2025 European Conference on Computing in Construction

CIB W78 Conference on IT in Construction

Porto, Portugal
July 14-17, 2025

AI-POWERED APPLICATION FOR CONSTRUCTION SCHEDULE MANAGEMENT USING NATURAL LANGUAGE

Aritra Pal[1,2], Danny Murguia[2], and Campbell Middleton[2]
[1]Department of Civil Engineering, Indian Institute of Technology Madras, Chennai, India
[2]Department of Engineering, University of Cambridge, Cambridge, United Kingdom

Abstract

Project schedules are critical for construction management, yet they remain inaccessible to many professionals due to complex scheduling software and costly licenses. As a result, site engineers and last planners rely on static, outdated schedule copies, limiting proactive decision-making. This study proposes an AI-powered schedule management application that enables natural language interaction with project schedules. By leveraging knowledge graphs (KG) and natural language processing (NLP), the app enhances accessibility, reduces inefficiencies, and ensures real-time interaction with updated schedules. Central to this approach is the improvement of Human-Data Interaction (HDI), allowing users without technical scheduling expertise to intuitively engage with complex project data. The results demonstrate improved usability and engagement with schedules. This research contributes an AI-powered schedule management solution to bridge the gap between scheduling expertise and project execution.

Introduction

Effective schedule management is critical in the construction industry, where project timelines, resource allocation, and task dependencies must be carefully coordinated (Hua et al., 2022). Poor handling of schedules often results in inefficiencies, miscommunications, and costly delays (Castañeda et al., 2025). These schedules contain essential details about project progress and resource distribution, but only professionals with scheduling expertise can accurately interpret them. Many construction professionals lack the necessary knowledge to derive meaningful insights from project schedules due to the steep learning curves of the existing scheduling tools (Brown, 2003), leading to a reactive rather than proactive approach to schedule utilisation. Another significant challenge is limited access to scheduling software, particularly for small or medium-sized construction firms. Many commercial or proprietary scheduling tools require expensive licenses, which are typically available only to project management departments and senior staff members (Kolosky, 2024). As a result, site engineers and last planners—who play a critical role in project execution—are often deprived of direct access. Instead, they rely on printed copies or PDF versions of schedules, making it challenging to interact dynamically with schedule data. This reliance on static formats can lead to inefficiencies, outdated information usage, and misalignment with project timelines. These issues highlight a broader challenge in Human-Data Interaction (HDI), where users must be able to access, understand, and engage meaningfully with complex data representations. In the context of construction, enabling intuitive and inclusive HDI mechanisms is essential for democratizing access to critical schedule information across project teams (Calvetti et al., 2024). Poor HDI design can result in information bottlenecks and hinder timely decision-making, especially for those without technical expertise in scheduling tools.

To address these challenges, this study proposes an AI-enabled schedule management approach that leverages knowledge graphs (KG) and Natural Language Processing (NLP) to enable project professionals to interact with schedules using natural language commands. This NLP-driven approach enhances accessibility, efficiency, and real-time decision-making, allowing users to query schedules, receive updates, and generate insights without requiring specialised scheduling expertise. Integrating NLP-driven automation ensures project teams work with up-to-date schedule information, improving collaboration, productivity, and overall project outcomes.

The proposed schedule management approach consists of three key phases. First, project schedules are converted into a structured knowledge graph, allowing for efficient data retrieval and relationship mapping between tasks, dependencies, and timelines. Next, users can query the schedule using natural language input, making it easier to retrieve specific information, track progress, and generate actionable insights without requiring specialised expertise. Finally, the application is deployed as a web-based platform, ensuring accessibility for all project stakeholders, including site engineers, planners, and management teams, enabling seamless interaction with up-to-date schedule information. To this end, a case study implementation demonstrates the effectiveness of the proposed NLP-driven schedule management application in improving the accessibility of construction schedules.

The remainder of the paper is structured as follows: the next section reviews previous research on graph-based and

NLP-driven approaches to construction schedule management, followed by an overview of the proposed methodology and its implementation. Subsequently, the results are discussed, and potential future research directions are highlighted. The paper concludes with a summary of key findings and final remarks.

Related Studies

Since the 1990s, research in construction project scheduling has focused on optimising activity sequences and resource allocation under various constraints (Li, 1996). Early studies explored heuristic and meta-heuristic methods, including ant colony optimisation, symbiotic organisms search, artificial immune systems, and genetic algorithms (GA). However, recent studies have focused more on AI-based methods for automatic schedule management. Related studies in this area can be broadly divided into three sub-areas: NLP in construction schedule management, graph-based approaches in construction scheduling, and information retrieval from knowledge graphs Using NLP. The following sections discuss them in detail.

