Linked-data based dynamic constraint solving framework to support look-ahead-planning in construction

Ranjith K. Soman,
Centre for Systems Engineering and Innovation, Imperial College London, United Kingdom
The Alan Turing Institute, United Kingdom
email: ranjithks17@imperial.ac.uk

Abstract

This paper presents a framework for solving dynamic constraints to aid the automation of look ahead planning. The constraints in construction processes are highly dynamic and complex with numerous interdependencies, making it difficult to be modelled using relational databases or similar informational models. Recent research has used linked-data based modelling approaches to address this issue. Linked-data enables linking of data across domains and heterogeneous sources, without being limited by the underlying schema. This addresses the problem of complexities and interdependencies in construction constraints. Although there is extensive research done on linked-data applications to address the limitations of current information modelling approaches in the constructions sector, the focus has been on the modelling of product information. Applications of linked-data to address the limitations in information models for representing process information such as constraints relationships and resource dependencies are relatively less explored. Building on the research on linked-data in the construction sector, this paper presents a framework for dynamic constraint solving using linked-data and Shapes Query Language (SHACL) to address this limitation. The process information is modelled in Resource Description Framework (RDF) on top of the ifcOWL ontology, and constraints are modelled in SHACL. A prototype is developed based on the framework presented, which uses a python server to process the constraints modelled in SHACL. The prototype validates the process information in the model, against the modelled constraints to check whether a schedule is feasible or not. This framework can be used to automate look-ahead-planning in construction.

Keywords: Linked-data, SHACL, constraints, dynamic constraint solving

1. Introduction

The construction sector is undergoing rapid digitization caused by the introduction of Building Information Modelling (BIM) , advancements in Information and Communication Technologies (ICT) (Alsafouri & Ayer, 2018), policy level interventions such BIM mandate in the UK (Waterhouse et al., 2016), etc. The digitization in the sector has transformed how the computers are used in the sector form being disconnected tools to being an integral part of the workflows and augmenting the daily activities. This has resulted in the generation of massive amounts of data from the projects (Anumba, Bouchlaghem, Whyte, & Duke, 2000; Whyte, Stasis, & Lindkvist, 2016). The huge volume of information generated offers the possibilities for implementing advanced decision support tools utilizing the state of the art in machine learning. Building on these developments in computing, the construction sector is moving towards agile and real-time project management practices to improve efficiency and performance (Levitt, 2011; Whyte & Levitt, 2011).

Despite having generated huge volumes of information, there are inherent problems within the generated data which limits the application of data science to support decision making, especially due to the quality of data. Bilal et al. (2016) had surveyed the Big Data techniques in the context of construction and has raised the issue of low data quality in construction data. Also, previous research in the area has reported the lack of process level information codification. Although, 4D BIM has led

to the codification of task sequences and their relationships to BIM objects (Huhnt, Richter, Wallner, Habashi, & Krämer, 2010; Subramanian, Songer, & Diekmann, 2000), Han & Golparvar-Fard (2017) has reported that the data in these 4D models lack the detail in work break down structure to match the operational details. Also, the tasks lack detailed semantics (such as the relationship between tasks, resources and constraints) leading to the nonexistence of linear dependence between resources employed and work progress (Giretti, Carbonari, Novembri, & Robuffo, 2012). Lack of detail and semantics in process information hinders the opportunities offered by the fourth industrial revolution to make the construction process management agile.

To address the issues of lack of detail and semantics, there exists a need to codify the relationships between tasks, resources and constraints in detail. However, the constraint relationship has inherent dynamicity and uncertainty limiting the users from coding them using the conventional BIM tools. BIM tools lack the ability to code these relationships in a dynamic nature as the relationships are complex and spanning across different domains (anecdotal evidence from observing BIM use in projects). This paper addresses this issue (lack of detail and semantics in the process information) by presenting a framework using Shapes Constraint Language (SHACL) and linked data principles to code complex constraints and performs dynamic constraint solving to support look ahead planning. Rest of the paper is divided into six sections. Section 2 provides background on the Linked Data and SHACL and section 3 explains the research method in brief. Section 4 presents the constraint validation framework and Section 5 presents a prototype implementation of the framework. Section 6 discusses the results from prototype testing and Section 7 presents the conclusions from this research.

