and inconsistent terms in the schedule [5, 6] and the structured knowledge required for robot action. While schedules convey high-level intent (e.g., "Concrete Casting in 1FL"), they do not specify the underlying interaction primitives, such as which agents act on which objects, how construction material transform over time [8]. This absence of semantics prevents direct use of schedules in control architectures that rely on executable and physics-aware models.

To address this gap, this research proposes a rule- and Large Language Model (LLM)-based framework that transforms raw schedule entries into interpretable, executable knowledge graphs. The approach decomposes compound tasks into granular (Object, Sub-Task, Action) primitives, grounds each in a construction resource standard to retrieve canonical object interactions, and generates a construction knowledge graphs. By extracting contextual task attributes through constrained LLM prompting, the system produces semantically rich, machine-actionable representations that support both planning validation. Therefore, this research two core contributions are as follows:

1. We develop an automated construction knowledge graph generation framework that operates directly on unstructured schedule text to enable machines to interpret plans as intended by practitioners.
2. We develop a standard-formalized multi-granular task representation that explicitly models Human-Object and Object-Object interactions to support granular construction action understanding.

2 Related Work

This section reviews prior research related to transforming construction schedules into machine-interpretable representations, linking schedule activities with resources, and enabling construction action understanding. The following subsections discuss relevant literature across these domains and position this research to bridge schedule understanding, resource grounding, and action-level reasoning for construction automation.

2.1 Constuction Schedule Understanding

The transformation of construction schedules from unstructured textual artifacts into semantically rich, machine-actionable knowledge has become a focus in construction informatics, driven by advances in LLMs. A core challenge lies in the inherent fragmentation, terminological inconsistency, and implicit context of schedule activity names, which hinder direct machine interpretation [5, 6]. Early efforts recognized that treating schedules as mere sequences of tasks is insufficient; instead, they must be understood as hierarchical, relational structures encoding procedural logic. In this vein, Hong et al. [9] demonstrated that traditional Natural Language Processing (NLP) clustering methods like Latent Semantic Analysis (LSA) can group repetitive activities despite inconsistent naming, laying groundwork for semantic normalization. More recently, Singh and Hsieh [10], proposed a multi-LLM pipeline that actively augments and generates synthetic schedules using Role-Guided Modular Prompt Chaining (RGPC), converting fragmented Work Breakdown Structures (WBS) into structured, hierarchical knowledge graphs enriched with task dependencies and execution logic.

This evolution culminates in frameworks explicitly designed for LLM-native schedule reasoning. CONSTRUCTA [11] introduces a three-pronged approach—Static Retrieval-Augmented Generation (RAG) for domain knowledge, Contextualized Knowledge RAG for hierarchical and sequential task relationships, and Construction Reinforcement Learning from Human Feedback (RLHF) for aligning outputs with expert preferences—to enable LLMs to not only parse but reason over complex scheduling constraints in semiconductor fabrication. Another research, RAG4CM [12], a retrieval-augmented meant for understanding construction management system that parses schedules into a hybrid knowledge pool combining hierarchical document structure with vector embeddings, enabling precise information retrieval for question answering and progress monitoring.

2.2 Resource Linking

Early approaches in resource linking to a schedule leveraged Building Information Modeling (BIM) [13] by integrating resource constraints directly into a BIM-based scheduling system, where available crews (e.g., formwork, reinforcement teams) were assigned to activities during a particle swarm optimization process ontology-driven linkage between tasks and resources, but require complete, high-fidelity BIM models and manual rule definition, limiting their applicability to projects with fragmented or unstructured planning data.

Recent work has adopted RAG and knowledge graph frameworks [14] to enable flexible, semantic resource linking from textual schedules. RAG4CM [12], which parses project documents into a hierarchical knowledge pool and retrieves relevant resource specifications by combining semantic similarity with document path features, allowing context-aware answers like "Which crew installs drywall?" Similarly, Kirner et al. [15] developed an RAG system over Linked Building Data (LBD) that maps natural language queries (e.g., "What machines are available for concrete pouring?") to machine-interpretable triples describing equipment, locations, and statuses. As a result, they do not explicitly model the underlying action structure or physical interactions required for task reasoning, nor do they transform schedule activities into multi-granular, machine-actionable knowledge suitable for downstream automation.