NLP in Construction Schedule Management

The integration of natural language processing (NLP) and artificial intelligence (AI) into construction scheduling has addressed inefficiencies, reduced human errors, and bridged automation gaps. Hong et al. (2021b) tackled inconsistencies in activity names, demonstrating that latent semantic analysis (LSA) outperformed other clustering methods in detecting repetitive activities, improving data preprocessing for AI applications. Building on this, ul Hassan and Le (2022) developed an NLP-based dependency parsing system to automatically extract scheduling activities from contract documents, achieving high accuracy (94% recall, 95% precision). Similarly, Ren and Zhang (2022) introduced a semantic NLP-based information extraction (IE) method to generate structured schedules from procedural documents, significantly reducing manual effort (by 89.33%) while achieving 95.83% precision. Moving toward schedule validation, Amer et al. (2022) developed a machine learning model capable of detecting logic errors with an F1 score of 88.3%. Expanding this approach, Amer et al. (2023) introduced the Implicit Logic Checker (ILC), a Transformer-based model that refines look-ahead plans with an F1 score of 91%. Lastly, Prieto et al. (2023) explored large language models (LLMs) like ChatGPT for automated schedule generation, finding them promising but in need of further refinement. Collectively, these studies highlight the growing role of NLP in automating construction scheduling, spanning from data extraction to logic validation and intelligent scheduling, while underscoring the need for continued research and development.

Graph-Based Approaches in Construction Scheduling

Graph-based methods have emerged as an effective approach for improving construction scheduling by leveraging historical project data and reducing reliance on experienced schedulers. Hong et al. (2021a) pioneered a graph-based approach to identify time- and risk-efficient construction patterns from past projects, demonstrating that excavation activities could be completed in just 0.6% of total project time. Building on this, Hong et al. (2023) introduced the Graph-Based Automated Scheduling (GAS) method, which captures tacit scheduling knowledge, classifies sequences, and optimises schedules for cost and time efficiency. This method produced results 6.70% closer to actual execution than traditional planned schedules. Further research by Hong et al. (2022) utilised a graph-based technique to analyse 353 historical schedules, identifying inefficiencies in earthwork sequencing, particularly in road construction, and revealing that learned sequences were 26.7% more accurate than planned ones. More recently, Yao et al. (2024) introduced a Deep Reinforcement Learning (DRL) model integrated with a Graph Convolutional Network (GCN) to dynamically optimise scheduling. This AI-driven approach reduced project duration and computational runtime, reinforcing the potential of graph-based methodologies in construction scheduling. Together, these studies demonstrate the potential of integrating NLP and graph-based modelling to optimise scheduling workflows and drive efficiency in construction management.

Information Retrieval from Knowledge Graph Using NLP

Recent studies have explored information retrieval from knowledge graphs using NLP to enhance compliance checking, risk assessment, accident analysis, and safety guideline recommendations in construction. Li et al. (2024) proposed an automated BIM compliance checking framework, leveraging ontology-based knowledge modelling and deep learning to structure building standards into a Neo4j knowledge graph. A matching algorithm enabled automated rule-checking. Lee and Lee (2024) introduced an NLP-based risk-assessment system, developing an entity-name recognition and keyword-extraction engine to extract predefined safety knowledge from unstructured construction data automatically. The extracted knowledge was structured into an entity-relationship-based risk-assessment knowledge base, enhancing accessibility and reducing reliance on individual expertise. Hong et al. (2024) focused on accident analysis, constructing an accident hazard ontology by modelling keyword relationships and structuring accident-related data into a graph database. The framework identified accident patterns, quantified severity, and predicted critical hazards using network analysis, enabling real-time safety tracking. Finally, Lee and Ahn (2024) addressed safety guideline inefficiencies by developing a recommendation framework that transforms unstructured safety texts into a knowledge graph. The PageRank and Louvain Clustering algorithms ranked safety measures based on relevance, improving retrieval efficiency. A case study on scaffolding safety demonstrated the system's effectiveness in identifying key

hazard clusters like ‘fall’ and ‘drop’.
Related studies highlighted that knowledge graphs with NLP have the potential for automatic information retrieval from text documents. However, similar methods need to be tested for automatic construction schedule management.

Methodology

This research is motivated by practical challenges observed in construction projects, particularly the limited accessibility and usability of project schedules among site-level professionals. Informal consultations with industry stakeholders revealed issues such as reliance on static schedule formats, steep learning curves of scheduling tools, and poor Human-Data Interaction (HDI). A structured literature review using databases like Scopus and Web of Science confirmed these gaps and highlighted the need for more intuitive, AI-driven solutions. In response, this study integrates Knowledge Graphs (KG) for structured schedule representation, Natural Language Processing (NLP) for conversational querying, and HDI principles to ensure the system supports clear, inclusive, and user-friendly interaction. The methodology is structured across three core stages: (1) conversion of schedules to a knowledge graph, (2) querying a schedule through natural language input, and (3) deployment of the application through a web-based platform. Fig. 1 shows the Workflow of the proposed methodology. In the subsequent sections, each stage is explained in detail.

