2021 European Conference on Computing in Construction
Ixia, Rhodes, Greece
July 25-27, 2021

DETERMINING CONSTRUCTION METHOD PATTERNS TO AUTOMATE AND OPTIMISE SCHEDULING – A GRAPH-BASED APPROACH

Ying Hong[1], Vahan Hovhannisyan[2], Haiyan Xie[3], and Ioannis Briklakis[1]
[1]University of Cambridge, Cambridge, United Kingdom
[2]nPlan, London, United Kingdom
[3]Illinois State University, Illinois, United States

ABSTRACT

Construction projects have been experiencing project delays for decades. As an executive guide to construction activities, construction schedules can mitigate delay risks and are essential to project success. Yet, creating a quality construction schedule is often the outcome of experienced schedulers, and what makes it harder is the fact that historic information including decision reasoning was not documented and disseminated for future use. This study proposes a graph-based method to find the time- and risk-efficient construction method patterns from historic projects to help schedulers improve productivity and accuracy. The method leverages schedule data (including activity names, Work Breakdown Structure, and start and finish date) that were obtained from a Tier-1 contractor for this study. The method was validated for excavation activities. The results indicate that the most time-efficient excavation activities can be done in 0.6% of total project time. The proposed method can help industry professionals standardise scheduling guidelines and automate the generation of construction schedules for critical subtasks.

Keywords: Construction schedules, construction method pattern, graph-based classification

INTRODUCTION

The question that this research addresses is how to find construction method patterns and use them to optimise construction planning for new projects. This study defines a construction method pattern as a common activity sequence for a subtask, sometimes also known as 'method logic' in practical scheduling software (e.g. Primavera). Frequency is the key feature that distinguishes a construction method pattern from an activity sequence. Method patterns are generalised and can be deployed into the scheduling of other projects; hence, method patterns should not be constrained by project characteristics. Another key term is the construction schedule, which is defined as the timetable for a project, programme or portfolio, which shows how the work will progress over a period of time given limited resources and uncertainty (Association for Project Management, 2017).

The construction industry has a global market share worth $10 trillion (Blanco *et al.*, 2016); however, construction projects are often plagued by project delays and cost overruns. 98% of megaprojects suffer cost overruns of more than 30% and 77% are at least 40% late (Changali *et al.*, 2015).

A construction schedule, as the executive guide to construction activities, is particularly important to manage the project effectively and mitigate potential delay risks. However, producing quality schedules requires schedulers to have comprehensive knowledge to identify potential risks and allocate time contingencies to mitigate against these risks (De Snoo *et al.*, 2011), in particular when addressing uncertainty and allocating time contingency (Brockmann, 2012). Therefore, learning from historic projects enables schedulers (particular inexperienced schedulers) to produce quality schedules in future projects.

However, this is impractical and rarely done. Inexperienced schedulers rarely have the time and patience to study raw schedule data and derive patterns of their own, due to the size of the datasets involved and the trivial nature fo the task. This is an area that machine learning would be expected to perform well. However, most scheduling data available are too unstructured for direct use in current machine learning systems, often yielding very poor performence.

A construction project often contains repetitive activities. However, many activity names with the same meaning are written differently with substantial identifying information often ommitted. This is because they are written to be read by humans, who are able to infer the ommitted content from a combination of experience and the context in which the activity is present in. This study proposed a graph-based approach that has the potential to address this challenge and encapsulate construction schedules with repetitive activities in construction method patterns.

For instance, the example shown in Figure 1 presents an excavation schedule. The excavation process of Base 3 and Base 4 are the same, starting from excavation followed by shutter, pouring, curing and striping concrete, and ending with backfill. Base 5 was scheduled slightly different. Blinding and placing reinforcement were put as separate activities. This example highlights the most common problem in the current scheduling naming practice: repeated activities are described in different levels of detail.