2.3 Construction Action Understanding

Recent advances in construction automation have shifted from static task execution to dynamic, context-aware action understanding, enabled by structured representations like scene graphs and multi-granular task primitives. Yu and Pan [16, 17] proposed a ConvNet-based framework that learns multi-granularity task primitives which decomposes complex activities into hierarchical levels: subtasks, steps, activities, and actions. Their model uses multi-task learning to simultaneously recognize these granularities from construction videos, finding that the activity level offers the best trade-off between recognition accuracy and semantic richness for HRC. This models cyclic, non-linear workflows common in real-world construction, enabling machine to interpret not just what is being done, but how and why to assist. Complementing this, VeriGraph [18] introduces an execution-verifiable robot planning system where an initial scene graph is incrementally updated as actions are performed; the final state is validated against a goal graph, ensuring physical plausibility and closing the loop between plan and execution.

Building on these foundations, recent work integrates LLMs with domain-specific knowledge to ground action understanding in both visual and textual contexts. In their interior painting case study, Kim et al. [19] established a structured dialogue between a machine and ChatGPT-4, where the LLM interprets natural language instructions and generates JSON-formatted action sequences (e.g., navigateToLocation, paintWall) based on a pre-defined task specification. This pipeline ensures operational integrity by constraining LLM outputs to valid functions, enabling safe, interpretable control. Similarly, Wu et al. [20] employed an ontology-based prompting to guide an LLM in inferring construction activities from construction images; by formalizing visual entities into a domain ontology and using in-context examples, their model accurately maps scenes to 41 predefined activity types. Similarly, Zhong et al. [21] proposed a transformer-based model for learning to generate scene graphs supervised by image-sentence pairs, with visual features as an input from a pair of detected object regions, text embeddings of their predicted categorical labels, and contextual features from other object regions. Their model learns to recognize the visual relationship between the input object pair, represented as a localized Subject-Predicte-Object triplet. Another related research, Liu et al. [7] demonstrate how scene graphs under partial observability can be generated using a Partially Observable Markov Decision Process (POMDP) to enable system to reason about unobserved objects and their relationships, significantly improving task planning robustness in cluttered environments.

3 Methodology

The core objective of this research is to bridge the semantic gap between high-level, natural-language structural construction tasks found in project schedules and the low-level, executable actions required for knowledge graph generation. To achieve this, a multi-stage pipeline was developed see Figure 1, that systematically deconstructs each task into its granular primitives, which are Sub-Task, Object, Action, and an SVO Triplets that serves as the basis of a knowledge graph.

3.1 Experiment Configurations and Setups

The entire pipeline is executed on a local workstation to ensure data privacy and reproducibility. The software stack is built on Python 3.9+ and relies on several key computational dependencies, which are:

1. **NetworkX** [22]: A dependency for constructing, storing, and visualizing the generated knowledge graphs. It enables the creation of directed graphs where nodes represent construction resource entities (Manpower, Material, Equipment) and edges represent their interactions.
2. **Requests**: To interface with the local Ollama server for LLM inference.
3. **Sentence-Transformers** [23] and **FAISS**: For dense vector embeddings and efficient similarity search, forming the backbone of the RAG system used for matching schedule's structural tasks to a construction resource Knowledge Base (KB).

The LLM model is powered by Ollama, a framework to run LLM locally. The specific model used throughout all LLM-powered stages is Llama3.2:latest ([24]). This choice was made for its strong performance in reasoning tasks, its open-source nature, and its ability to run efficiently on specified hardware with a context window of 4,096 tokens. All LLM calls are configured with a temperature of 0.2 to ensure deterministic and focused output.

