2023 Conference on Systems Engineering Research

# Verification Complexity: An Initial Look at Verification Artifacts

Raphael Jung[a], Alejandro Salado[a,*]

[a]*The University of Arizona, 1127 E James E Rogers Way, Tucson, AZ, USA*

**Abstract**

There has been increasing interest in developing numerical methods to inform the design of verification strategies in recent years. Existing work has been applied to toy problems that significantly simplify the complexity of verification in real-life applications, both in terms of the size of the problem and the interdependencies between the different elements of a verification strategy. To our knowledge, there is no publicly available evidence of the complexity of the verification "problem" in terms of requirements that need to be verified, verification activities that are conducted, and their interrelationships. In this paper, we use knowledge graphs to visualize the size of the verification problem for two industrial projects, as captured in their verification artifacts, which include requirements traceability matrices and verification matrices.


*Keywords*: verification; modelling; knowledge graphs; complexity;

## 1. Introduction

There has been an increasing interest in the systems engineering community toward fundamental research that is based on strong mathematical foundations [1-4]. Most of the recent research efforts have been dedicated toward the early stages of the system lifecycle, starting from the concept of operations to design [4-9], specifically in optimizing the decisions involved in these phases of the lifecycle (e.g. [7, 8]**)**. This paper focuses on verification, which may take place during the entire system's lifecycle.

Verification is commonly defined by the question *"Was the system built right?"* and is achieved when system requirements are satisfied [10, 11]. Traditional verification methods include inspection, analysis, or testing [12], although other ones may also be envisioned. Various verification activities are usually combined in a specific sequence

* Corresponding author. Tel.: +1-520-626-0728; fax: +0-000-000-0000 .
*E-mail address:* alejandrosalado@arizona.edu

to form a verification strategy. The design of the verification strategy is driven by the necessity to achieve certain confidence in the correct functioning of the system, while efficiently investing resources [12].

A verification strategy may be abstracted as an acyclic directed graph where the verification activities are represented by nodes and the edges represent their information influence [13, 14]. A verification activity is conceptualized as the collection of information about a specific aspect of the system under development and verification evidence refers to such information. Assuming a decision-based design framework [15, 16], verification activities are critical to system development because they shape the uncertainty associated with the functioning or performance of the system that is being developed [10, 17]. Since uncertainty may be quantified in probabilistic terms, irrespective of the type of uncertainty [18], such probabilities must be subjective if a decision-based paradigm is adopted as they must represent the beliefs of the decision maker [19]. For example, different analysts use different sources of information to arrive at their prior probabilities, leading to imprecise probabilities [20]. In essence, while data are objective, their interpretations by the engineer into the level of confidence are subjective [21].

Traditionally, verification strategies are qualitatively designed based on the experience of subject matter experts or following industry standards (e.g. [22, 23]). The process by which engineers reason about different pieces of information to arrive at their judgments on the verification of a system is central to this work. Empirical research shows that people update their beliefs when new information becomes available [24, 25]. Mathematics, however, shows that certain prior beliefs can have substantial anchoring effects irrespective of the update and vice versa [18]. This finding poses a challenge, as well as an opportunity, to verification, since it emphasizes the importance of the subjective interpretation of the engineer over the data collected by the verification activity. In this context, to what extent can an engineer reason through a vast amount of requirements and verification artifacts? Or, if they do, what assumptions and unconscious approximations do they make?

This paper is framed around these questions and provides an initial knowledge graph analysis of verification artifacts from two industrial projects. Existing literature addressing the design of verification strategies relies on toy problems that significantly simplify the complexity of verification in real-life applications, both in terms of the size of the problem and the interdependencies between the different elements of a verification strategy (e.g., [12, 26-30]). To our knowledge, this is the first research paper where the complexity of verification strategies is shown at scale. We suggest that understanding this complexity may be used in future research to improve the planning and execution of verification strategies.