2. Background

2.1 Linked data

Linked data is a method of publishing and interlinking structured information on the web (Bizer, Heath, & Berners-Lee, 2009) following the principles stated by Berners-Lee (2006). In Linked data, information artefacts are modelled as 'concepts' and related to associated 'properties' in the form of a statement. Each statement (called as triples) has three resources which are subject, predicate and object. Multiple triples are combined in the form of a graph where subject and object resources form nodes while predicates form edges of the graph (Pieter Pauwels, McGlinn, Törmä, & Beetz, 2018).The main principles of publishing information using linked data are 1) Use Uniform Resource Identifiers (URI) to name the resources (both concepts and properties) 2) Use HTTP URI, so that the modelled resources can be accessed using internet 3)Provide more information about the resources using standards (RDF,SPARQL) when the URIs are looked up and 4) Providing links to the related URIs so that the resource can act as a starting point to explore the related data spaces (Berners-Lee, 2006). These principles enable Linked-data technologies to link and share data across heterogeneous sources without being limited by the scope of underlying schemas of the source data (Zhang & Beetz, 2016).

P. Pauwels, Zhang, & Lee (2017) have surveyed the application of Linked Data in the AEC domain. Researchers have attempted to use linked data to address the limitations of BIM related to structuring and querying product information (Beetz, Van Leeuwen, & De Vries, 2009; Liu, Lu, & Al-Hussein, 2016; Zhang & Beetz, 2016), facility management (Kim et al., 2018; Terkaj, Schneider, & Pauwels, 2017), solving interoperability (Pieter Pauwels, De Meyer, Van Campenhout, Meyer, & Campenhout, 2010) etc. Although limitations of the use of BIM tools during the construction stage leading to semantic and detailed process informaiton can be achieved using linked data (by linking data across domains and using logical inferencing), it is relatively less explored.

2.2 Shapes Query Language (SHACL)

Shapes Constraint Language (SHACL) is a data modelling language to model constraints against which a linked data modelled as Resource Description Framework (RDF) could be validated (Knublauch, Allemang, & Steyskal, 2017). Validating an RDF model with a set of constraints using SHACL addresses the inherent issue of open world assumption in RDF graph as SHACL offers the

opportunity to define structural constraints (Ekaputra & Xiashuo Lin, 2016). SHACL has two components, a data graph and a shapes graph. Data graph contains the data to be validated and shapes graph contain constraints against which resources in the data graph are validated. Data graphs and shapes graph may be coded in a single RDF file or multiple RDF files as it is the triples in data graph that is validated against triples in shapes graph. There are two types of shapes in SHACL.; Node shape and Property shape. Node shapes are constraints which acts on subject resources (or concept instances) of a specific type (type can be defined while modelling constraints) in the data graph and the property shapes are constraints which acts on the predicate resources (or attribute instances). SHACL SPARQL is an advanced feature of SHACL which contains all the features mentioned above, and offers the expressive power of SPARQL to target specific instances of a subject resource satisfying certain conditions (flexible to define the conditions) (Knublauch et al., 2017). When a data graph is validated against a shapes graph, all the data instances in the data graph is checked against the applicable shapes in the shapes graph and a validation report (in the form or another RDF graph) is produced, which details the instances which have passed and violated the constraints.

The opportunities offered by linked data to interlink heterogeneous data sets without being limited by the scope of the underlying schema and the SHACL's ability to model constraints spanning domains, enables the modelling of complex constraints necessary for representing detailed and semantic construction process information, addressing the problems in conventional BIM tools. The construction process information can be represented as an interlinked web of data across domains and SHACL constraints act as a model checker for the web of data. This paper presents a framework to achieve such a representation building on the research on ifcOWL and SHACL-SPARQL.

3. Research method

The proposed research followed research methodology comprising of three phases:

1. Investigating the problems with current management practices focusing on the information flow between the design team and the constructing site using the available literature in the area and construction projects in the late design and construction phases. This part of the methodology followed the interpretive case study method (Eisenhardt & Graebner, 2007) and is out of the scope of this paper (will be discussed in detail in a forthcoming paper). However, one of the findings from the study the lack of codification of process information led to the development of the dynamic constraint solving framework.
2. Developing a framework and a prototype for codifying the process information to support look ahead planning drawing insights from the challenges observed in the previous phase of the research.
3. Evaluate the developed framework using a public dataset. The public dataset is enriched with supporting data and test scenarios are generated. Test scenarios are checked using the developed prototype to determine whether the developed prototype can detect constraint violations.