The construction standard for this research is derived from the 78$^{\text{th}}$ Annual Edition of RSMeans Data [25], which provides construction resource information, including manpower, materials, and equipment. The interactions between resources are further informed by established standards, namely ACI PRC-304-00 [26] for concrete construction and TMS 402/602-22 [27] for masonry construction, enabling the definition of Human-Object Interaction (HOI) and Object-Object Interaction (OOI) relationships. A curated subset of these standards is then preprocessed into a structured JSON format, mapping combinations of *Work Package*, *Object*, and *Sub-Task* to corresponding resources and their interactions. This KB serves as the authoritative source for grounding the extracted semantics in real-world construction practices.

3.2 Schedule Task Match to Construction Resource Knowledge Base using RAG and LLM

The construction schedule is provided as raw input in commonly used digital formats, including text, comma separated values, and spreadsheet files. All schedule entries are first subjected to a preprocessing stage that standardizes textual representations, removes nonessential formatting, and prepares each task row for downstream analysis. This step ensures that differences in file format and writing conventions do not affect the consistency and reliability of subsequent processing.

After preprocessing, each task row is analyzed using a rule-based LLM prompting extraction that follows the

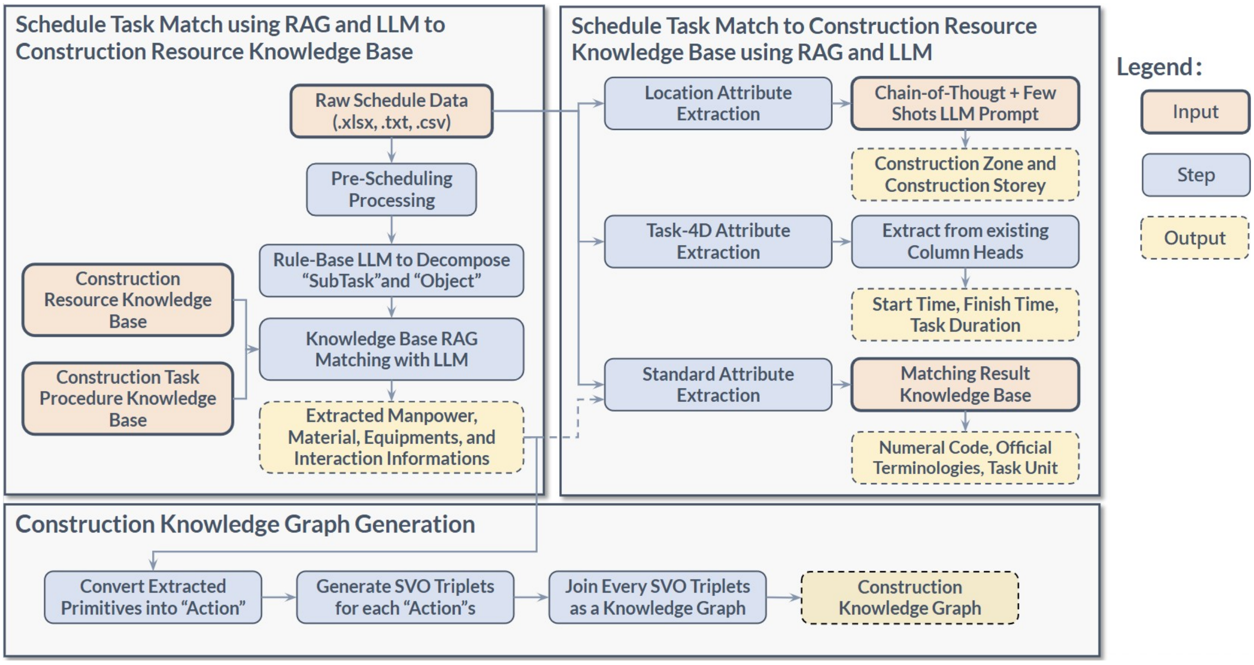

Figure 1. Framework flowchart governing the integration of LLM and construction resource standard to generate a knowledge graph using construction schedule as an input

defined multi-granular task primitives in 2. The system identifies the candidate sub tasks and objects contained within each schedule description. When a task includes multiple objects or construction intents, it is decomposed into separate rows so that each row represents a single granular construction activity.

