EPiC Series in Built Environment

Volume 5, 2024, Pages 549–557

Proceedings of 60th Annual Associated Schools of Construction International Conference

Graph Theory Implementation Towards Artificial Intelligence Generated Construction Schedule

Tulio Sulbaran, Ph.D.
The University of Texas at San Antonio
San Antonio, Texas

Commercial construction scheduling software (such as Primavera Project Planner and Microsoft Project) has been used by construction professionals for close to 40 years. Although those scheduling tools have been significantly modified over the years, the fundamental basis of the interaction between the construction professional and the scheduling software has remained unchanged where the construction professionals enter information and the software makes calculations and displays the schedule information in tables and graphs (such as CPM Diagram and Gantt Chart). However, in the same period, significant computational advances have been achieved leading to the recent explosion of Artificial Intelligence (AI) capabilities. The problem that serves as a point of departure for this research is that the current construction scheduling software workflow is unable to fully benefit from AI capabilities. Thus, the objective and original contribution of this paper is to describe the implementation of graph theory in Python to facilitate the realization of AI-generated construction schedules. The results presented in this paper are very encouraging as a piece in the puzzle of elements needed to create a new generation of construction scheduling software could benefit from AI capabilities and assist construction professionals in the creation of the construction schedule beyond simply making calculations and displaying information.

Key Words: CPM, CPM Motif, Weighted CPM, Node Anomaly, Relationship Prediction

Background & Literature Review

Construction Scheduling, Commercial Software, and Artificial Intelligence

Construction schedules are created by decomposing a project into manageable activities aggregated by level usually represented as a Work Breakdown Structure (WBS). Later, dependencies between those activities are included and represented in a precedence diagram (Pellerin & Perrier, 2019). Given the network and the duration of the activities, the Program Evaluation and Review Technique (PERT) and Critical Path Method (CPM) are used to calculate the earliest and latest start and finish times of the activities (Pellerin & Perrier, 2019).

T. Leathem, W. Collins and A. Perrenoud (eds.), ASC 2024 (EPiC Series in Built Environment, vol. 5), pp. 549–557

Commercially available construction scheduling software (such as Primavera Project Planner and Microsoft Project) has served the construction industry well making calculations and displaying the information in tables and graphs (such as CPM Diagram and Gantt Chart). Given their age, the commercially available scheduling software has been developed in programming languages that precede the recent explosion of Artificial Intelligence (AI) capabilities. For example, Primavera Project Planner was originally launched in 1983 (Winter, 2017) and it was developed in Disk Operating System (DOS) using the programming language Pascal. In 1994, Primavera Project Planner 3 for Windows (Winter, 2017) was released and it was developed using the programming language C++. In 2000, Primavera Project Planner 5 was released and it was developed using the programming language Java. Similarly, Microsoft Project 1 was released in 1984 and it was written in the programming language C (and some Assembly) for Microsoft Disk Operating System (MS-DOS) (Wikipedia, 2023). In 1992, Microsoft Project for Windows was released and it was developed in C++. Since then, all released versions have been developed in C++ (Kolpackov, 2014).

Although Java (used to develop Primavera Project Planner) and C++ (used to develop Microsoft Project) can be used to implement AI, the best and most popular programming language to implement AI is Python (Alba, 2022; White, 2021). Python is an interpreted, cross-platform, high-level, programming language with dynamic semantics (Nagpal & Gabrani, 2019).

In addition to Python AI capabilities, some groundwork has already been completed in Python for construction schedules. For example, NetworkX is a package for modeling, analyzing, and visualizing networks (Platt, 2019) which can be implemented to generate and display node networks (as the CPM Diagram) and bar chats (as the Gantt Chart). Building on top of NetworkX other Python modules have been developed such as PyGantt, CPM-Python, and CheChe-pm (among others). PyGantt is a module to plot event data characterized by a start and an end that naturally applies to scheduling (Komaroc, 2023). CPM-Python is a Python module that implements the Critical Path Method (CPM) algorithm for scheduling a set of project activities (Ramadan, 2024). CheChe-pm is a Python module that seamlessly integrates the outputs of a project schedule into a web application or a dynamic dashboard (Perez, 2023). Although all of these modules are beneficial as they allow users to perform and display construction schedule information, they still do not take full advantage of the recent explosion of AI Capabilities. Thus, the objective and original contribution of this paper centers on describing the graph theory in Python to facilitate the realization of AI-generated construction schedules. Because of its potential to implement AI for construction project scheduling which can significantly enhance the efficiency, accuracy, and overall success of construction projects (Alisher, 2023).