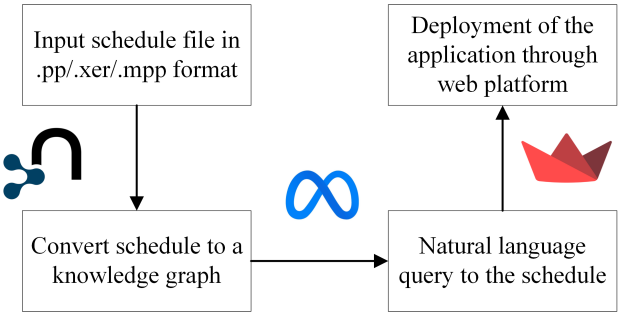


Figure 1: Workflow of the proposed method.

Conversion of Schedule to Knowledge Graph

Using the Python wrapper for the MPXJ Java library, construction schedules can be automatically read from various file formats and databases, including Asta Power Projects, Primavera P6, and Microsoft Project. MPXJ extracts tasks and work breakdown structure (WBS) elements from the schedule database and maps them as nodes in a Neo4j graph database. Each task or WBS element is stored as a node with attributes such as name, UID, start and finish dates, duration, total float, and critical path status. To represent the project hierarchy, the WBS elements are linked to their respective tasks using the “HAS” relationship. Additionally, logical dependencies between tasks, such as successor and predecessor relationships, are captured with attributes like relationship type [start-to-start (SS), finish-to-start (FS), start-to-finish (SF), and finish-to-finish (FF)] and lag time, represented by the “SUCCESSOR” relationship in the graph. At the start of graph creation, existing nodes are cleared before processing, and nonnull attributes are validated to ensure data integrity and prevent errors. Finally, each node is assigned a path attribute to define its hierarchical position within the project, enabling efficient querying and visualisation of the schedule in Neo4j. A graph representation of a construction project schedule is shown in Fig. 2. Fig. 3 shows a detailed view of WBS and Task nodes and relationships visualised in a Neo4j browser. Here, task nodes and WBS nodes are represented by orange and magenta colours, respectively. The right side of Fig. 3 shows the attributes of a task stored in the graph database.

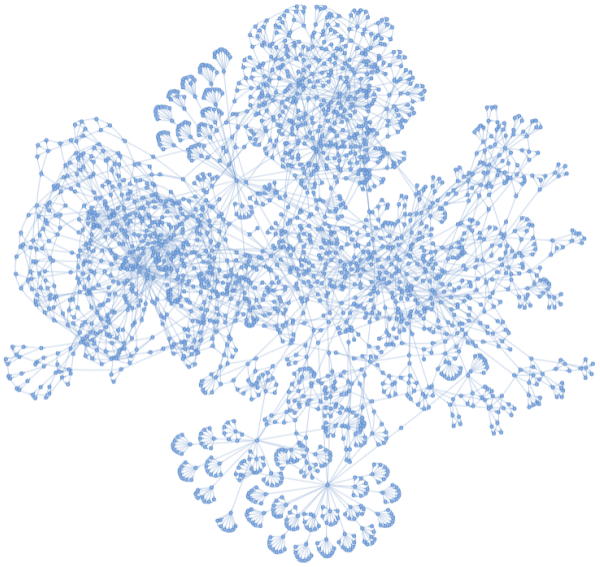


Figure 2: A graph representation of a construction schedule.