This paper is organized as follows. Section 2 discusses the research design, including the scale of verification strategies, knowledge graph construction, descriptions of two data sources, and the metrics used during the analysis. Section 3 visualizes the generated knowledge graphs for each dataset and provides analysis on their graph structures. Section 4 provides discussions and concluding remarks on the study.

## 2. Research design

### *2.1. Data sources*

Two verification strategy datasets, corresponding to two different systems, were independently provided by two international private companies. In accordance with non-disclosure agreements made between the University of Arizona and both companies, any competitively sensitive information is protected through anonymization. The two datasets are therefore named *Strategy 1* and *Strategy 2*, respectively, based on the order they were received, and, as mentioned, represent the verification strategies of two different systems. Table 1 lists the number of distinct requirements and distinct verification activities in the verification matrices of the data sources, along with those of other data sources used in the existing literature. The references in the column heading represent research with a specific data source. The difference in size is several orders of magnitude.

Table 1. Distinct requirements and verification activities of the data sources and comparison against existing literature.

| *Source* | *Strategy 1* | *Strategy 2* | *[27, 31]* | *[13]* | *[32, 33]* | *[34]* | *[35]* | *[29]* | *[36]* | *[37]* | *[38]* | *[28]* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # of distinct requirements | 129 | 3,385 | n/a | 15 | 4 | 1 | 2 | 1 | 2 | 5 | 9 | 6 |
| # of distinct verification activities | 179 | 2,576 | 11 | 15 | 6 | 6 | 2 | 3 | 4 | 9 | 17 | 6 |

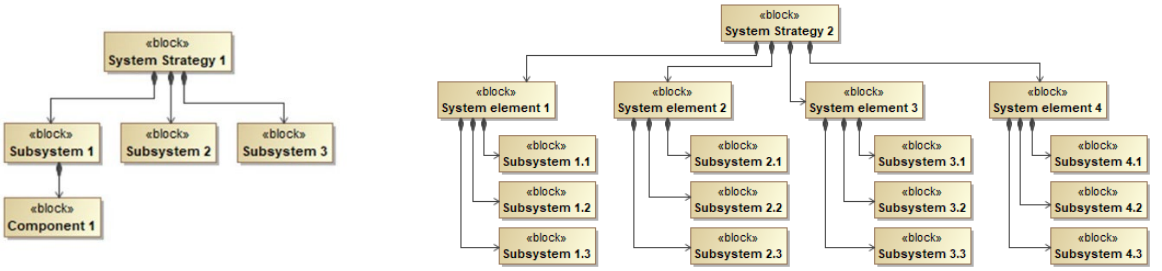


Fig. 1. (a) Product tree of the system subjected to *Strategy 1*; (b) Product tree of the system subjected to *Strategy 2*.

### *2.1.1. Strategy 1*

*Strategy 1* contained the requirements and verification matrix of a product aimed at providing non-pharmacological treatments for a specific medical condition. The artifacts were produced and delivered as a single .xlsx file. The verification matrix contained system requirements, associated verification activities, associated verification evidence in the form of verification closeout documents, and additional information about the model used in the verification activities. Traceability between the different objects was provided by having them in the same row in one sheet or manually assigning unique identifiers across different sheets.

The system was asymmetrically decomposed two levels down. Fig. 1 (a) shows the product tree that maps the different requirement sets that were produced. Requirements and verification information were provided for each of the elements in the figure and formed the complete verification information for the system. They are all included in the analysis. Requirements in normative documents were not incorporated.

### *2.1.2. Strategy 2*

*Strategy 2* contained the requirements and verification matrix of a defense system. The artifacts were produced in a DOORS database, which was exported and delivered as a single .xlsx file. The verification matrix contained system requirements and associated verification events and/or activities. Traceability between the different objects was managed in DOORS and was exported into the .xlsx file by providing objects in the same row in one sheet or assigning unique identifiers across different sheets.