Graph Theory in Construction Scheduling

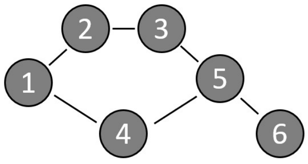
Python has many libraries for almost every need for an AI project and it is preferred by many businesses for many Artificial Intelligence (AI) applications (Nagpal & Gabrani, 2019). Artificial Intelligence has many applications and subfields. One of those subfields of particular interest for construction scheduling is pathfinding. Pathfinding focuses on finding the shortest path between two points or nodes (i.e., Construction Activities) solved through Graph Theory.

Graph theory is a branch of (discrete) mathematics that deals with the way objects are connected (Trinajstic, 2018) involving a binary relation (Harary, 2019). This binary relation (or pairs) can be visualized in a directed graph (or digraph) as per Eq. (1).

$$G = (V, E) \qquad V = \text{Vertices} \qquad E \subseteq \{ (x1,y1) \dashv |(x2,y2) \in V \text{ and } (1 \neq 2) \qquad (1)$$

Graph theory is a logical and systematic approach with very well-documented applications (Rao, 2007) that can be used to represent a wide variety of real-world entities, such as social networks, transportation networks, computer networks, data structure, routing, and construction scheduling. Graph theory applications help in identifying attributes, and offer a visual appraisal of the attributes and their interrelations (Rao, 2007). The visual appraisal can be represented mainly as Adjacency Matrices, and/or Adjacency lists (Chakraborty et al., 2018).

- Adjacency Matrices: the objects are represented as an n X n square Matrix M where n represents the number of vertices/nodes present in the graph. The edge connecting those vertices are represented by Mij with a value of 1 if the two vertices/nodes are connected and a value of 0 if the two vertices are not connected. It is very simple but it has the disadvantage that has a lot of zeros wasting a lot of space as shown in Figure 1.
- Adjacency List: Represents the same vertices and edges, but all zeros (representing no connections) are eliminated and only the vertices with connecting edges are represented as neighboring nodes as shown in Figure 1.



Graphical Representation

	1	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	0	1
2	1	0	1	0	1	0
3	0	1	0	0	1	0
4	1	0	0	0	1	0
5	0	1	1	1	0	1
6	0	0	0	0	1	0

Adjacency Matrix

Neighboring nodes of **1**: **2, 4**
Neighboring nodes of **2**: **1, 3**
Neighboring nodes of **3**: **2, 5**
Neighboring nodes of **4: 1, 5**
Neighboring nodes of **5**: **3,4,6**
Neighboring nodes of **6**: **5**

Adjacency List

Figure 1. Graph theory network, adjacency matrix representation, and adjacency list representation.

Methodology

A mixed method case study research methodology was followed in this research. This methodology was used because it combines the strengths of the quantitative and qualitative methods in the same study (Molina-Azorin & Cameron, 2010). This research methodology was implemented using three main components: 1- Graph Theory (as the theoretical framework), 2- Jupyter Notebook (as programming workbench), and 3- Construction schedule case study (illustrative example) as shown in Figure 2.

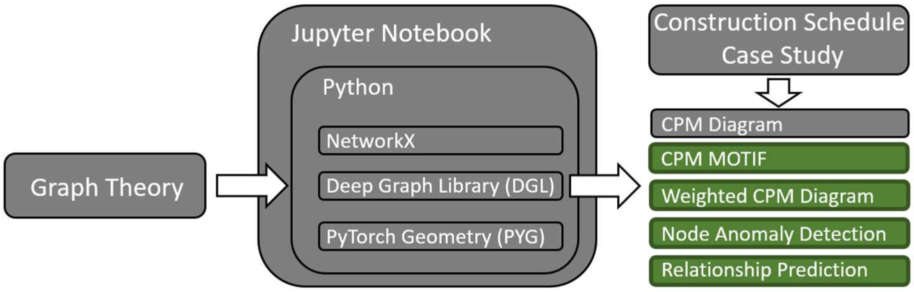


Figure 2. Research Methodology Workflow.

One of the reasons for using Python (as a programming workbench) in this research project and future AI implementations is the availability of a number of graphic pathfinding libraries that can be used to implement Graph Theory in AI. These libraries allow to create, visualize, manipulate, and analyze graph data. Three of those libraries were used and the following is a brief description of each of them:

- NetworkX: comprehensive Python library for the creation, manipulation, analysis, and study of the structure, dynamics, and functions of complex graphs and networks.
- Deep Graph Library (DGL): graph library for deep learning. It supports multiple Artificial Intelligence (AI) deep learning frameworks, including PyTorch, TensorFlow, and MXNet.
- PyTorch Geometric (PyG): graph library for geometric deep learning that includes a number of node anomaly detection algorithms, such as graph autoencoders, graph diffusion models, and graph kernel methods.