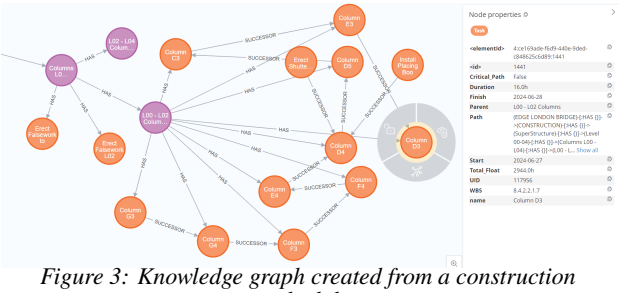
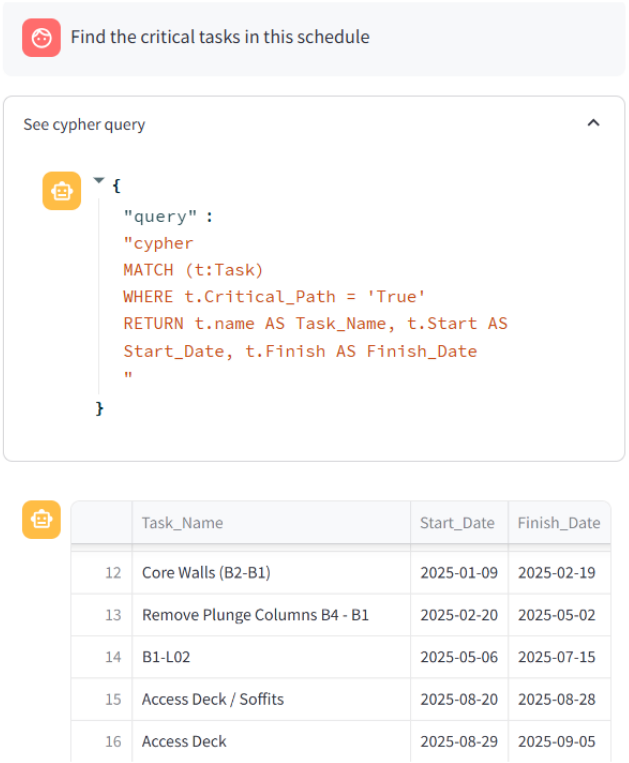


Figure 3: Knowledge graph created from a construction schedule

Natural Language Query to the Schedule

Once the graph database is created, users can query schedule information in natural language by leveraging a large language model (LLM). The proposed application facilitates this by integrating LangChain’s GraphCypherQAChain module and prompt engineering to transform user queries into Cypher statements for retrieving project data. The application utilises the Meta Llama 3.1 8B open-source model or similar LLM to dynamically interpret natural language queries and generate

structured Cypher queries. The graph schema helps in Retrieval-augmented generation (RAG), which enhances accuracy, allowing the model to fetch relevant schedule data from the Neo4j graph database before formulating responses. This RAG-based approach ensures the AI remains context-aware, improving query precision and response reliability. Additionally, prompt engineering plays a crucial role by using a custom Cypher generation template that ensures schema compliance and structured output. LangChain's GraphCypherQAChain processes queries within this structured framework, dynamically constructing optimised Cypher statements for execution. Users can extract key insights such as critical activities, look-ahead plans, and task dependencies, with results visualised through interactive Gantt charts. An example is shown in Fig. 4. It can be seen that, given a user query, the application converts it into a valid Cypher query to retrieve tasks from the graph database. Further, these tasks are plotted as a Gantt chart for better visualisation, as shown in Fig. 5.



	Task_Name	Start_Date	Finish_Date
12	Core Walls (B2-B1)	2025-01-09	2025-02-19
13	Remove Plunge Columns B4 - B1	2025-02-20	2025-05-02
14	B1-L02	2025-05-06	2025-07-15
15	Access Deck / Soffits	2025-08-20	2025-08-28
16	Access Deck	2025-08-29	2025-09-05

Figure 4: Example of a natural language query to the schedule.

Deployment of the Application through Web Platform

The schedule management application is deployed locally using Streamlit, running on localhost for easy access through a web browser. The user interface (UI) provides options for selecting different LLM models, allowing users to choose the most suitable model for processing their queries. The application features a two-page layout: one page is dedicated to converting project schedules into a graph database and visualising the data, while the second

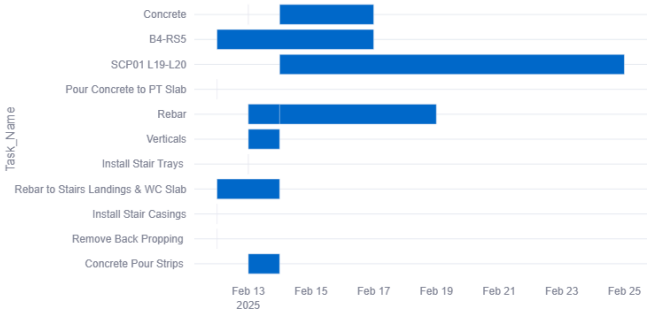


Figure 5: A Gantt chart representation of retrieved project tasks.

page enables users to interact with the schedule in natural language. The system connects to a Neo4j database, ensuring efficient data retrieval. Users can launch the application by running a Python script, which starts the Streamlit server, providing an intuitive and interactive experience, making it ideal for offline use. In the future, the application will be hosted on a cloud platform, enabling users to access it over the internet without requiring local installations. Cloud deployment will also support scalability, remote collaboration, and real-time data updates, enhancing its accessibility and usability for construction planners and project managers. The UI of the Schedule Assistant app is shown in Fig. 6.