4. Constraint validation framework

The results of examining five construction projects through an interpretive case study method showed the lack of codification of detailed process information and constraints (Forthcoming paper). This finding was also evident in the literature and suggested the development of a framework to codify detailed construction process information and constraints and enable dynamic constraint solving for aiding decision-making during look ahead planning. The framework for solving the dynamic constraints is shown in Figure 1. This framework is intended to be used at the look ahead planning meetings to aid the decision making in the meetings. There are three layers in this framework; 1) User Interface Layer 2) Processing Layer and 3) Ontology layer.

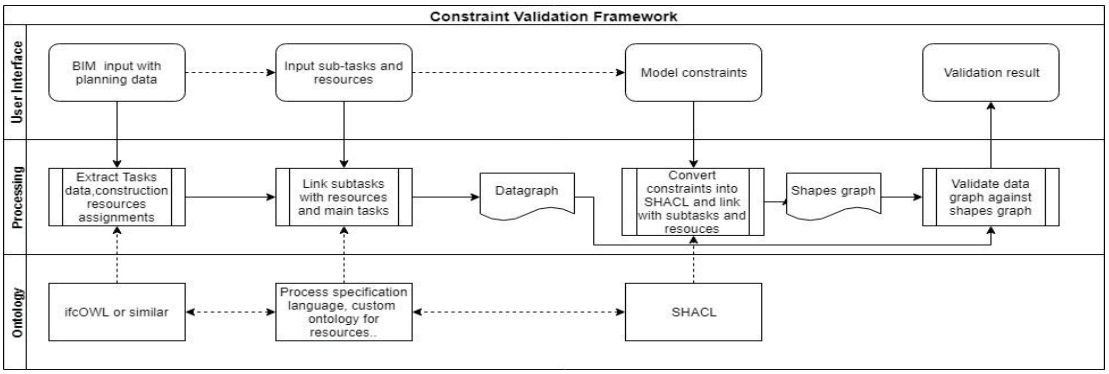

Figure 1: Framework for dynamic constraint solving

1. User interface layer: This layer is for interaction with the user. Hartmann (2008) has stated that the technologies and tools should suite with the existing work practices for successful adaptation. There are three major inputs from the user in this layer. Hence, it is recommended to design the user interface layer as an increment to the current 4D planning user interface. The first input is a BIM model with embedded planning information. This information has the planning data at the master planning/phase planning level. The second input from the user would be the information required to enrich the planning data. The information input would include subtasks for main tasks/activities in the master planning, renewable and nonrenewable resources needed for executing the tasks etc. The third input from the user includes information on constraints, and the relationship between activities, subtasks, resources, approvals etc. The output from this layer is the feasibility report of activities scheduled using the information from the input layer.
2. Processing layer: The second layer in this framework is the processing layer. There are three main processing tasks in this layer. The first task involves extracting the planning data from the BIM model and enriching it with the detailed data input from the user. For extracting the planning data, BIM ontology (eg. ifcOWL or similar) is used. To enrich the data, interlinking is done with the support of the base ontology for the resources and processes (such as process specification language.). This leads to the generation of 'data graph' for the particular look ahead window in RDF. The second task involves creating a Shape graph from the constraint information taken as an input from the user. This task would also import the previous shape graphs generated from the prior meeting. This uses the SHACL ontology. The third task is to validate the data graph against shapes graph to determine the constraint violations is any. This task uses the methodology explained in Soman (2019) to formulate the scheduling constraints and validate them.
3. Ontology layer: This layer holds the schema for different data such as BIM data, resource data SHACL etc. The schemas are stored on the web and published as per the principles stated by Berners-Lee (2006). As this layer is accessed during the execution phase, the latest updates to the ontologies and constraint relationships introduce dynamicity in this framework. Schemas are referenced and not hardcoded. Hence each schema can evolve on its own without the need for related schemas to be altered.