The decomposed task rows are then matched to a construction resource KB using RAG approach that is actively guided by an LLM. The matching process is triggered and controlled through Chain-of-Thought (CoT) and few-shot prompting [28], which directs the model to reason step by step over the extracted "sub-task" and "object" before selecting the most relevant KB entries. The construction resource KB used in this research not only contains descriptions of manpower, material, and equipment, but also explicitly encodes their interactions within specific construction activities. By decomposing schedule tasks through this guided retrieval process, this phase outputs a task-specific construction resources together with their associated interaction information.

3.3 Construction Knowledge Graph Generation

The transformation from a raw schedule task to a knowledge graph is a hierarchical that requires a multi-step process designed to extract increasing levels of semantic granularity, see Figure 1.

Following the resource matching stage, the extracted manpower, material, equipment, and their interaction information are first concatenated into concise action level sentences. Each sentence represents a task-specific construction action derived directly from the construction resource KB, for example, "$Laborer\ installs\ formwork$" or "$Concrete\_vibrator\ evens\ concrete$". These action sentences provide a standardized and interpretable description of how resources are involved in executing a given schedule task and serve as an intermediate representation between resource extraction and graph construction.

Each action sentence is then converted into a Subject-Verb-Object (SVO) triplets to enable structured semantic modeling. The resulting triplets are made using the NetworkX library [22], where unique subjects and objects form nodes and interactions defined as edges between the associated nodes. By joining all triplets into a single graph, the system generates a construction knowledge graph that captures task-specific resource entities and their interactions in a structured and machine-interpretable form.

3.4 Construction Schedule's Attribute Extraction

In parallel with the construction knowledge graph generations phase, another phase of processing extracts a rich set of contextual attributes for each schedule task. These attributes provide the necessary metadata to place the abstract knowledge graph within a real-world scenario. The attributes are categorized into four main clusters:

1. **Location Attributes**: This group includes

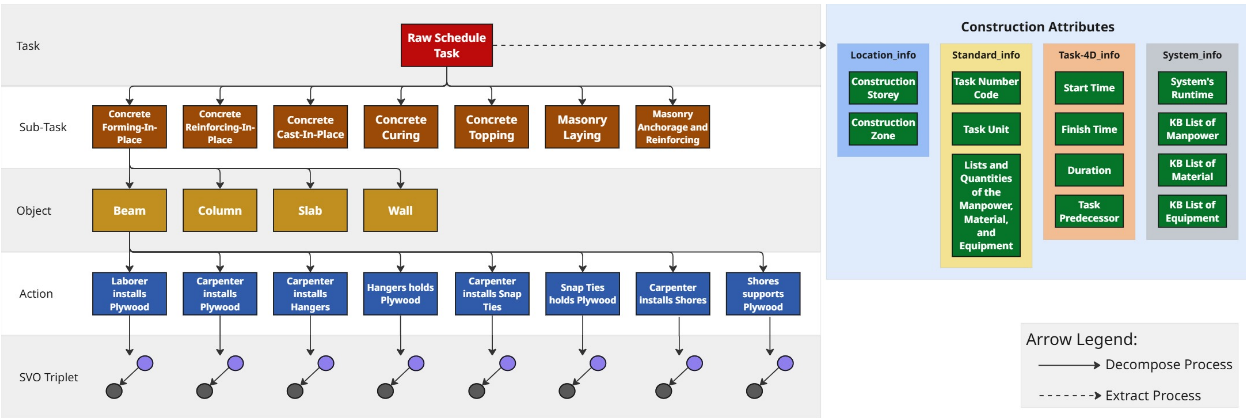

Figure 2. On the left scheme is the derived multi-granularity task primitives which breaks down the raw schedule text into actions. On the right scheme are the attributes that are extracted from the raw schedule

$location\_storey$ (e.g., "2nd floor", "Level 5") and $location\_zone$ (e.g., "C.L. 9 - 14", "Area E"). These are extracted using LLMs with an engineered CoT and few-shot prompts. The LLM is instructed to first check for explicit mentions in the task text. If none are found, it is directed to infer the location from the context of preceding tasks in the schedule, forcing the LLM to reason using the sequential nature of construction schedule.