The system was symmetrically decomposed two levels down. Fig. 1 (b) shows the product tree. Requirements and verification information were provided for the highest-level system and all subsystems but not for system elements. These are all included in the analysis. Furthermore, some subsystems were decomposed into some components, but requirements and verification information for these were handled outside of DOORS and have not been provided for analysis at the time of writing this article. Requirements in normative documents were not incorporated.

## *2.2. Construction of knowledge graphs*

Neo4J was employed to build the knowledge graphs, which were constructed using the elemental patterns in [13] as underlying structures. The patterns establish that (1) a system parameter, which represents an aspect of the system that addresses a requirement, can be verified by a verification activity, (2) a system parameter may represent any system model, (2) a system parameter may provide information about another system parameter (e.g., the mass of a component may inform the mass of a system or the mass a prototype may inform the mass of the final product), and (3) the result of a verification activity may inform the result of another verification activity. Note that in this paper, each requirement has been associated with a system parameter and that relation (3) has not been assessed.

The datasets under evaluation used different metamodels. These metamodels were mapped to the core elements in the patterns, as shown in Fig. 2. The nodes represent entities in the verification strategies. The parameter nodes are included in both graphs with the same three attributes *reqId* (a unique identifier representing a distinct requirement), *description* (description of the requirement), and *satisfies* (a traceability relationship to another requirement). This attribute structure is represented as (:Parameter {*reqId*, *desc*, *satisfies*}). Verification artifacts differ slightly between

datasets. The verification elements in both datasets share two attributes: *reqId* (pointing at the requirement it verifies) and *method* (type of the verification activity). The verification identifiers vary between the two. *Strategy 1* uses *procedure* (verification procedures used for the targeted system parameter) and *closeout* (verification evidence), while *Strategy 2* has a single *verId* (a unique verification identifier). (:Verification {*reqId*, *method*, *procedure*, *closeout*}) is added for *Graph 1* while (:Verification {*reqId*, *verId, method*}) is added for *Graph 2*. Only *Strategy 1* provided extensive information on the models and documents related to the verification strategy, therefore (:Model {*modelId*}) and (:Document {*docId*}) are only added to *Graph 1*. They are treated as external information nodes, and descriptive attributes such as *level* and *type* of the models and *name* of the documents were added for analytical purposes.

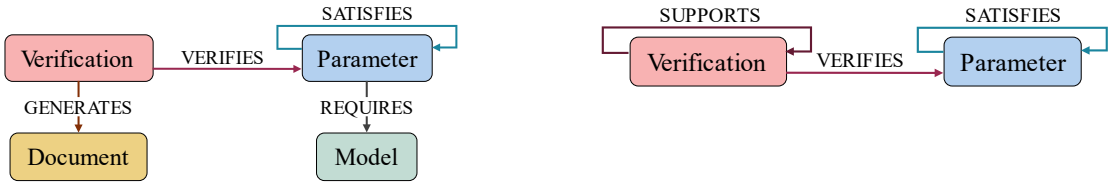


Fig. 2. (a) *Graph 1* structure; (b) *Graph 2* structure.

The edges represent relationships between nodes where the aforementioned elemental patterns can be detected. The figure shows that two graphs share two edge types; [:VERIFIES] and [:SATISFIES]. [:VERIFIES] is a many-to-many connection edge between *Parameter* and *Verification* generated by connecting the nodes with identical *reqId* attributes. [:SATISFIES] on the other hand are generated by utilizing *satisfies* attribute in the *Parameter* nodes. There are also three edge types only shown in either one of the datasets. [:SUPPORTS] relationships built from the *supported_by* attribute allow depictions of Pattern II and IV in *Graph 2* by providing hierarchical connections between *Verification* nodes. [:GENERATES] and [:REQUIRES] in *Graph 1* are used to represent connections to two external information sources *Documents* and *Models*.