The developed framework has the capability to solve dynamic constraints as the data graph and the constraint graphs exists as separate graphs. The constraints are not hard-coded into the system but can be modelled during the look-ahead planning meeting and edited or appended after that. The modelled constraints apply to the data in the data graph. This enables the capture of knowledge generated during meetings to be codified into a machine-readable format and reused. For example, if constraints related to a resource (say crane) is discussed in a look ahead planning meeting and codified as SHACL constraint. These constraints would automatically be reused in the further meetings automatically even when it is not explicitly mentioned. If not relevant, there are provisions to ignore the constraints.

5. Prototype Implementation and evaluation

A prototype was developed based on the framework discussed in the previous section. The prototype demonstrates the three layers as mentioned in the framework. The prototype was developed as a web application and can be deployed on the cloud. The architecture of the prototype is as shown in Figure 2

User interface (UI) of the prototype was developed in such a way that it could easily integrated into look ahead planning meeting. Hence, a UI similar to that of a 4D planning tool was adopted (refer to Figure 3). UI is created using HTML and CSS as front end, and Javascript as a back end. UI consists of a Gantt chart showing activities for the week and interaction enabled the model view. These activities are derived from the planning data (at master planning level). When you click on an activity, object to which the activity is connected to is highlighted and vice versa. There are options to add subtasks, resources and constraints to the activity. Once these details are added, the validate button checks the data graph created from the user input with the shape graph generated from the modelled constraints to find constraint violation.

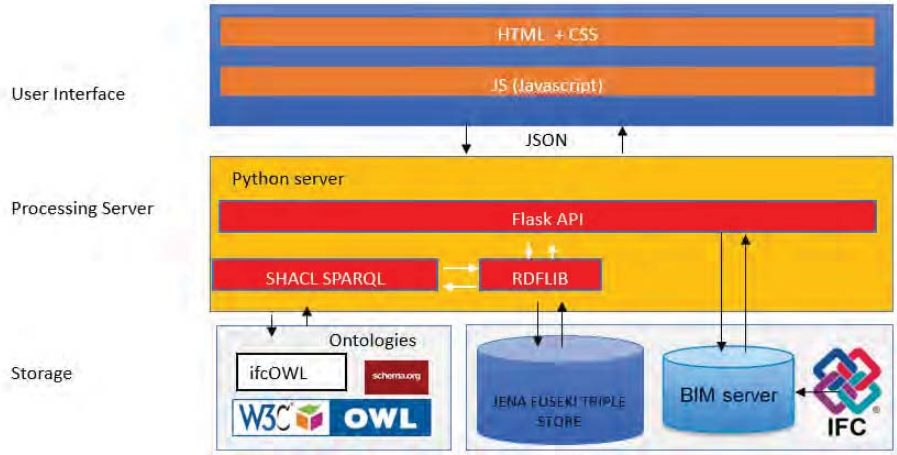

Figure 2: Prototype architecture

The processing server is coded using FLASK API in python. RDFlib library in python is used to process the data modelled in RDF. The constraints modelled are validated using SHACL-SPARQL functions in pySHACL library accessed through python server. The validation results were parsed using RDFLib and the violations if any were reported within the python server. The data is stored in a Jena Fuseki triple store. SPARQL queries to the Jena Fuseki were served over HTTP requests. IFC data in IFC OWL 2x3 is served as the model instance for the process data. To interact with the modelview, IFC open-shell and BIMs erver was used.

The ontology layer was handled through RDFLib. RDFLib accessed the schema definitions over the internet. This was queried to determine the relationship between schema. The main ontologies used for the prototype are OWL ontology, RDFs ontology, RDF ontology, ifcOWL, SHACL and a custom process ontology based on Process Specification Language. As planned model from the dataset Schependomlaan was imported for the testing. The data set had activities in it coded as IFC Tasks class with the schedule information coded into the IFC Schedule Time Control class. Ifc Rel Assigns To Process class connected the particular object instance to the IFC Task class. IFC Rel Sequence defines the precedence relationship of the IFC Tasks For the demonstration of the prototype, Construction resources and subtasks are defined as custom classes Subtask and Resource. Subtasks are assigned to IFC Tasks using sub task of and Resources are assigned to IFC Tasks using resource OF as shown in Figure 4 for this paper. Subtasks are inherited from IFC Tasks The enriched data along with the planning

data from the BIM forms the data graph.