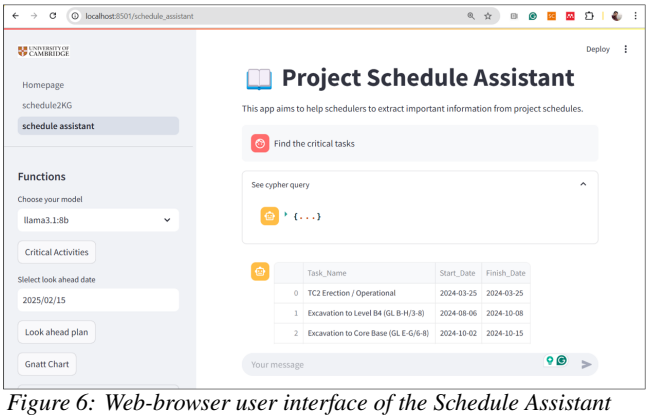


Figure 6: Web-browser user interface of the Schedule Assistant app

Implementation

The application was implemented and tested on an office building construction project in central London. The building consists of 26 floors with a gross floor area of 275,500 sq ft. The master schedule, developed in Asta Power Projects, comprised nine WBS levels and approximately 1,800 tasks. Using the schedule-to-knowledge graph (KG) conversion methodology, the project schedule was transformed into a Neo4j graph database. Once the knowledge graph was created, various common schedule queries were tested, including identifying critical tasks, exporting look-ahead schedules for a user-defined duration,

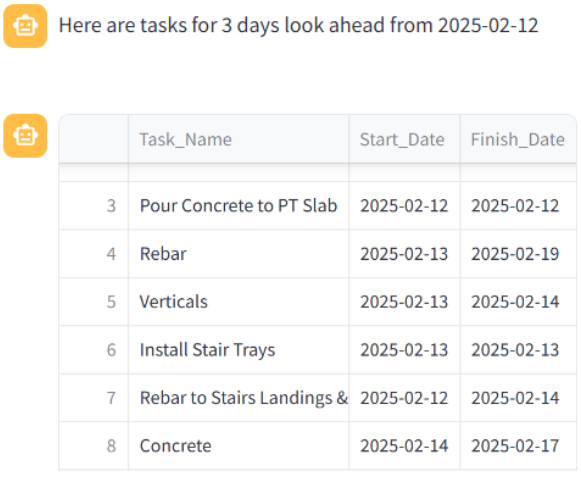
filtering tasks based on attributes, retrieving specific details of tasks and WBS levels, and analysing task relationships such as preceding and succeeding dependencies. For natural language interaction, a chat interface developed using Streamlit was implemented, enabling users to query the project schedule efficiently. The following sections present the results of various interactions.

Identification of Critical Tasks

In construction schedule management, critical tasks are activities that directly impact the overall project timeline. Delays in these tasks can cause significant project disruptions, making their identification crucial for effective planning and risk mitigation. The Streamlit-based chat interface enabled users to retrieve critical tasks using natural language commands such as "What are the critical tasks?" or "Find the critical tasks in this schedule." The application converted the user's command into a Cypher statement and queried all tasks where the Critical Path attribute is marked as "True," ensuring precise extraction of critical activities. The application dynamically returned the list of critical tasks along with their start and finish dates. An example of the critical task retrieval is shown in Fig. 4.

Generation of Look-Ahead Schedules

Look-ahead schedules provide a short-term planning view of upcoming tasks, helping project teams plan and coordinate work efficiently. These schedules enable last planners to anticipate critical activities, allocate resources effectively, and mitigate potential delays. Users selected a target date through the UI, and the application retrieved tasks scheduled to start between the current and selected dates. The application queried the Neo4j database using Cypher, ensuring efficient data extraction. The extracted tasks were displayed in a structured table and visualised using an interactive Gantt chart. Fig. 7 shows an example of 3 days of look-ahead planning using the proposed application.