The Construction schedule case study was used as an illustrative example, beginning with the creation of the traditional CPM Diagram. More importantly, this was followed by showcasing the capabilities of AI in Python through CPM MOTIF, Weighted CPM diagram, Node anomaly detection, and relationship prediction (highlighted in Figure 2). None of these (to the knowledge of the author) are currently possible in the commercially available scheduling software, but they are key to facilitate the implementation of AI to help generate construction schedules in the future.

Results

The following are the results in accordance with the research methodology shown in Figure 2.

CPM Diagram

PERT and CPM are widely used by construction professionals in planning and controlling both small and large project and all type of projects (Ba'Its et al., 2020). Therefore, before showcasing the capabilities that are not currently available in the commercial construction scheduling software, it was important to create a traditional CPM Diagram for three sets of activities and relationships corresponding to one singular project (two of them shown in Figure 3). It is worth noting that although, the schedules are for one singular project, they have different activities and different relationships demonstrating the diversity in the preparation of real schedules by the construction professionals.

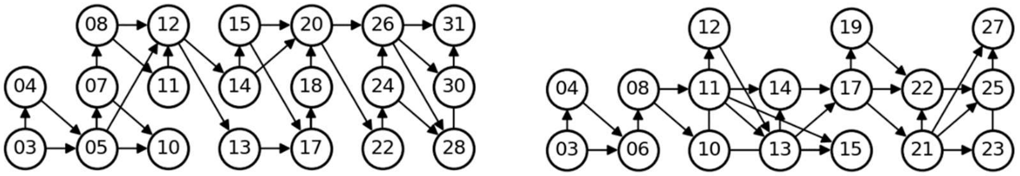


Figure 3. CPM Diagrams based on three sets of activities and relationships for one project.

CPM Motif

A network motif is an interconnected pattern within a network. It is a subgraph formed by a small number of n nodes occurring in a network at numbers that are significantly higher than its average number in an ensemble of randomized networks (Liu et al., 2021). An analysis of the three CPM diagrams shows that MOTIF exists with any two schedules of the case study as well as all three schedules in the case study. For example, activity "04" was preceded by activity "03" in all three schedules as shown in Figure 4. Thus, it could be inferred that to build this project not only activities 03 and 04 shall be included but also the relationship among them shall always exist. This type of Motif analysis is not currently possible with commercial scheduling software.

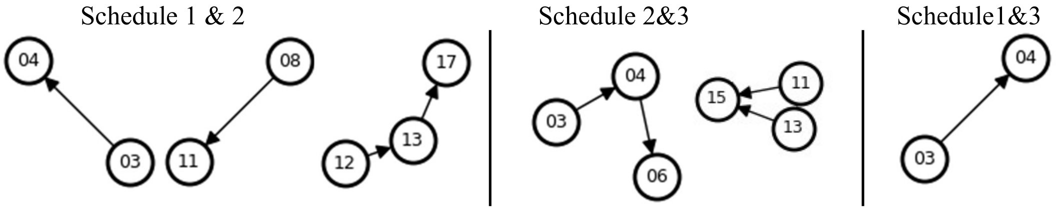


Figure 4. Motif in Construction Schedules.

Weighted CPM Diagram

One of the benefits of AI is automation (Pan & Zhang, 2021) to achieve this automation human knowledge must be collected and consolidated. Weighted CPM diagrams consolidate the knowledge of multiple humans about projects. Figure 5 shows the weighted CPM Diagram that considers three schedules for one unique project. The node size indicates its appearance frequency in the construction case study. Where larger nodes, denote higher frequency. Likewise, the thickness of the relationship indicates the appearance of that relationship among the schedules. Where thicker relationships, indicate more prevalent relationships. It's worth mentioning that activities '03','04','08','10','11',13', and '15' were used the most in the schedules. Therefore, it could be inferred that the likelihood of including those activities in future schedules is significantly higher than the activities that were included less frequently in previous schedules.

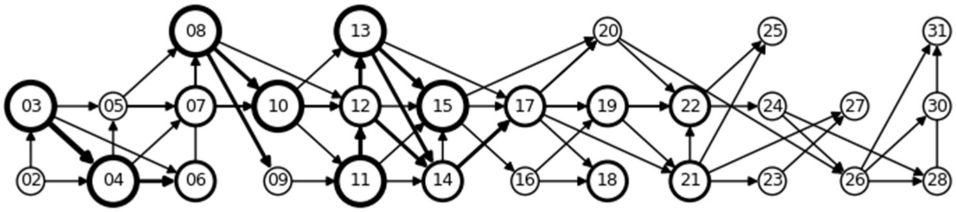


Figure 5. CPM Diagram Weighted Nodes and Relationships.