The data as given in Table 1 were given as an input to the prototype. The instances for IfcTask class which were enriched can be referenced as Tasks T1 and T2 in this paper. Subtasks for T1 and T2 are S1,S2,S3 and S4,S5,S6 respectively. R1 ,R2,R3,R4 are the resources available to the activities. are

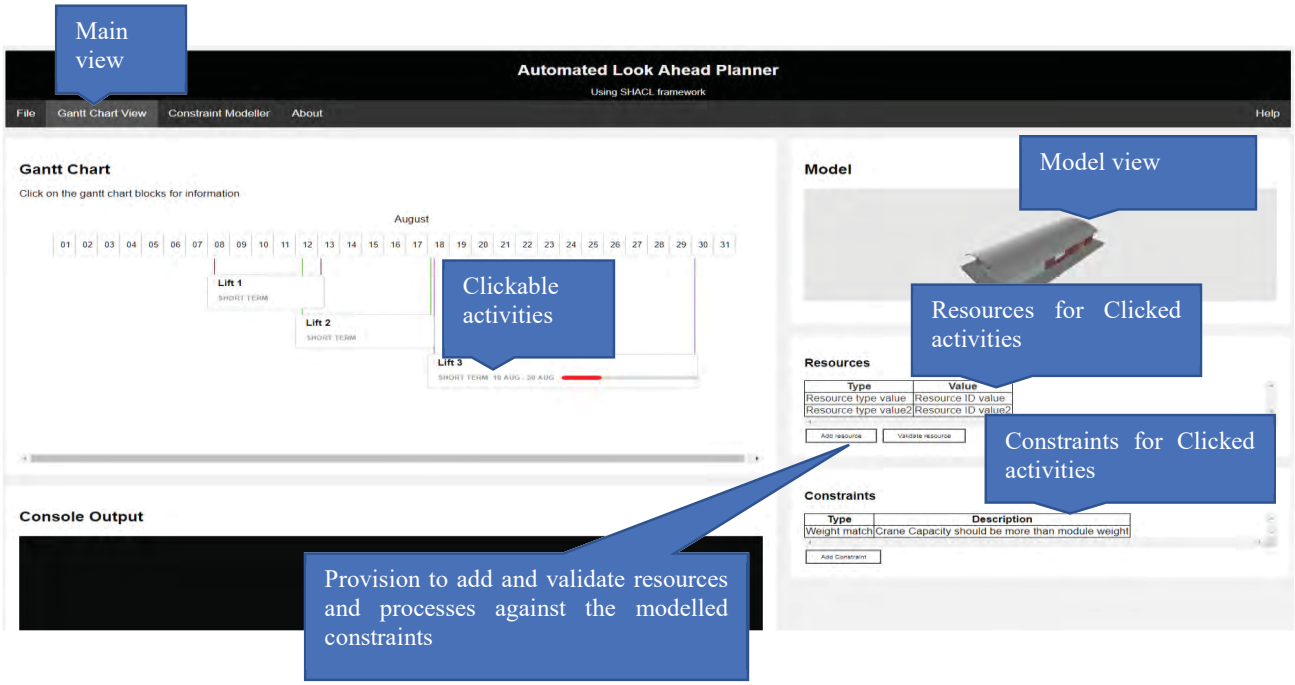

Figure 3: Screenshot of the prototype user interface

the O1 and O2 objects for which tasks and subtasks to be carried out. The detailed inputs were given in two sessions. First session for S1,S2,S3 and second session for S4,S5,S6.

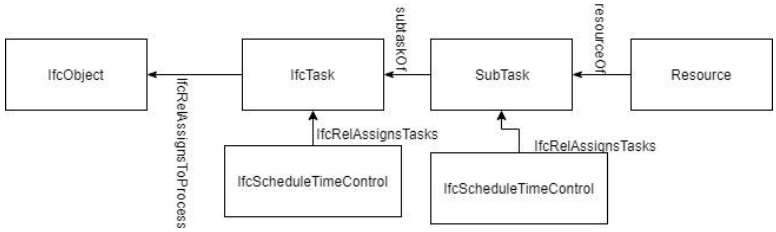

Figure 4: Ontology for linking process information

Input from the users on constraints was used to generate SHACL queries. Precedence and disjunctive constraints as specified in Morkos (2014) were given for the activities. S1,S2,S3 had a start to finish relationship as precedence constraint (meaning S1 happens first followed by S2 and finally S3). Similarly S4,S5,S5 had start to finish relationship. Also the resources R1,R2,R3 and R4 were assigned the disjunctive property. This means that they can not be shared between activities. The collection of the SHACL queries formed the shapes graph. During the validation Data graph generated were checked against constraints modelled in the Shapes graph to find the constraint violation.