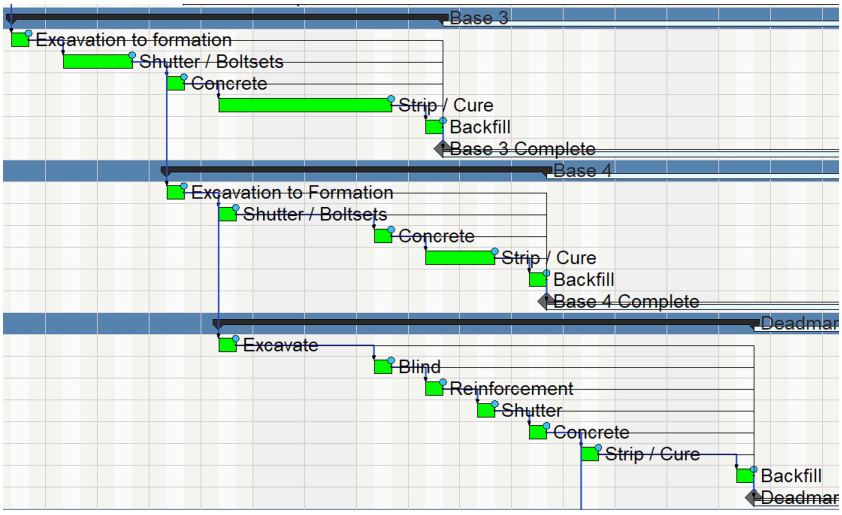

Figure 1: Motivating Case

The next section summarises literature in deriving construction method patterns and introduces the concept of a graph. This is followed by an explanation of the proposed graph-approach, and then the research methodology and results. Discussion and conclusions are presented at the end.

LITERATURE REVIEW

There are overall three types of approaches to find construction method patterns: Case-Based Reasoning (CBR), model-based scheduling, and ontology-based.

CBR is an approach that recalls an earlier case comparable to the present case and uses that to suggest a solution to the new problem (Kolodner, 1992). Benjamin et al. (1990) built a knowledge-based prototype using CBR to identify precedence relationships and the Work Breakdown Structure (WBS) by mimicking the process of an expert's decision making. Muñoz-Avila et al. (2001) introduced a case-based planning algorithm to generate project plans using previous cases. The follow-up research integrated CBR to help project planners create WBS more efficiently (Muñoz-Avila *et al.*, 2002). CBR requires the situational context of both the earlier situation and the present situation. CBR is an effective approach to applying decisions made in previous similar situations, but not to find generalised construction method patterns.

Model-based scheduling approaches find construction method patterns from building models. A typical example of model-based automatic scheduling is MOCA, which formalises construction methods based on product models (Fischer *et al.*, 1994). Wu et al. (2010) used a similar approach to look at the construction logic of bridge construction. Fırat et al. (2007) broadened the scope of model-based automatic scheduling to resource and cost allocation. The follow-up research (i.e. Firat et al. (2009)) interacts with automatically generated schedules with project managers to improve the scheduling quality and customised it to project constraints. However, model-based scheduling approaches heavily rely on the availability and the level of details of building models. The current application of Building Information Modelling (BIM) in practice is not sufficient to deploy model-based scheduling approaches if its availability throughout the lifecycle and the level of detail is considered.

Ontology-based approaches use an ontology-based or rule-based framework to configure construction processes. Benevolenskiy et al. (2012) developed a framework that combines ontology-based with rule-based process modeling to configure the construction process for 'structural concrete subtasks. Cao and Hall (2020) developed an ontology to configure product information for modular construction by enriching process information. Ontology-based approaches provide the flexibility to configure and modify construction processes. However, it is a labour intensive process to generalise the configuration of construction processes across all types of projects.

There are also other approaches to find construction method patterns. Pan (2008) used fuzzy Analytical Hierarchy Process to select the suitable bridge construction method. Hegazy and Kamarah (2008) optimised repetitive scheduling for high-rise construction using genetic algorithms. However, these approaches focus on small datasets and are project-based.

What is a Graph?

A graph G(V,E) is comprised of vertices V and edges E (Diestel, 2017). There are two types of graphs: undirected and directed. An undirected graph connects vertices with bidirectional edges; whereas, a directed graph connects vertices with uni-directional edges (Sedgewick and Wayne, 2011). Two connected vertices can be transversed in either direction in an undirected graph, but cannot in a directed graph.

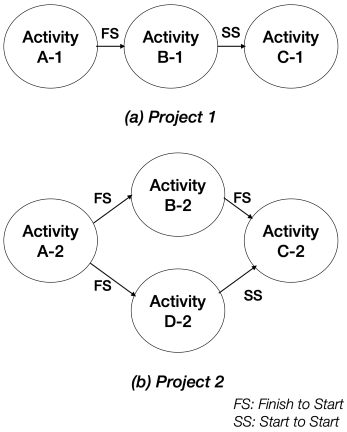

Figure 2: Schedules = Directed Graphs

In a construction project schedule, a vertex is an activity and an edge is the logical relationship between two activities. A construction project schedule is a directed graph since the construction process is not bidirectional. Figure 2 depicts the above description.