2. **Standard Attributes**: This group links the task to standardized industry standard. It includes $RSMeans\_Name$ (its full name of the RSMeans item), $RSMeans\_Code$ (its unique numeral identifier), and $RSMeans\_Unit$ (its unit of measure, e.g., "C.Y," "S.F", "cwt"). This information is extracted using a three-tiered approach: exact matching, FAISS-based semantic search, and LLM fallback.
3. **Task-4D Attributes**: This group contains temporal and logical data directly from the schedule file, which are: Start Time, Finish Time, Duration, and Task Predecessors. These are extracted during the initial loading phase and are used to understand the task's position in the overall project timeline.
4. **System Attributes**: This group is a meta-category generated during processing. It includes $System\_Runtime$, which logs the total time taken to process the task through the entire pipeline. Moreover, the $Lists\_of\_Manpower$, $Lists\_of\_Material$, and $Lists\_of\_Equipment$ are also listed. These are derived directly from the SVO triplets. By aggregating all subjects and objects from the triplets and filtering them by their inferred types.

4 Results and Discussions

The proposed pipeline successfully processed four construction schedules, generating interpretable knowledge graphs for each. As illustrated in Figures 3(a) and (b) the system correctly identifies canonical objects (beam, slab, etc) and sub-tasks (forming-in-place, water concrete curing, etc.), and retrieves semantically grounded actions from the construction resource KB. The resulting knowledge graphs explicitly model Human–Object Interactions (e.g., *Carpenter_1 installs Plywood*) and Object–Object Interactions (e.g., *Concrete_pump pours Concrete*). This granularity enables downstream reasoning about task feasibility, resource allocation, and a probable safety constraints which are very crucial for automation in construction.

Another drawback was observed in the location extraction module, where the LLM-based CoT prompting occasionally inferred plausible but unsupported storey or zone attributes (e.g., inferring "Level 4" for a task lacking explicit locational cues, based on contextual proximity to prior tasks). However, these instances remained physically plausible (e.g., no "Basement 3" in a 5-story building) and did not propagate to erroneous structures. Crucially, the architecture (rule-based filtering + KB grounding + constrained prompting) ensured that all core in Figure 2 were robust and reliable, preserving the knowledge graph's fidelity for improving decision-making.

An interesting point of comparison arises between our planning-phase knowledge graph generation and the video-based human-object interaction (HOI) detection framework developed by Khosiin et al. [17]. While our system generates knowledge graphs ahead of time from schedule standpoint, their method extracts knowledge graphs from site videos during execution standpoint, capturing real-time human-object interactions. This align-