Node Anomaly Detection

Node anomaly detection consists of the AI identifying nodes that deviate from the generally expected nodes in a network. This is of particular interest to automatically check construction schedules to highlight activities that are not expected to be in the schedule or are expected to be at a different location in the schedule. This is a hard task for humans given the complexity and variability of the schedules, but the AI can do this through multiple methods such as Graph Kernel, Graph Clustering, or Feedforward Network. The Feedforward Network was implemented here to demonstrate the concept of node anomaly detection. As shown in Figure 6, the activity “AA” is highlighted in red indicating that it is an activity that is not expected to be in the schedule. The activity “04” is highlighted in orange indicating that it was expected to be at a different location in the schedule.

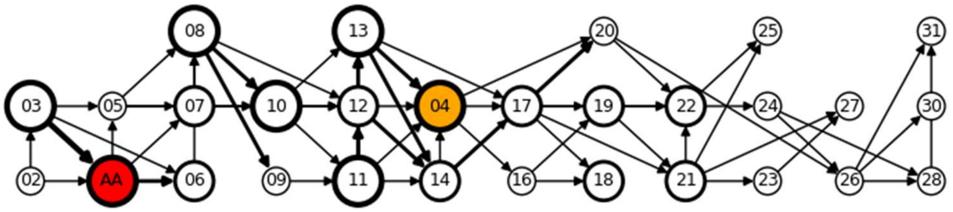


Figure 6. CPM Diagram with Nodes Anomaly.

Relationship Prediction

Relationship prediction in AI has multiple scopes. One of those scopes is the forecasting of relationships between nodes that do not exist based on current relationships between nodes. Implementation of this scope of relation prediction is widely used in social networks (i.e., Facebook makes friend recommendations using relationship prediction). Relationship predication scope is also the identification of errors or omissions of relationships in a network. This is of interest to automate the checking of construction schedules to identify missing relationships between activities or identify relationships between activities that should not exist. As shown in Figure 7, the relationships between activities “08” and “09” as well as ”08” and “10” are highlighted in blue to indicate they were expected relationships that were most likely omitted from the schedule. Likewise, the relationships between activities ”25” and “27” as well as “25” and “31” are highlighted in red to indicate that they were unexpected relationships that were most likely added in error to the schedule.

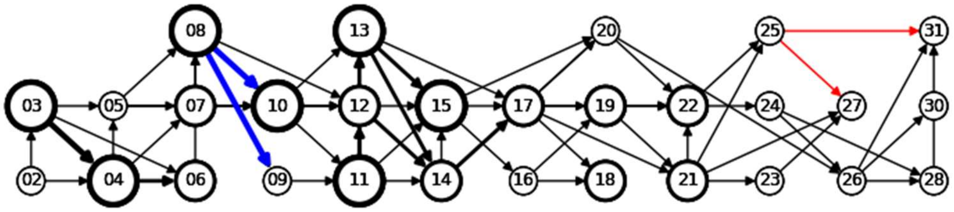


Figure 7. CPM Diagram with Relationship Predication.

Summary

Although for several decades, the fundamental interaction between the construction professional and the commercially available scheduling software has remained unchanged, significant computational advances have been achieved leading to the recent explosion of Artificial Intelligence (AI) capabilities. This paper's objective and original contribution was to describe the implementation of graph theory in Python to facilitate the realization of AI-generated construction schedules. Python was used because it is preferred by many businesses for many Artificial Intelligence (AI) applications.

The results of this research demonstrated that by implementing graph theory using Python it is possible to create the traditional CPM Diagram. But, more importantly, it is possible to go beyond what is possible to be done with commercially available software, such as to identify CPM Motifs among multiple schedules, create weighted CPM diagrams, detect node/activity anomalies, and predict relationships between nodes/activities. The results presented in this paper are essential elements needed to create a new generation of construction scheduling software where AI will assist construction professionals in the creation of the construction schedule beyond simply making calculations and displaying information.

It is also important to acknowledge the limitations of this study including (but not limited to): 1- The implementation of the graph theory using Python was done under the premise that the construction activities could be unambiguous, in the study they were represented by numbers that have the same interpretation, 2- Construction project activities descriptions vary significantly which presents a challenge for the proposed approached, 3- Construction activities depending on the project or the scheduler could be done in different stages of the projects and therefore have significant different predecessor and successors, and 4- Accuracy of this model is yet to be determined.