Node classification is one of the classic applications of graph learning, which predicts the label associated with all the nodes. Examples of node classification include classifying social network data (Bhagat *et al.*, 2011), classifying the function of proteins in the interactome (Hamilton *et al.*, 2017), and classifying the topic of documents based on hyperlink or citation graphs (Kipf and Welling, 2017). Available graph-based node classification methods include graph kernel methods and Graph Neural Networks (GNNs). Weisfeiler-Lehman kernel is a typical graph kernel method that approximates graph isomorphism to check whether two graphs have the same label set (Shervashidze *et al.*, 2011). GNNs is a general framework for defining deep neural networks on graph data.

The advantage that distinguishes GNNs from other classifiers, particularly Multi-Layer Perceptrons, is permutation invariance. Permutation invariance is a property that processes inputs regardless of their arbitrary ordering (Hamilton, 2020). This characteristic helps graph-based processes because construction activities are connected with a logical order. Therefore, GNNs are the theoretically best approach to categorise construction schedule data.

A Graph Convolution Network (GCN) is a type of GNNs designed for graph datasets. It is a simple and effective method for capturing high order neighbourhood information (Kipf and Welling, 2017). Therefore, the proposed graph-based method is built based on GCN.

Knowledge Gaps, Objectives and Research Questions

Previous studies used case-, model-, and ontology-based methods to find construction method patterns. However, these methods have their limits in practical use. Case-based methods have the ability to forecast decisions based on the context of both the earlier case and the present case. Model-based methods are limited by the often unavailability of detailed project information models at the project planning stage. Ontology-based methods are limited in their ability to generalise construction method patterns across project types. This study aims to address these limitations by leveraging graph-based approaches to categorise construction schedule data.

This study proposes a graph-based method to identify construction method patterns, and hence to automate and optimise construction planning for new projects. This study aims to answer the question: How to find the most time- and risk-efficient construction method patterns? The following objectives are summarised to achieve the aim:

- Extract construction activities' feature representation,
- Identify construction method patterns,
- Find the most time- and risk-efficient construction method pattern.

PROPOSED METHOD

This section describes the proposed graph-based method to classify construction method patterns. Figure 3 summarises the steps of the proposed method starting from data pre-processing to formulating the feature matrix and the adjacency matrix before building a graph. The proposed method was validated on earthwork and foundation construction activities because they are common to most construciton projects and their first onsite step. The GCN model consists of an input layer, a hidden layer, and an output layer. The input of a GCN model comprises of a feature matrix and an adjacency matrix.

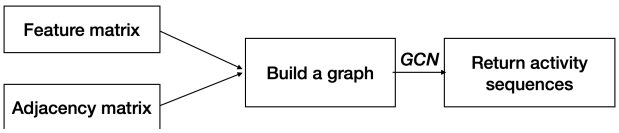

Figure 3: Flowchart of the Proposed Method

Step 1 – Formulating feature matrix. A feature matrix describes the features of nodes in a graph. In the case of this study, a feature matrix describes the features of all the schedule activities; i.e. numerical features (i.e. activity duration) and text features (i.e. activity names and its WBS).

This study used two metrics to describe the numerical features of an activity - relative activity duration and long duration. Relative activity duration is the ratio of an activity duration to the project duration (estimated as Eq. (1)). It is used because the time needed to complete an activity is affected by the project size. Laying a shallow foundation for an independent house is quicker than laying a deep foundation for a skyscraper.

$$RAD_i = D_i / \sum_i D_i \qquad \text{Eq. (1)}$$

where i is the i^{th} activity, D_i is the duration of i^{th} activity

Log duration is the logarithm to activity duration used to account for zero-duration activities (estimated as Eq. (2)). Activities such as milestones are important to method pattern classification because they signal the commencement and completion of a sequence of activities. However, durations of these milestones are

usually zeros. This study used the logarithm to activity duration to avoid infinite values when building the adjacency matrix.

$$LD_i = log(D_i + 1) \quad \text{Eq. (2)}$$

This study used word embedding to represent text features of an activity numerically to integrate text features and numerical features. Word embedding maps words into the vector space by training language models. *word2vec* and *FastText* are two typical examples of language models developed by Mikolov et al. (Mikolov *et al.*, 2013) and Bojanowski et al. (Bojanowski *et al.*, 2017). The architecture of *word2vec* takes one-hot encoded words as input, passes the input to a linear projection layer and a non-linear hidden layer for training, and returns the probabilities of target words as output (Mikolov *et al.*, 2013). The architecture of *FastText* is quite similar to the *word2vec*, but takes n-gram features as input (Joulin *et al.*, 2017); thus the vector of a word is made of the sum of its n-gram (e.g., the word vector 'rebar' is the sum of the vectors of 'reb', 'eba', 'bar'). This study tested *FastText* on a set of test words that are frequently seen in the construction context and summarised based on our domain knowledge.