Table 1: Input data for generating the data graph

SubTask	Parent Task	Start date	End date	Resource assignment	Object assignment
S1	T1	21/09/2015	23/09/2015	R1,R2	O1
S2	T1	24/09/205	27/09/2015	R2,R3	O1
S3	T1	28/09/2015	30/09/2015	R1,R2	O1
S4	T2	22/09/2015	25/09/2015	R4	O2
S5	T2	26/09/2015	30/09/2015	R1,R4	O2
S6	T2	29/09/2015	03/10/2015	R2,R3	O2

6. Results and Discussion

The results after validating the inputs into the prototype are shown in Table 2. There were no violations for subtasks S1, S2 and S4. However, there were disjunctive constraint violation for S3.Also, disjunctive and precedence constraint violations for S5 and S6. The resource R1 assigned to S3 was already used by S5 which resulted in a disjunctive constraint as the resource cannot be shared between two activities at the same time. Similarly, Resource R2 assigned to S6 were used by S3 at the same time. Also, their precedence constraint applied to S5 and S6 stated S6 to start after S5 which was violated as S6 started before S5 ended. Therefore, the constraint violation framework could identify all the violations.

Table 2: Results after constraint validation

Subtask	Result
S1	No violation
S2	No Violation
S3	Disjunctive constraint violation
S4	No violation
S5	Disjunctive constraint violation, precedence violation
S6	Disjunctive constraint violation, precedence violation

Although this is an expected result and it could be solved without using any framework, the complexity of the problem increases with increase in the number of simultaneous activities, making it difficult to solve it without the help of a tool. In addition, the constraints are dynamic. Hence while adding an extra constraint, there is no need to hardcode the constraints again or alter the data graph. Only the shapes graph will be modified. The validation applies to current (which might have not violated any constraints previously) and future data. Isolating the data from the constraints makes it possible for keeping the constraints dynamic as the shapes graph can be modified at any stage without modifying the data graph and vice versa. Whenever, variables in a data graph changes, for example a process gets delayed or a resource is broken down, the remaining processes in data graph is evaluated against the modelled constraints to see what activities are affected and how they are affected. Keeping data graph and constraints separate also supports the implementation machine learning algorithms such as reinforcement learning where the agent and environment are different. Similarly, here constraint solver could act as an environment and provide rewards for actions taken by agent based on the no of constraint violations. These tools help to reschedule the processes and aid decision making during disruptions as the current state of work and the constraints are codified. It also acts as a 'unit testing' for the short term schedule data and supports collaboration, as it checks the data every time an edit is made with knowledge codified as constraints.

This problem also demonstrates the ability of linked data to traverse across the web of data. The disjunctive constraint was applied to the resources which don't have any schedule data associated with it. The schedule data was coded into the subtask as shown in Figure 4. However, the SHACL query could crawl through the data to identify the relevant time data from the resource assignments and making logical inferences on whether there were multiple assignments during the same time period. This shows the ability of linked data approaches to make semantic queries by traversing through the

data. As the ontology layer is separate from the processing layer, queries need not have the know-how of the whole schema but only the attributes necessary for making the traversal.

The designed prototype can be used to look ahead meetings just like any 4D planning tool. The details about the relationship between the subtasks and resources, resources and constraints, constraints and subtasks, and subtasks and subtasks can easily be codified into a machine-readable format using the framework. The codified knowledge on the relationships and constraints can be reused further. This reduces the need for data input after a stage of saturation as the shapes graph gets populated during every session. The codified constraints in shape graph could be used for further session. For example, once a disjunctive constraint is assigned to a resource, whenever that resource is used, the constraint is automatically applied to it. Therefore, this framework helps to codify the constraint and relationships between resources, subtasks and tasks and this codified data can support decision making in the future. Indirectly, the knowledge concerning resources and task gets captured and codified within the system.