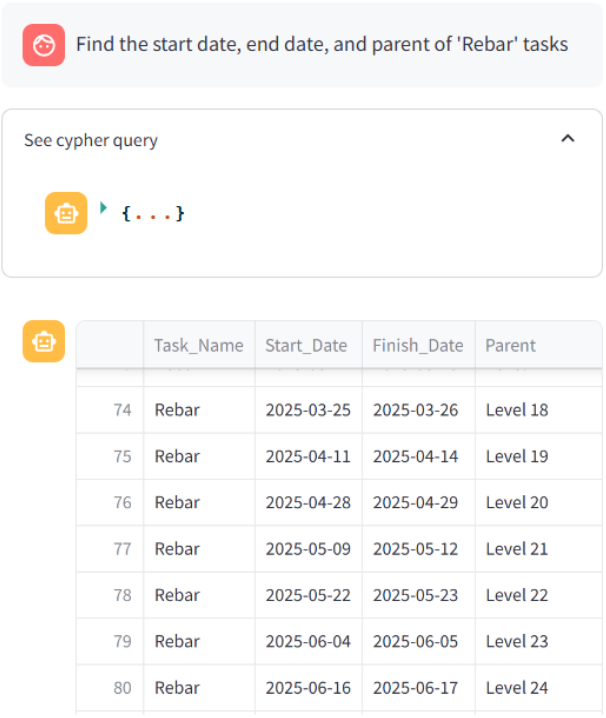


	Task_Name	Start_Date	Finish_Date
3	Pour Concrete to PT Slab	2025-02-12	2025-02-12
4	Rebar	2025-02-13	2025-02-19
5	Verticals	2025-02-13	2025-02-14
6	Install Stair Trays	2025-02-13	2025-02-13
7	Rebar to Stairs Landings &	2025-02-12	2025-02-14
8	Concrete	2025-02-14	2025-02-17

Figure 7: Look ahead plans

Filtering tasks

Filtering tasks based on specific attributes is crucial for efficiently accessing relevant project information. It enables project teams to quickly identify tasks using criteria such as name, start and finish dates, assigned resources, task status, WBS levels, or dependencies, ensuring more effective planning and decision-making. By utilising natural language queries, users retrieved only the most pertinent tasks, allowing them to focus on key project elements while enhancing accessibility and streamlining the scheduling process. Fig. 8 shows one such filtering example. Here, the user was interested in filtering the 'Rebar' activities corresponding to different levels of the building. The application accurately returned start and finish dates and WBS level of all the 'Rebar' activities.



	Task_Name	Start_Date	Finish_Date	Parent
74	Rebar	2025-03-25	2025-03-26	Level 18
75	Rebar	2025-04-11	2025-04-14	Level 19
76	Rebar	2025-04-28	2025-04-29	Level 20
77	Rebar	2025-05-09	2025-05-12	Level 21
78	Rebar	2025-05-22	2025-05-23	Level 22
79	Rebar	2025-06-04	2025-06-05	Level 23
80	Rebar	2025-06-16	2025-06-17	Level 24

Figure 8: Task finder

Retrieving Specific Details of Tasks/ WBSs

Accessing detailed information about tasks and WBS elements is essential for effective construction schedule management. Project teams often need to retrieve specific details such as durations, start and finish dates, assigned resources, dependencies, and progress status to ensure alignment with project goals. In this project, each task node contained attributes such as name, unique ID (UID), start and finish dates, duration, total float, critical path status, and parent WBS. Similarly, WBS nodes stored hierarchical information, including WBS name, UID, start and finish dates, and child tasks. Users accessed these details through natural language queries, which were dynamically converted into Cypher queries using LangChain and Meta Llama 3.1 8B. The retrieved data was displayed in a structured table. Fig. 9 illustrates an example where a user re-

trieved the duration of a task with a simple query, such as "Duration of Install Placing Boom."

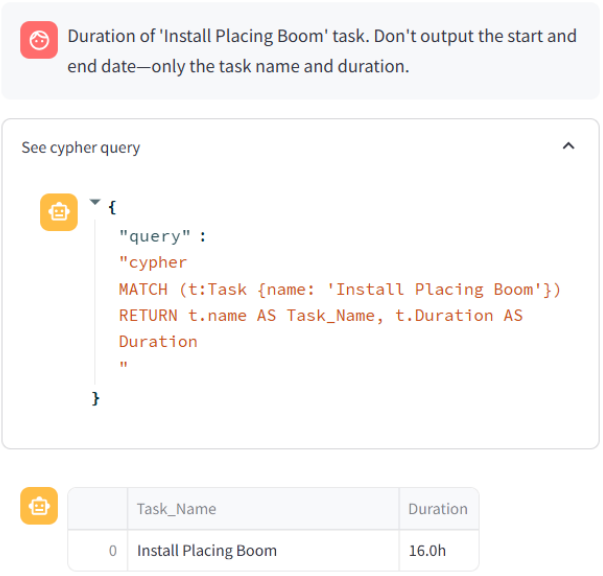


Figure 9: Retrieve specific details of an activity

Figure 10: Predecessor relationship

Analysing Task Dependencies

The succeeding and preceding activities help users analyse task dependencies by retrieving activities that come before (predecessors) and after (successors) a selected task. Identifying these dependencies helps teams assess the impact of delays, optimise workflows, and maintain smooth project execution. The application queried the Neo4j graph database using Cypher to find relationships marked as "SUCCESSOR," where the current task served as either a source (predecessor) or target (successor). The application efficiently identified predecessor and successor tasks, enabling users to track dependencies and assess their influence on the overall schedule. Queries such as "What are the predecessor tasks of Activity X?" or "List the succeeding tasks of Task Y" allowed stakeholders to retrieve relevant information instantly. This output helped project managers understand task sequences, manage constraints, and mitigate scheduling risks. Fig. 10 and 11 show how the application can effectively retrieve preceding and succeeding tasks of a given task.