Text features of an activity in this study include the activity name and the name of its WBS level to capture more contextual information of an activity. *FastText* trains weights of the hidden layers by minimising the cross-entropy loss using gradient backpropagation (Mikolov *et al.*, 2013). Weights of the hidden layer form the embedding matrix and are stored locally to produce embedding for construction activity names.

Once a *FastText* is finely tuned, activity names and WBS were fed into the tuned model to derive word vectors. *FastText* represent words in vectors; whereas construction activity names usually comprise multiple words. This study counted the frequency of each word in every construction activity, obtained the word vector of each word via feeding words into the tuned *FastText* model, and then sums the vector of an activity.

Step 2 – Dimension Reduction. This study employed Principal Component Analysis (PCA) for dimensionality reduction. This suppresses noise and speeds up the computation of pairwise distances between samples once all features have complied. PCA is a statistical procedure used to transform attributes of a dataset into a new set of uncorrelated variables called Principal Components (PCs), while retaining the variability of the original dataset (Howley *et al.*, 2006). This study fed the feature matrix obtained in the last step into a PCA model to achieve 99% explained variance and represented a sparsed feature matrix with condensed PCs.

Step 3 – Formualting adjacency matrix. An adjacency matrix describes the connection of nodes in a graph and each node has a self-loop that connects a vertex to itself (Kipf and Welling, 2017)(Sedgewick and Wayne, 2011). An adjacency matrix describes how each activity is linked with others in the context of this study. Given the nature of construction schedules, this study incorporated two types of information in an adjacency matrix: the graph structure of a project schedule and logic links between activities (i.e. Finish to Start). We assigned a weight to each edge based on the frequency of its logic link in the project. Figure 4 shows the corresponding adjacency matrix of Figure 2.

	A-1	B-1	C-1	A-2	B-2	D-2	C-2
A-1	**1**	0	0	0	0	0	0
B-1	**0.5**	**1**	0	0	0	0	0
C-1	0	**0.5**	**1**	0	0	0	0
A-2	0	0	0	**1**	0	0	0
B-2	0	0	0	**0.75**	**1**	0	0
D-2	0	0	0	**0.75**	0	**1**	0
C-2	0	0	0	0	**0.75**	**0.25**	**1**

Figure 4: Adjacency Matrix – Example

RESEARCH METHODOLOGY

5,050 project schedules were collected from a Tier 1 contractor in the UK. They contain work breakdown structures, construction activities and their durations, early start dates and early finish dates. Collected data are in '.xml' format and then parsed to 'csv' format using our project partner's library (nPlan, 2019). This study further selected 27 completed projects which have a planned date and an actual date. These 27 projects are very diverse as shown in Figure 5.

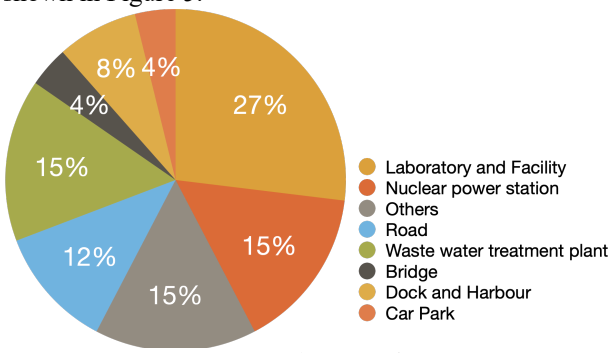

Figure 5: Type diversity of projects

The collected project schedules are unlabelled data. The authors manually and randomly labelled 23% of the activities based on the classification system found in the Standard Methods of Measurement (The Royal Instituion of Charted Surveyors, 1988). This study used GCN as a semi-supervised learning method to learn from labelled activities to predict unlabelled activities. The trainning set has both labelled and unlabelled activities since this study takes a project schedule as a sub-graph and all project schedules in the training set as a whole graph. Therefore, there are three types of activities: labelled activities in the training set (a.k.a. train nodes), unlabelled activities in the training set (a.k.a. transductive test nodes), and unlabelled activities in the testing set (a.k.a. inductive test nodes) (Hamilton, 2020). Both training nodes and transductive testing nodes will be used in GCN training, but the loss function is only computed on train nodes (Hamilton, 2020).

This study pre-processed activity names to secure accuracy before feeding activity names into analysis

models. The pre-processing steps include tokenisation, lemmatisation, stemming, and removing stop words. Tokenisation is a process that transforms text into tokens which are readable in a computer language (Manning *et al.*, 2015). Lemmatisation and stemming are used to reduce the effects of inflectional form and words' morphology (Habash *et al.*, 2009). Stop words (e.g. 'and', 'the') and punctuation was removed to eliminate unmeaningful words.