### *2.3. Metrics used for analysis*

Centrality measures the rankings of nodes, such as the importance of influence, within graphs. Six graph centrality metrics were calculated to statistically summarize the characteristics of each knowledge graph. Firstly, four variations of centrality measures were calculated. These measures represent how *central* each node is within a given network, with high-valued nodes treated as hub nodes. A *degree* centrality or a node degree is one of the most simple node importance measures where nodes with heavier connections become more *central* proportional to their degree. It is calculated to show as the basic node importance measure. The *betweenness* and *closeness* centralities were calculated to respectively capture the flow and the distances between nodes. Both utilize shortest paths in their calculation; *betweenness* measures the ratio of membership in shortest paths and *closeness* measures the average length of shortest distances. The transitive nature of centrality values is more reflected in the *eigenvector* centrality by iteratively updating the eigenvalues. Two variations of *eigenvector* centrality are then computed to include the dampening of influence as it traverses over the nodes. This simulates the diminishing influence a node has over distant neighbors, which is often observed in social networks. The *PageRank* algorithm is one of the first algorithms to recognize this effect, introducing jumping (or hyperlinking) capabilities with a damping factor to reflect the diminishing importance and influence a node has over other nodes as you move further from it. The *articleRank* is an algorithmic variant of *PageRank* giving less weight to low-degree nodes, effectively mimicking the transitive influence between journal articles. While they are directly developed to reflect the centrality of verification strategy artifacts, they are a set of widely accepted centrality measures over multiple domains. The distribution of centrality values reflects the overall structure of a given graph, therefore similarities between verification strategies can numerically be measured using the measures. They also allow numerical comparison within and between verification activities and system parameters by quantifying the influences they have on other artifacts.

## 3. Knowledge graphs from verification strategies and analysis results

Using the data sources and metamodels described in the previous section, two graphs (*Graph 1* and *Graph 2*) were generated. Table 2 and Table 3 show the size of two knowledge graphs generated from respective datasets. A significantly larger size of *Graph 2* can be seen with 14.75 times more nodes and 14.33 times more edges. The distributions of nodes and edges are different, as well. *Graph 2* has more *Parameters* than *Verifications* as opposed to *Graph 1* while having more than half of its edges as SATISFIES. This indicates that *Graph 2* has much heavier connections between *Verifications*. It also has SUPPORTS edges between *Parameters* unlike *Graph 1*.

Table 2. The number of nodes for *Graph 1* and *Graph 2*.

| Number of nodes | *Graph 1* | *Graph 2* |
|---|---|---|
| *Parameter* | 129 | 3,385 |
| *Verification* | 179 | 2,576 |
| *Document* | 48 | - |
| *Model* | 48 | - |
| Total | 404 | 5,961 |

Table 3. The number of edges for *Graph 1* and *Graph 2*.

| Number of edges | *Graph 1* | *Graph 2* |
|---|---|---|
| VERIFIES | 179 | 1,945 |
| SATISFIES | 61 | 4,308 |
| SUPPORTS | - | 1,817 |
| REQUIRES | 129 | - |
| GENERATES | 194 | - |
| Total | 563 | 8,070 |

### *3.1. For Strategy 1*

*Strategy 1* is visualized as a knowledge graph in Fig. 3 (a). Each node labels are color-coded; the Parameter is shown as blue, the Verification as red, the Document as yellow, and the Model as green. Manual inspection reveals a pseudo-tree-shaped graph, having a single major connected component and several smaller isolated components as naturally-occurring clusters. Such clusters became more visible when only the verification *procedures* were concerned. Disregarding the verification *closeouts*, a variation of *Graph 1* in Fig. 3 (b) shows more pronounced cluster structures surrounded by more isolated nodes and small components. Two mainly red (*Verification*) clusters centering around respective brown (*Document*) nodes indicate that the verification plan/procedures mainly revolve around the two documents when they are not connected to the verification *closeouts*.