Discussion

This section examines the qualitative accuracy of the proposed application in retrieving schedule information. Additionally, it provides a comparative analysis of the proposed application and traditional scheduling software, emphasising the operational knowledge required for effective schedule management.

Qualitative Evaluation of the Proposed Application's Accuracy.

A validation process was conducted to qualitatively assess the accuracy of the proposed application by verifying the retrieval of preceding and succeeding activities. The application-generated Cypher query was extracted and executed directly in the Neo4j browser to cross-check the output. Additionally, the same task dependencies were manually reviewed in the scheduling software to ensure consistency. The results confirmed that the application accurately retrieved the preceding and succeeding tasks without discrepancies. Furthermore, the visualisation of the retrieved relationships from the graph database is illustrated in Fig. 12, providing a graphical representation of task dependencies. This visualisation was compared against the application output displayed in Fig. 10 and Fig. 11, both of which showed an exact match with the validated data in Fig. 12. These results demonstrate the reliability of the proposed application in extracting schedule information accurately, reinforcing its capability as an effective tool for project schedule analysis and management.

Comparison of Knowledge Required: Scheduling Software v. Proposed Application

The table 1 compares the level of knowledge required to perform various schedule management tasks using traditional scheduling software versus the proposed AI-powered application. Each scheduling software has a different user interface (UI), and users unfamiliar with a particular tool may require time to navigate and learn its features. Traditional scheduling software demands higher expertise in scheduling principles, tool functionalities, and manual data extraction. Users must navigate complex menus, apply filters, and run queries to obtain insights, which can be challenging without prior experience. Conversely, the proposed AI-driven application simpli-

Figure 11: Successor relationship

Figure 12: Qualitative assessment of information retrieval: Predecessor relationship (left) and Successor relationship (right)

fies schedule interaction using natural language processing (NLP), automated Cypher query generation, and intuitive visualisation. This significantly reduces the learning curve, making schedule analysis more accessible to a wider range of users, including project managers, site engineers, and stakeholders with limited technical expertise.

Future Research Directions

Future research can explore several avenues to enhance the proposed application's capabilities in construction schedule management. One potential direction is the integration of multi-modal AI, combining text, images, and BIM (Building Information Modeling) data to provide a more comprehensive project visualisation. Additionally, improving the natural language processing (NLP) capabilities by fine-tuning large language models (LLMs) on construction-specific datasets can enhance query accuracy and contextual understanding. Another promising area is the incorporation of reinforcement learning to allow the system to improve its responses based on user feedback over time. Cloud deployment and scalability enhancements would enable broader accessibility, supporting real-time collaboration among project stakeholders. Furthermore, extending the application's functionality to incorporate cost and resource management analysis could provide a more holistic project control solution. Future work

Table 1: Comparison of the proposed application with traditional scheduling software in terms of required operational knowledge

Description	Schedulig software	Proposed application
Identification of critical tasks within a specific period	High	Low
Generation of Look-ahead Schedules	High	Low
Identification of critical tasks	Medium	Low
Filtering tasks	Medium	Low
Retrieving Specific details of Tasks/ WBSs	Low	Low
Analysing Task Dependencies	Medium	Low

can also integrate planned resource allocations against actual utilization, allowing for real-time performance monitoring. Delay cause information retrieved from site diaries or progress reports using LLMs (Pal et al., 2024) can be added to the graph nodes, helping to explain variability in project schedules and improve decision-making. Finally, research into explainable AI (XAI) methods could improve transparency and user trust by offering justifications for the AI-generated responses.

Summary and Conclusions

This study presents an AI-powered application for construction schedule management, integrating natural language processing (NLP) and graph-based data representation to enhance project accessibility and efficiency. By leveraging Neo4j for knowledge graph construction, LangChain for Cypher query generation, and the Meta Llama 3.1 8B model for AI-driven interpretation, the system allows users to interact with construction schedules using natural language. Key functionalities include retrieving critical path activities, generating look-ahead plans, identifying task dependencies, and applying filtering options, all visualised through interactive Gantt charts. The user interface offers model selection and two main modules—one for schedule-to-graph conversion and visualisation and another for natural language interaction with the schedule. Currently deployed on a local host, the application aims to transition to a cloud-based platform for broader accessibility in the future.