RESULTS

GCN Input

This study trained a *FastText* model on 5,050 project schedules with a vocabulary size at 42,420. There are three main hyperparameters in *FastText*: embedding size and learning rate (Rehurek and Sojka, 2010). Embedding is a mapping from a word to a vector and embedding size is the dimension of the vector (Rehurek and Sojka, 2010). Learning rate is a parameter that determines how quick a model approaches the optimal solution (Goodfellow *et al.*, 2016). A small learning rate may get stuck on a suboptimal solution; whereas, a large learning rate may lead to an oscillating performance and never converge (Goodfellow *et al.*, 2016). Length of character n-grams is another hyperparameter that decides how many neighbouring words are considered (Joulin *et al.*, 2017).

The optimal *FastText* model was determined by the accuracy in finding similar words. This study pre-defined a value range for each hyperparameter and computed the accuracy of finding similar words for the test words. The optimal language model is the one with hyperparameters that demonstrated the highest accuracy in finding similar words for the test words. The optimal *FastText* model has the following characteristics: 60 embedding size, 7 negative samples and 7 character n-grams.

The next step is deriving text features of construction activities by feeding activity names and WBS into *FastText* model, and then integrating relative activity duration and log duration before reducing feature dimensions. Figure 6 shows the explained variance of PCA over the numbers of PCs. This study used the oldest method – the elbow method - to determine the numbers of PCs. In the elbow method, the moment when the cost function value drops dramatically and reaches plateau afterwards indicates that the ideal is reached (Kodinariya and Makwana, 2013). Five PCs are sufficient according to elbow method.

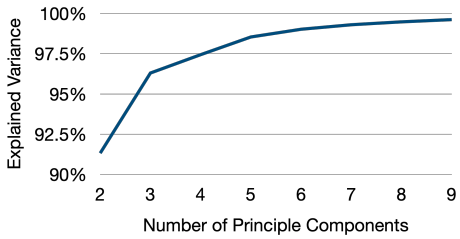

Figure 6: Explained Variance - PCA

GCN Tuning

The next step is GCN tuning and this study tuned two hyperparameters: learning rate and the size of the hidden layer. The previous section described the learning rate and its impacts. The number of neurons in the hidden layer is another hyperparameter. A deep hidden layer may lead to overfitting; whereas a shallow hidden layer may be underfitting (Goodfellow *et al.*, 2016). This study used the hyperparameters tuned by (Kipf and Welling, 2017) as the baseline to benchmark the tuning performance. The baseline model has 16 neurons in the hidden layer and the learning rate is 0.01.

Figure 7 shows the train accuracy and test accuracy of different hyperparameters, and the results indicated that 70 neurons in the hidden layer and 0.02 learning rate is the most accurate combination. Other three key characteristics that describe a GCN are optimiser, loss function, and regularisation. This study used Adam optimisation to optimise the cross-entropy loss at a drop-out rate of 0.3.