Building up this research and the acknowledging its limitations, it will be valuable to conduct future research partnering with other researchers to: 1- Secure the pre-construction documents of one real simple project (i.e., Residential, Commercial, Mixed-used, Institutional, Industrial, or Heavy civil), 2- Engage multiple schedulers to create multiple schedules for that one real project, 3- Codify the activities in the multiple schedules using a common taxonomy (i.e. Master format, Uniformat, Omniclass, etc), 4- Run the schedules through the graph theory AI Python code illustrated in this paper, and 5- Measure accuracy of the AI to perform the task indicated above. Other measures such as cluster coefficient, degree distribution, and network density to investigate the relationship patterns in the construction schedule would also be worthy of further investigation.

References

Alba, L. (2022). Best Programming Language for AI. Sitepoint. https://www.sitepoint.com/best-programming-language-for-ai/

Alisher, A. (2023). Leveraging Artificial Intelligence for Construction Project Scheduling. Genius Repository, 22, 33–36.

Ba'Its, H. A., Puspita, I. A., & Bay, A. F. (2020). Combination of Program Evaluation and Review Technique (PERT) and Critical Path Method (CPM) for Project Schedule Development. International Journal of Integrated Engineering, 12(3), Article 3.

Chakraborty, A., Dutta, T., Mondal, S., & Nath, A. (2018). Application of Graph Theory in Social Media. International Journal of Computer Sciences And Engineering, 6, 722–729. https://doi.org/10.26438/ ijcse/v6i10.722729

Harary, F. (2019). Graph Theory (on Demand Printing Of 02787). CRC Press. https://doi.org/ 10.1201/9780429493768

Kolpackov, B. (2014,). Bonus Talk: C++ in MS Office [C++Conference]. https://cppcon.org/bonus-talk-cxx-in-ms-office-2014/

Komaroc, O. (2023). PyGantt/PyGantt. https://github.com/PyGantt/PyGantt

Liu, L., Zhao, M., Fu, L., & Cao, J. (2021). Unraveling local relationship patterns in project networks: A network motif approach. International Journal of Project Management, 39(5), 437–448. https://doi.org/10.1016/j.ijproman.2021.02.004

Molina-Azorin, J., & Cameron, R. (2010). The Application of Mixed Methods in Organisational Research: A Literature Review. Electronic Journal of Business Research Methods, 8, 95–105.

Nagpal, A., & Gabrani, G. (2019). Python for Data Analytics, Scientific and Technical Applications. 2019 Amity International Conference on Artificial Intelligence (AICAI), 140–145. https://doi.org/10.1109/ AICAI.2019.8701341

Pan, Y., & Zhang, L. (2021). Roles of artificial intelligence in construction engineering and management: A critical review and future trends. Automation in Construction, 122, 103517. https://doi.org/10.1016/ j.autcon.2020.103517

Pellerin, R., & Perrier, N. (2019). A review of methods, techniques, and tools for project planning and control. International Journal of Production Research, 57(7), 2160–2178. https://doi.org/10.1080/ 00207543.2018.1524168

Perez, L. F. (2023). Efficient Project Scheduling with Python. The Critical Path Method. Medium. https://medium.com/@luisfernandopa1212/efficient-project-scheduling-with-python-the-critical-path-method-19a3f8235f91

Platt, E. L. (2019). Network Science with Python and NetworkX Quick Start Guide: Explore and visualize network data effectively. Packt Publishing Ltd.

Ramadan, M. (2024). CPM-python. https://github.com/cppcoders/cpm-python

Rao, R. V. (2007). Decision Making in the Manufacturing Environment: Using Graph Theory and Fuzzy Multiple Attribute Decision Making Methods (Springer Series in Advanced Manufacturing) (1. Aufl., Vol. 2). Springer Verlag London Limited. https://doi.org/10.1007/978-1-84628-819-7

Trinajstic, N. (2018). Chemical Graph Theory. Routledge.

White, E. (2021, July 7). Top 9 Programming Languages For Artificial Intelligence. BairesDev Blog: Insights on Software Development - Tech Talent. https://www.bairesdev.com/blog/top-languages-for-ai/

Wikipedia. (2023). Microsoft Project. In Wikipedia. https://en.wikipedia.org/w/index.php?title= Microsoft_ Project&oldid=1158971124

Winter, R. (2017). The History of Primavera Scheduling Software. Ron Winter Consulting LLC. https://www.ronwinterconsulting.com/The_History_of_Primavera_Scheduling.pdf