7. Summary and Conclusions

This paper presented a dynamic constraint solving framework to support look-ahead-planning in construction based on linked data. A prototype was developed based on the framework to test the efficacy of the framework. The prototype was developed based on 4D planning tool so that it gets integrated into the workflow. The developed prototype was able to identify the constraint violations within the detailed schedule data. This demonstrates the potential of the frameworks to be used to look ahead planning meetings to assist in decision-making process.

The current method using linked data enables the use of semantic queries to understand the constraint violations by traversing across datasets without the complete knowledge of the dataset's underlying schema. The framework enables domain knowledge generated and used during the look-ahead planning to be captured and codified so that it can be reused down the lane. This enables the experts to feed into the information during a meeting and then the system reuses the knowledge hence forth, without having to bring the expert again during disruptions. In addition, isolating the data modelling and constraint modelling to two different parts helps to enable dynamic constraint processing which is a necessary requirement for look ahead planning as the data inputs and constraints might change independently based on the dynamic nature of a construction site, however the data has to be validated and checked against the modelled constraints. The constraints modelled in SHACL acts similar to unit testing (in software testing) for the scheduling. Every time a change is made in the process information in the data graph, it is checked against the modelled constraints. In addition, separating constraint solving into a different graph supports the introduction of machine learning techniques such as reinforcement learning to automate look ahead planning, paving way for agile construction management practices as mentioned in Levitt (2011) to make decisions on the go based on inputs from various sources.

This paper has presented and demonstrated the framework for dynamic constraint solving using linked data. However, the testing has been done using a public construction dataset as a proof of concept. There is a need to test this framework in actual look ahead planning meetings to demonstrate the effectiveness and understanding the limitations. Also, future research should focus on an intuitive visual interface to generate the complex constraints so that it can be easily used by practitioners. In addition, there is an opportunity for learning from the codified constraint information, given the tasks and resources to predict the construction methods and generate schedules, which needs to be explored.

Acknowledgements

The PhD research of the author is co-funded by Bentley systems UK and, through a Skempton Scholarship, the Department of Civil and Environmental Engineering, Imperial College London. During the development of this paper, author was supported by the PhD enrichment scholarship from The Alan Turing Institute. The author acknowledges the supervisory inputs offered by Prof Jennifer Whyte and Dr David Birch and Dean Bowman. Also, author acknowledges the guidance offered by Prof. Jakob Beetz.

References

Alsafouri, S., & Ayer, S. K. (2018). Review of ICT Implementations for Facilitating Information Flow between Virtual Models and Construction Project Sites. Automation in Construction, 86, 176–189. https://doi.org/10.1016/J.AUTCON.2017.10.005

Anumba, C. J., Bouchlaghem, N. M., Whyte, J., & Duke, A. (2000). Perspectives on an integrated construction project modeL. International Journal of Cooperative Information Systems, 09(03), 283–313. https://doi.org/10.1142/S0218843000000107

Beetz, J., Van Leeuwen, J., & De Vries, B. (2009). IfcOWL: A case of transforming EXPRESS schemas into ontologies. Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM, 23(1), 89–101. https://doi.org/10.1017/S0890060409000122

Berners-Lee, T. (2006a). Linked Data - Design Issues. Retrieved January 14, 2019, from https://www.w3.org/DesignIssues/LinkedData.html

Berners-Lee, T. (2006b). Linked Data - Design Issues. Retrieved from https://www.w3.org/DesignIssues/LinkedData.html

Bilal, M., Oyedele, L. O., Qadir, J., Munir, K., Ajayi, S. O., Akinade, O. O., … Pasha, M. (2016). Big Data in the construction industry: A review of present status, opportunities, and future trends. Advanced Engineering Informatics, 30(3), 500–521. https://doi.org/10.1016/j.aei.2016.07.001

Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems, 5(3), 1–22. https://doi.org/10.4018/jswis.2009081901

Eisenhardt, K. M., & Graebner, M. E. (2007). Theory Building from Cases: Opportunities and Challenges. Academy of Management Journal, 50(1), 25–32. https://doi.org/10.5465/AMJ.2007.24160888

Ekaputra, F. J., & Xiashuo Lin. (2016). SHACL4P: SHACL constraints validation within Protégé ontology editor. 2016 International Conference on Data and Software Engineering (ICoDSE), 1–6. https://doi.org/10.1109/ICODSE.2016.7936162

Giretti, A., Carbonari, A., Novembri, G., & Robuffo, F. (2012). Estimation of job-site work progress through on-site monitoring. 2012 Proceedings of the 29th International Symposium of Automation and Robotics in Construction, ISARC 2012, 11. https://doi.org/10.4017/gt.2012.11.02.348.704

Han, K. K., & Golparvar-Fard, M. (2017). Potential of big visual data and building information modeling for construction performance analytics: An exploratory study. Automation in Construction, 73, 184–198. https://doi.org/10.1016/j.autcon.2016.11.004

Hartmann, T. (2008). A grassroots model of decision support system implications by construction project teams. Stanford University, California, USA.