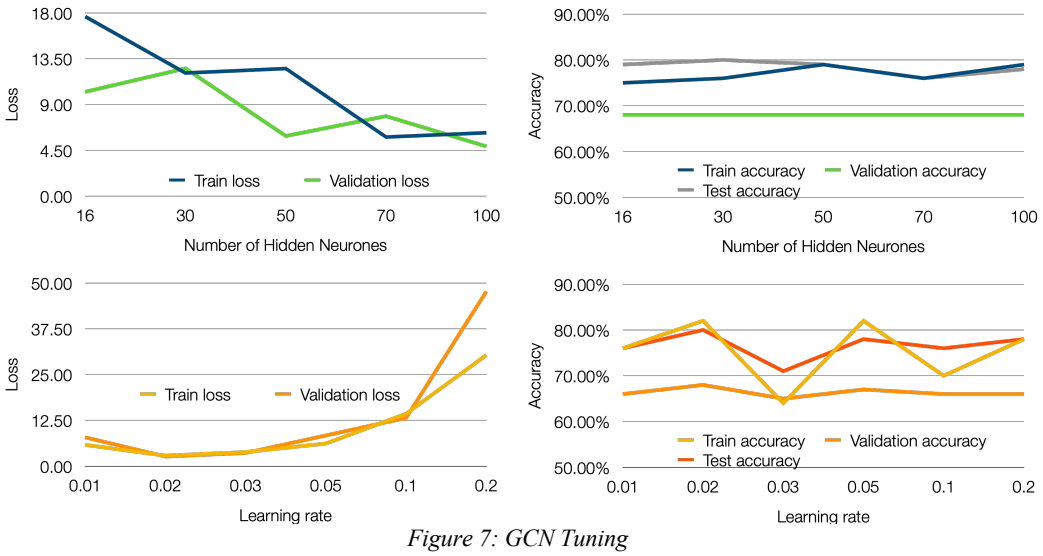

Figure 7: GCN Tuning

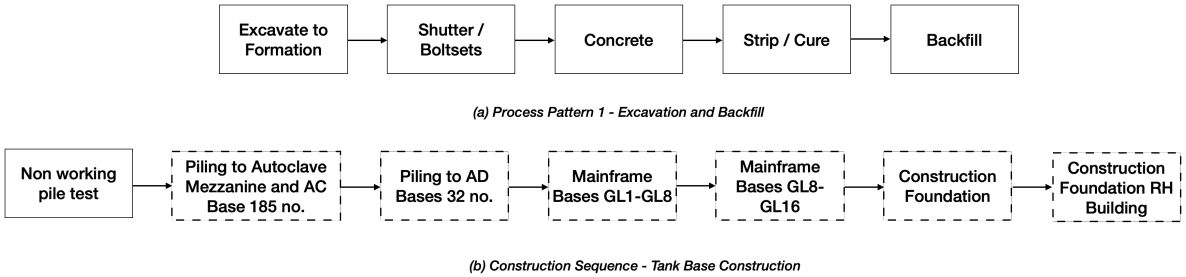

Figure 8: Method patterns – Results

Table 1: Cosine Similarity of Activities in Tank Base Construction Sequence

	Piling to Autoclave Mezzanine and AC Base 185 no.	Piling to AD Base 32 no.	Mainframe Bases GL1-GL8	Mainframe Bases GL8-GL16
Piling to Autoclave Mezzanine and AC Base 185 no.	1	0.99	0.98	0.97
Piling to AD Base 32 no.	0.99	1	0.99	0.98
Mainframe Bases GL8-GL16	0.98	0.99	1	1
Mainframe Bases GL8-GL16	0.97	0.98	1	1

Foundation Construction Method Patterns

This study found 1 method pattern and 34 sequences in total. Figure 8 (a), an excavation and backfill method pattern, is the most common method pattern found that has shown 4 times. It starts from *'Excavate to Formation'*, followed by *'Shutter'*, *'Concrete'* (or *'Pour concrete'*), and *'Strip / Cure (concrete)'*, then ends with *'Backfill'*. This method pattern was scheduled in two different timeframes. One needs 6 days to complete, and the relative duration of this sequence is 0.006. Another one needs 13 days to complete, and the relative duration of this sequence is 0.013. Therefore, authors argue that foundation excavation activities in a construction project can be done in 0.6% of total project time. However, not all the activities in this sequence were updated with the actual start date and finish date. Therefore, this study cannot judge which sequence is the best risk-efficient practice.

Although 34 sequences were found, many of these sequences are project-based because it only appeared once and has project constraints (e.g. location). Figure 8 (b) – Tank Base Construction sequence is a typical example. There are two pairs of activities *('Mainframe Base GL1-GL8'* vs *'Mainframe Base GL8-GL16'* and *'Piling to Autoclave Mezzanine and AC Base 185 no.'* vs *'Piling to AD Base 32 no.'*) are, in essence, the 'similar' activities, but performed in different areas.

One may argue that finding a similarity threshold to find and eliminate 'similar' activities can solve the repetition problem and translate sequences into patterns. Table 1 summarises the cosine similarity of the two pairs 'similar' activities in Figure 8 (b). Cosine similarity is a metric (dot product over magnitudes) used to measure how similar the documents are irrespective of their size (Singhal, 2001). All the activities are quite similar to others since the cosine similarity ranges between 0.97 and 1.0.

The similarity between *'Mainframe Base GL1-GL8'* and *'Mainframe Base GL8-GL16'* is approximately 1.0, which indicates that these two activities are 'nearly' the same. The similarity between another pair (*'Piling to Autoclave Mezzanine and AC Base 185 no.'* vs *'Piling to AD Base 32 no.'*) is 0.99, which is close to 'nearly' the same. However, 0.99 is not a good threshold value since the similarity between *'Mainframe Base GL1-GL8'* and *'Piling to AD Base 32 no.'* is also 0.99. Therefore, translating sequences to patterns by finding and eliminating 'similar' activities is not feasible.

DISCUSSIONS

This study designed and tested a graph-based approach to classify construction activities into construction method patterns, which can then be used as the generalised best practice to initiate the schedule of new projects. The proposed method captures the text, numerical and graphical features in construction schedules. The results of this study indicated that the proposed method can encapsulate construction activities into construction method patterns.