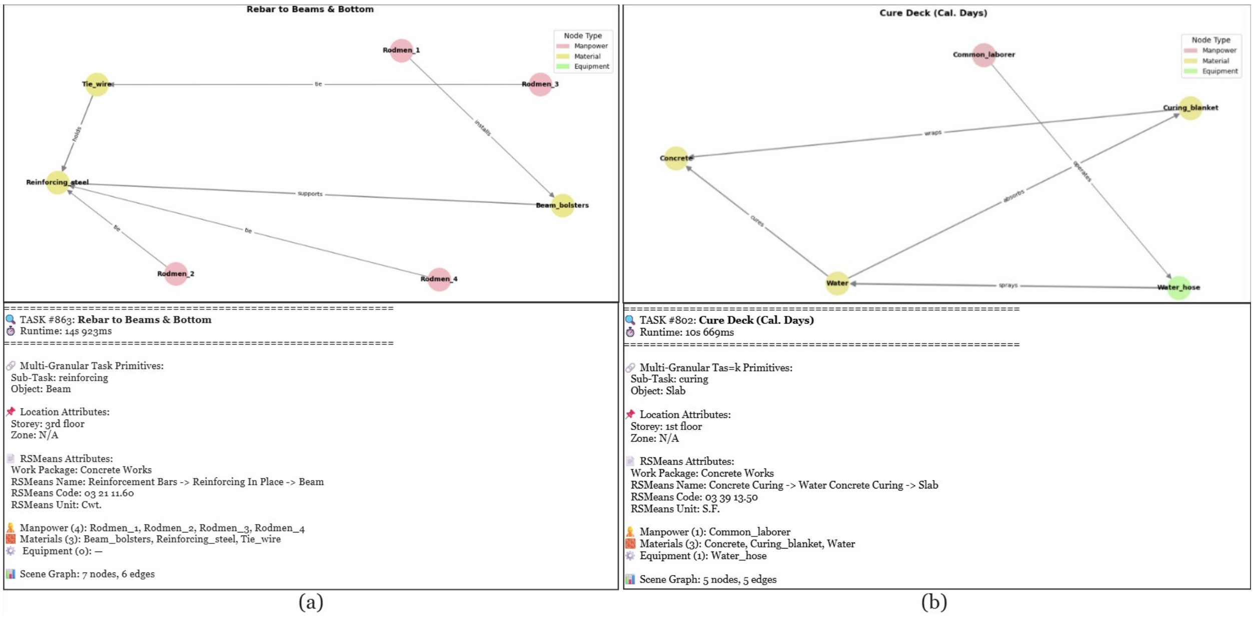

Figure 3. Results of the generated construction knowledge graphs for (a) reinforcing-cast-in-place for beam structure; (b) water concrete curing for slab structure

ment presents a potential closed-loop system. On the one hand, in the planning phase, our knowledge graphs provide the system with a semantic framework for expected interactions, on the other hand, in the execution phase, the video-based knowledge graphs allow for continuous monitoring of actual performance, enabling real-time comparison with the plan. Any deviations, such as pouring without formwork, can be flagged, triggering alerts or adjustments.

Notably, this system's performance was achieved on modest hardware which is MSI GF65 Thin 10UE laptop equipped with an Intel(R) Core(TM) i7-10750H CPU (12 cores), 16GB RAM, and Windows 11 Home. Despite the absence of dedicated GPU acceleration or cloud resources, the pipeline processes tasks efficiently through its architecture. This demonstrates the framework's practical feasibility for real-world deployment, where access to high-end computing infrastructure can possibly be limited.

Such integration transforms knowledge graphs from static documentation artifacts into dynamic, updatable world models which enables the system to understand planning and verify execution. This is foundational for construction action automation, where robot or other machines must move beyond passive execution to proactive, context-aware assistance [8, 16]. In addition, the extracted attributes transforms a simple task description into a multi-dimensional data point, enabling future integration with BIM and other resource management systems. Therefore, the demonstrated interpretability of our knowledge graph framework confirms that rule- and LLM-based knowledge graph is a viable pathway to bridge the planning-execution gap in task planning on a construction site.

5 Research Limitations

This research operates under several deliberate and practical constraints. The identification of construction resources is primarily grounded in the 78th Annual Edition of RSMeans Data as the benchmarked construction resource standard, while the interaction is further guided by ACI PRC-304-00 and TMS 402/602-22. Although this combination establishes a coherent and standardized framework for resource definition and interaction modeling, it inherently limits the system's coverage to the scope defined by these standards. Tasks falling outside this domain—such as custom or emerging construction methods (e.g. 3D-printed formwork)—will not be accurately parsed or grounded, potentially leading to omission or misrepresentation in the resulting knowledge graphs.

The pipeline's RAG-based KB matching relies on a rule-augmented, not learned, KB. This design prioritizes interpretability and avoids black-box LLM hallucination, but it demands a comprehensive and well-structured KB. As such, the scalability and robustness of the system are directly contingent on the breadth and granularity of the underlying RSMeans-derived KB. Future extensions to new work types will require systematic KB expansion.

6 Research Validations

To validate the practical utility and field relevance of the generated knowledge graphs and associated attributes, seven on-site construction experts reviewed representative system outputs. Across all evaluations, the experts unanimously affirmed that the representations are operationally accurate and reflect real-world construction practices. They emphasized the value of explicitly modeling OOI, such as "$Rodmen\ installs\ beam\_bolsters$" or "$Form\_oil\ protects\ plywood$", noting that such interactions are often omitted in traditional schedules yet are essential for quality control, compliance, and logical sequencing. The granularity of manpower allocation was also validated as realistic and actionable. As one expert remarked: "It is better to have 4 people to operate the concrete pump: 1 worker drives, 1 worker parks and positions the truck; 2 workers manage the electrical systems and concrete transfer from the truck to pump boom; and 1 guides and aligns the placing boom during pouring." This level of detail was recognized as directly transferable to on-site planning. Overall, expert feedback was instrumental in refining the attribute extraction scheme and optimizing the construction resource KB, enhancing the system's fidelity to real-world workflows.