Huhnt, W., Richter, S., Wallner, S., Habashi, T., & Krämer, T. (2010). Data management for animation of construction processes. Advanced Engineering Informatics, 24(4), 404–416. https://doi.org/10.1016/j.aei.2010.07.009

Kim, K., Kim, H., Kim, W., Kim, C., Kim, J., & Yu, J. (2018). Integration of ifc objects and facility management work information using Semantic Web. Automation in Construction, 87, 173–187. https://doi.org/10.1016/J.AUTCON.2017.12.019

Knublauch, H., Allemang, D., & Steyskal, S. (2017). SHACL Advanced Features.

Levitt, R. E. (2011). Towards project management 2.0. Engineering Project Organization Journal, 1(3), 197–210. https://doi.org/10.1080/21573727.2011.609558

Liu, H., Lu, M., & Al-Hussein, M. (2016). Ontology-based semantic approach for construction-oriented quantity take-off from BIM models in the light-frame building industry. Advanced Engineering Informatics, 30(2), 190–207. https://doi.org/10.1016/j.aei.2016.03.001

Morkos, R. (2014). Operational efficiency frontier: Visualizing, manipulating, and navigating the construction scheduling state space with precedence, discrete, and disjunctive constraints. Stanford University.

Pauwels, P., De Meyer, R., Van Campenhout, J., Meyer, R. De, & Campenhout, J. Van. (2010). Interoperability for the Design and Construction Industry through Semantic Web Technology. In: Declerck T., Granitzer M., Grzegorzek M., Romanelli M., Rüger S., Sintek M. (eds) Semantic Multimedia. SAMT 2010. Lecture Notes in Computer Science, vol 6725. Springer, Berlin, Heidelberg.

Pauwels, P., McGlinn, K., Törmä, S., & Beetz, J. (2018). Linked Data. In Building Information Modeling (pp. 181–197). https://doi.org/10.1007/978-3-319-92862-3_10

Pauwels, P., Zhang, S., & Lee, Y.-C. (2017). Semantic web technologies in AEC industry: A literature overview. Automation in Construction, 73, 145–165. https://doi.org/10.1016/J.AUTCON.2016.10.003

Soman, R. K. (2019). Modelling construction scheduling constraints using shapes constraint language (SHACL). *2019 European Conference on Computing in Construction (EC3)*. Chania, Greece.

Subramanian, P. S., Songer, A. D., & Diekmann, J. E. (2000). Visualizing Process Performance. Computing in Civil and Building Engineering (2000), 1–6. https://doi.org/10.1061/40513(279)1

Terkaj, W., Schneider, G. F., & Pauwels, P. (2017). Reusing Domain Ontologies in Linked Building Data : the Case of Building Automation and Control. *Proceedings of the 8th Workshop Formal Ontologies Meet Industry, Joint Ontology Workshops* 2017, CEUR Workshop Proceedings, (p2050).

Waterhouse, R., Kemp, A., Matthews, A., Moulds, A., Mordue, S., Malleson, A., & Philp, D. (2016). National BIM Report. In National BIM Library.

Whyte, J., & Levitt, R. (2011). Information Management and the Management of Projects. In The Oxford Handbook of Project Management. https://doi.org/10.1093/oxfordhb/9780199563142.003.0016

Whyte, J., Stasis, A., & Lindkvist, C. (2016). Managing change in the delivery of complex projects: Configuration management, asset information and "big data." International Journal of Project Management, 34(2), 339–351. https://doi.org/10.1016/j.ijproman.2015.02.006

Zhang, C., & Beetz, J. (2016). Querying Linked Building Data Using SPARQL with Functional Extensions Querying Linked Building Data Using SPARQL with Functional Extensions. Ecppm2016, (October).