This study found 1 construction method pattern and 34 sequences. An excavation method pattern was found 4 times; whereas other sequences were found only once. The proposed did partially addressed the 'repetition' challenge explained in the motivating case since it encapsulate excavation activities that have repeated 4 times into one process pattern. However, the 'repetition' challenge is not fully addressed. The proposed graph-based method is unable to distinguish two 'similar' activities that are connected (Figure 8 (b)). Most of these

connected ‘similar’ activities have the same actions but at different locations. Translating sequences to patterns by finding and eliminating ‘similar’ activities is also not feasible.

Future studies can address this problem via two solutions. One is training a grammatical model (e.g. Part-of-Speech) to find and remove the project constraints (such as location) in each activity before using the proposed method. Another solution is data-inefficient. Researchers can eliminate sequences that have only show once since project-based sequences constrained with project characters. Thus, it is unlikely to find the same sequence again.

Results of this study can be applied in multiple disciplines (e.g. prefabrication). Researchers (i.e. Cao and Hall (2020)) investigated the configuration of product information for modular construction by enriching process information. The proposed graph-based method can help to mine the construction schedules to configure the construction or the prefabrication process for each modular component.

Results of this study are more applicable in industry practice, particularly at the early construction stages where project information was not sufficient and accurate. Schedulers and project managers can load generalised construction method patterns and then ‘projects’ the schedules to fit project needs. By doing so, schedulers can mitigate subjective risks (e.g. personal experience) and the impact of the lack of information to produce quality front planning and avoid project delays. Meanwhile, new schedules created based on method patterns are more standardised and readable for either human or machine. Such reading problems explained in the motivating case will not occur again.

CONCLUSIONS

This study investigated how to find the most time-efficient construction method patterns. The proposed graph-based approach was tested to encapsulate construction activities into construction method patterns. The best practice to schedule a subtask was then found by comparing the time-efficiency of activity sequences in the same method patterns.

We found an excavation method pattern and observed that the most time-efficient foundation excavation activities can be done in 0.6% of project time. The results of this study support researchers and industry professionals to generalise construction methods, and thus initiate and automate the construction schedule from an early stage of a project to minimise subjective risks.

However, this result is not supported statistically given the limited sequences found. Another limitation of this study is that the project constraints in the construction activities still exist, which should be categorised to identify influencing factors. Future studies can investigate how to translate construction sequences to method patterns.

ACKNOWLEDGMENTS

We thank Kier for sharing experiences and knowledge in scheduling practice and using scheduling software and nPlan for sharing the data parsing prototype and valuable discussion about machine learning techniques. The presented work was based on research funded by InnovateUK (Project reference: 104795).

REFERENCES

Association for Project Management. (2017), “Introduction to Schedule management”, *Association for Project Management*, available at: https://www.apm.org.uk/body-of-knowledge/delivery/schedule-management/.

Benevolenskiy, A., Roos, K., Katranuschkov, P. and Scherer, R.J. (2012), “Construction processes configuration using process patterns”, *Advanced Engineering Informatics*, available at:https://doi.org/10.1016/j.aei.2012.04.003.

Benjamin, C.O., Babcock, D.L., Yunus, N.B. and Kincaid, J. (1990), “Knowledge-based prototype for improving scheduling productivity”, *Journal of Computing in Civil Engineering*, Vol. 4 No. 2, pp. 124–134.

Bhagat, S., Cormode, G. and Muthukrishnan, S. (2011), “Node Classification in Social Networks”, *Social Network Data Analytics*, available at:https://doi.org/10.1007/978-1-4419-8462-3_5.

Blanco, J.L., Janauskas, M. and Ribeirinho, M.J. (2016), “Beating the low-productivity trap: How to transform construction operations”, *McKinsey Quarterly*.

Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T. (2017), “Enriching word vectors with subword information”, *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 135–146.

Brockmann, C. (2012), “Construction Project Scheduling and Control (CourseSmart), 2nd edn”, *Construction Management and Economics*, John Wiley & Sons, Vol. 30 No. 11, pp. 1012–1013.

Cao, J. and Hall, D. (2020), “Ontology-based Product Configuration for Modular Buildings”, *Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC)*, available at:https://doi.org/10.22260/isarc2020/0026.

Changali, S., Mohammad, A. and Van Nieuwland, M. (2015), “The construction productivity imperative”, *McKinsey Quarterly*, McKinsey Quarterly, No. June, pp. 1–10.

Diestel, R. (2017), *Graph Theory. 5th Edition.*, 5th ed., Springer-Verlag Berlin Heidelberg, Berlin, available at:https://doi.org/10.1007/978-3-662-53622-3.