The developed application enhances construction schedule accessibility by enabling AI-driven natural language querying and graph-based visualisation. By automating schedule analysis and decision-making, it reduces manual effort and improves efficiency in project management.

Future improvements include cloud deployment, advanced AI-driven schedule optimisation, and potential integration with real-time project updates. This application represents a significant step toward automated construction schedule management, making complex project data timely, accessible, and actionable.

Acknowledgments

We would like to thank our industry partners for providing us with access to actual construction data, their engagement with this research, and the visionary approach to data-driven decision-making.

References

Amer, F., Hockenmaier, J., and Golparvar-Fard, M. (2022). Learning and critiquing pairwise activity relationships for schedule quality control via deep learning-based natural language processing. Automation in Construction, 134:104036.

Amer, F., Jung, Y., and Golparvar-Fard, M. (2023). Construction schedule augmentation with implicit dependency constraints and automated generation of lookahead plan revisions. Automation in Construction, 152:104896.

Brown, A. S. (2003). Modeling tough scheduling problems with project management software. In PMI® Global Congress 2003—North America. Project Management Institute.

Calvetti, D., Kifokeris, D., Mêda, P., and Sousa, H. (2024). Human-data interaction as a critical enabler of electronic performance monitoring at construction sites. Journal of Information Technology in Construction, 29:722–749.

Castañeda, K., Sánchez, O., Herrera, R. F., and Mejía, G. (2025). Deficiencies causes in road construction scheduling: Perspectives from construction professionals. Heliyon, 11(2):e41514.

Hong, E., Lee, S., Kim, H., Park, J., Seo, M. B., and Yi, J.-S. (2024). Graph-based intelligent accident hazard ontology using natural language processing for tracking, prediction, and learning. Automation in Construction, 168:105800.

Hong, Y., Hovhannisyan, V., Xie, H., and Brilakis, I. (2021a). Determining construction method patterns to automate and optimise scheduling - a graph-based approach. In Proceedings of the 2021 European Conference on Computing in Construction, volume 2 of Computing in Construction, pages 59–66, Online Conference. European Council on Computing in Construction.

Hong, Y., Xie, H., Agapaki, E., and Brilakis, I. (2023). Graph-based automated construction scheduling without the use of bim. Journal of Construction Engineering and Management, 149(2):05022020.

Hong, Y., Xie, H., Bhumbra, G., and Brilakis, I. (2021b). Comparing natural language processing methods to cluster construction schedules. Journal of Construction Engineering and Management, 147(10):04021136.

Hong, Y., Xie, H., Hovhannisyan, V., and Brilakis, I. (2022). A graph-based approach for unpacking construction sequence analysis to evaluate schedules. Advanced Engineering Informatics, 52:101625.

Hua, Z., Liu, Z., Yang, L., and Yang, L. (2022). Improved genetic algorithm based on time windows decomposition for solving resource-constrained project scheduling problem. Automation in Construction, 142:104503.

Kolosky, C. (2024). Project management software: the pros and cons of traditional tools. `https://tinyurl.com/238c5wcj`. Accessed: 2025-04-01.

Lee, J. and Ahn, S. (2024). Pagerank algorithm-based recommendation system for construction safety guidelines. Buildings, 14(10).

Lee, W. and Lee, S. (2024). Development of a knowledge base for construction risk assessments using bert and graph models. Buildings, 14(11).

Li, S. (1996). New approach for optimization of overall construction schedule. Journal of Construction Engineering and Management, 122(1):7–13.

Li, S., Wang, J., and Xu, Z. (2024). Automated compliance checking for BIM models based on Chinese-NLP and knowledge graph: an integrative conceptual framework. Engineering, Construction and Architectural Management, ahead-of-p(ahead-of-print).

Pal, A., Murguia, D., and Middleton, C. (2024). Automatic inference of construction delays through analysis of weekly progress reports using llms. In Proceedings of the 41st International Conference of CIB W78.

Prieto, S. A., Mengiste, E. T., and García de Soto, B. (2023). Investigating the use of chatgpt for the scheduling of construction projects. Buildings, 13(4).

Ren, R. and Zhang, J. (2022). Construction Procedural Information Extraction from Textual Sources to Support Scheduling, pages 330–339.

ul Hassan, F. and Le, T. (2022). Extraction of Activities Information from Construction Contracts Using Natural Language Processing (NLP) Methods to Support Scheduling, pages 773–781.

Yao, Y., Tam, V. W., Wang, J., Le, K. N., and Butera, A. (2024). Automated construction scheduling using deep reinforcement learning with valid action sampling. Automation in Construction, 166:105622.