7 Conclusions

This research presents a structured pipeline that transforms unstructured construction schedules into executable semantic knowledge for system reasoning and planning. By integrating rule-based task decomposition, LLM-assisted semantic parsing, and resource-standard-driven retrieval-augmented generation, the framework enables automated construction of knowledge graphs that capture tasks across multiple levels of abstraction, from planning intent to physical interactions among manpower, materials, and equipment. Grounded in industry standards, these representations allow construction schedules to function as machine-interpretable knowledge sources for downstream construction automation. Through domain-constrained prompting and knowledge-base decomposition, the outputs remain interpretable, traceable, and aligned with real-world practices. While limitations persist, including dependence on benchmark coverage and uncertainty in locational inference, the results demonstrate the viability of schedule-driven knowledge graph generation for context-aware construction automation. Future work will enhance evaluation by incorporating both qualitative and quantitative measures, including LLM-as-a-qualitative-judge with Likert-scale scoring for logical correctness and constructability, as well as Graph Edit Distance (GED) for measuring structural similarity against expert-annotated graphs.

Acknowledgment The authors would like to thank seven construction experts for their cooperation and knowledge in validating the suggested knowledge graphs' practicality and interpretability. We thank Muhammad Asa Raushan Fatih, Princessa Auditorina Masayu, Hank Tsung-Han Tsai, Eric ChaoYung Huang, Reubenhard Sonda, Andreas Maxwel Sitompul, and Obed Adi Kusuma for their invaluable contributions to the expert interview process.

References

[1] P. Khandelwal, S. Q. Zhang, J. Sinapov, M. Leonetti, ..., and R. Mooney. Bwibots: A platform for bridging the gap between ai and human–robot interaction research. *The International Journal of Robotics Research*, 36(5-7):635–659, 2017. doi:10.1177/0278364916688949.

[2] N. Hawes, C. Burbridge, ..., and B. Lacerda. The strands project: Long-term autonomy in everyday environments. *IEEE Robotics Automation Magazine*, 24(3):146–156, 2017. doi:10.1109/MRA.2016.2636359.

[3] E. C. Grigore, A. Roncone, O. Mangin, and B. Scassellati. Preference-based assistance prediction for human-robot collaboration tasks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4441–4448, 2018. doi:10.1109/IROS.2018.8593716.

[4] *Project Management Book of Knowledge*, volume 7th Edition. Project Management Institute, New Newtown Square, PA, August 2021.

[5] Robert B. H. and Photios G. I. Scheduling projects with repeating activities. *Journal of Construction Engineering and Management*, 124 (4):269–278, 1998. doi:10.1061/(ASCE)0733-9364(1998)124:4(269).

[6] R. B. Harris and P. G. Ioannou. Scheduling projects with repeating activities. *Journal of Construction Engineering and Management*, 124(4):269–278, 1998. doi:10.1061/(ASCE)0733-9364(1998)124:4(269).

[7] S. Amiri, K. Chandan, and S. Zhang. Reasoning with scene graphs for robot planning under partial observability. *IEEE Robotics and Automation Letters*, 7(2): 5560–5567, 2022. doi:10.1109/LRA.2022.3157567.

[8] Z. Pan and Y. Yu. Learning multi-granularity task primitives from construction videos for human-robot collaboration. In *Computing in Civil Engineering 2023*, pages 674–681, 01 2024. doi:10.1061/9780784485224.081.

[9] Y. Hong, H. Y. Xie, G Bhumbra, and I. Brilakis. Comparing natural language processing methods to cluster construction schedules. *Journal of Construction Engineering and Management*, 147(10): 04021136, 2021. doi:10.1061/(ASCE)CO.1943-7862.0002165.