Firat, C.E., Arditi, D., Hämäläinen, J.P. and Kiiras, J. (2009), “Extended model-based master scheduling for building projects using advanced line of

balance", *Managing IT in Construction/Managing Construction for Tomorrow*, p. 85.
Fischer, M., Aalami, F. and O'Brien Evans, M. (1994), "Model-based constructibility analysis: the MOCA system", *CIB W78 Workshop on Computer Integrated Construction. CIB, Helsinki*.
Fırat, C.E., Kiiras, J., Kähkönen, K. and Huovinen, P. (2007), "Model Based Scheduling in Building Projects–Is It Oxymoron?", *24th CIB W78 Conference*, No. April.
Goodfellow, I., Bengio, Y., Courville, A. and Bengio, Y. (2016), *Deep Learning*, Vol. 1, MIT press Cambridge.
Habash, N., Rambow, O. and Roth, R. (2009), "MADA+ TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization", *Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR), Cairo, Egypt*, Vol. 41, p. 62.
Hamilton, W.L. (2020), "Graph Representation Learning", *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Vol. 14 No. 3, pp. 1–159.
Hamilton, W.L., Ying, R. and Leskovec, J. (2017), "Inductive representation learning on large graphs", *Advances in Neural Information Processing Systems*.
Hegazy, T. and Kamarah, E. (2008), "Efficient repetitive scheduling for high-rise construction", *Journal of Construction Engineering and Management*, Vol. 134 No. 4, pp. 253–264.
Howley, T., Madden, M.G., O'Connell, M.-L. and Ryder, A.G. (2006), "The effect of principal component analysis on machine learning accuracy with high-dimensional spectral data", *Knowledge-Based Systems*, Vol. 19 No. 5, pp. 363–370.
Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2017), "Bag of tricks for efficient text classification", *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, available at:https://doi.org/10.18653/v1/e17-2068.
Kipf, T.N. and Welling, M. (2017), "Semi-supervised classification with graph convolutional networks", *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*.
Kodinariya, T.M. and Makwana, P.R. (2013), "Review on determining number of Cluster in K-Means Clustering", *International Journal of Advance Research in Computer Science and Management Studies*.
Kolodner, J.L. (1992), "An introduction to case-based reasoning", *Artificial Intelligence Review*, Vol. 6 No. 1, pp. 3–34.
Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. and McClosky, D. (2015), "The Stanford CoreNLP Natural Language Processing Toolkit", *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60.
Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013), "Efficient Estimation of Word Representations in Vector Space", *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, available at: http://arxiv.org/abs/1301.3781.
Muñoz-Avila, H., Aha, D.W., Nau, D.S., Weber, R., Breslow, L. and Yaman, F. (2001), *SiN: Integrating Case-Based Reasoning with Task Decomposition*, MARYLAND UNIV COLLEGE PARK DEPT OF COMPUTER SCIENCE.
Muñoz-Avila, H., Gupta, K., Aha, D.W. and Nau, D. (2002), "Knowledge-Based Project Planning", in Dieng-Kuntz, R. and Matta, N. (Eds.), *Knowledge Management and Organizational Memories*, Springer US, Boston, MA, pp. 125–134.
nPlan. (2019), "WO2019234447 - Methods of Constructing a Construction, and Related Systems and Computer Program Products".
Pan, N.-F. (2008), "Fuzzy AHP approach for selecting the suitable bridge construction method", *Automation in Construction*, Vol. 17 No. 8, pp. 958–965.
Rehurek, R. and Sojka, P. (2010), "Software framework for topic modelling with large corpora", *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Citeseer, Valletta, Malta, pp. 45–50.
Sedgewick, R. and Wayne, K. (2011), *Algorithms 4th Edition*, *Journal of Chemical Information and Modeling*.
Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K. and Borgwardt, K.M. (2011), "Weisfeiler-Lehman graph kernels", *Journal of Machine Learning Research*.
Singhal, A. (2001), "Modern Information Retrieval: A Brief Overview", *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, Vol. 24 No. 4, pp. 35–43.
De Snoo, C., Van Wezel, W. and Jorna, R.J. (2011), "An empirical investigation of scheduling performance criteria", *Journal of Operations Management*, Vol. 29 No. 3, pp. 181–193.
The Royal Instituion of Charted Surveyors. (1988), *SMM7 - A Code of Procedure for Measurement of Building Works*, *Civil Engineering Standard Method of Measurement*.
Wu, I.-C., Borrmann, A., Beißert, U., König, M. and Rank, E. (2010), "Bridge construction schedule generation with pattern-based construction methods and constraint-based simulation", *Advanced Engineering Informatics*, Vol. 24 No. 4, pp. 379–388.