[10] A. K. Singh and S. H. Hsieh. Multi-llm-based augmentation and synthetic data generation of construction schedules and task descriptions with slm-as-a-judge assessment. *Advanced Engineering Informatics*, 69:103825, 2026. ISSN 1474-0346. doi:https://doi.org/10.1016/j.aei.2025.103825.

[11] Y. F. Zhang and X. Yang. Constructa: Automating commercial construction schedules in fabrication facilities with large language models, 2025. URL `https://arxiv.org/abs/2502.12066`.

[12] C. K. Wu, W. J. Ding, ..., and X. Li. Retrieval augmented generation-driven information retrieval and question answering in construction management. *Advanced Engineering Informatics*, 65:103158, 2025. ISSN 1474-0346. doi:https://doi.org/10.1016/j.aei.2025.103158.

[13] H. X. Liu, M. A. Hussein, and M. Lu. Bim-based integrated approach for detailed construction scheduling under resource constraints. *Automation in Construction*, 53:29–43, 2015. ISSN 0926-5805. doi:https://doi.org/10.1016/j.autcon.2015.03.008.

[14] P. Lewis, E. Perez, ..., and D. Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020.

[15] L. Kirner, J. Oraskari, and S. Brell-Cokcan. Developing a rag-based system for natural language access to linked building data on construction sites. 07 2025.

[16] Z. Y. Yu, J. Y. Zhang, ..., and C. L. Wu. Recode: Unify plan and action for universal granularity control, 2025. URL `https://arxiv.org/abs/2510.23564`.

[17] M. W. Khosiin, J. J. Lin, E. A. Suryo, K. P. Negara, I. Aknuranda, and C. S. Chen. Video-based productivity monitoring of worker and large-scale object interactions in construction sites. In *Proceedings of the 42nd International Symposium on Automation and Robotics in Construction*, pages 580–587, Canada, 2025. doi:10.22260/ISARC2025/0076.

[18] D. Ekpo, M. Levy, S. Suri, C. Huynh, and A. Shrivastava. Verigraph: Scene graphs for execution verifiable robot planning, 2024. URL `https://arxiv.org/abs/2411.10446`.

[19] K. K. Kim, P. Ghimire, and P. C. Huang. Framework for llm-enabled construction robot task planning: Knowledge base preparation and robot–llm dialogue for interior wall painting. *Robotics*, 14(9), 2025. ISSN 2218-6581. doi:10.3390/robotics14090117.

[20] C. Zeng, T. Hartmann, and L. Y Ma. Ontology-based prompting with large language models for inferring construction activities from construction images. *Advanced Engineering Informatics*, 69:103869, 2026. ISSN 1474-0346. doi:https://doi.org/10.1016/j.aei.2025.103869.

[21] Y. W. Zhong, J. Shi, J. W. Yang, C. L Xu, and Y. Li. Learning to generate scene graph from natural language supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1823–1834, October 2021.

[22] A. Hagberg, A. Schult, and J. Swart. Exploring network structure, dynamics, and function using networkx. *Python in Science Conference*, 2008. doi:10.25080/TCWV9851.

[23] N. Reimers and I. Gurevych. all-minilm-l6-v2 sentence transformer. [Online]. Available: https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.

[24] A. Grattafiori, A. Dubey, ..., and A. Jauhri. The llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

[25] *Building Construction Costs with RSMeans data*, volume 78th Annual Edition. R S Means Co, Greenville, November 2019.

[26] ACI Committee 304. *Guide for Measuring, Mixing, Transporting, and Placing Concrete*, volume 00. American Concrete Institute, November 2009.

[27] The Masonry Society. *Building Code Requirements and Specification for Masonry Structures*. Number TMS 402-22/TMS 602-22. The Masonry Society, Longmont, CO, 2022. ISBN 978-1-948547-06-8.

[28] D. Yang, M. Kim, S. Kim, ..., and J. Yeo. Llm meets scene graph: Can large language models understand and generate scene graphs? a benchmark and empirical study, 2025. URL `https://arxiv.org/abs/2505.19510`.