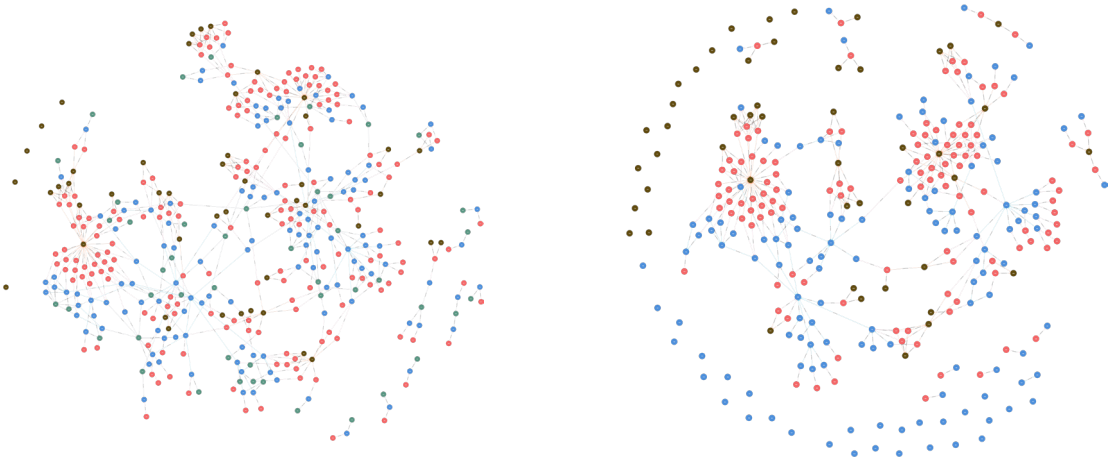

Fig. 3. (a) Visualization of the complete *Graph 1*; (b) Visualization of *Graph 1* with only *Verifications* with *procedures*.

Table 4 shows the descriptive statistics of *Graph 1* in terms of centrality. An average *degree* value of 1.39 indicates that the majority of nodes are leaf nodes having one-to-one relationships. Other centrality measures support this observation, showing that the lower half of the value distributions are much more condensed compared to the upper

half. The distance between the minimum and median (p50) values is considerably little compared to the distance between p50 and max values. There are few extremely connected nodes inflating the overall mean values shown by sudden increases from p95 and max values, which is especially visible in *betweenness* centrality. These indicate the existence of a few *hub* nodes within the networks. Interactive graph inspection through Neo4J's UI allowed the authors to identify detailed information about any individual nodes and understand the characteristics of such nodes. Examples include basic prerequisite parameters such as *safety certificates*, generic verification activities such as *site inspection* and *capability verification*, major verification *Models* such as *simulated system* and *system in the final configuration*, and a *test procedure report* Document responsible for many verification activities. Detection of such hub nodes can aid a more effective formation of verification strategies by providing avenues for balancing workloads and differentiating vaguely defined entities. On the other hand, there are a number of nodes disconnected from the main graph as well. Fully isolated nodes mainly include independent documents such as internally used *patch requesting documents* or deliverable reports such as a *final report*.

Table 4. Statistical summary of the centrality in *Graph 1*.

| measure | min | p50 | p75 | p90 | p95 | max | mean |
|---|---|---|---|---|---|---|---|
| *articleRank* | 0.1500 | 0.1876 | 0.2222 | 0.3020 | 0.3959 | 1.2572 | 0.2134 |
| *betweenness* | 0.0000 | 0.0000 | 1.0000 | 5.0000 | 7.0000 | 63.0002 | 1.8243 |
| *closeness* | 0.0000 | 0.6000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 0.4819 |
| *degree* | 0 | 1 | 2 | 2 | 3 | 6 | 1.3936 |
| *eigenvector* | 0.0000 | 0.0000 | 0.0000 | 0.0003 | 0.0008 | 0.9801 | 0.0035 |
| *pageRank* | 0.1500 | 0.2137 | 0.2775 | 0.5038 | 0.7025 | 4.0058 | 0.2915 |

There are also verification activities independent of the main task, which is verifying requirements associated with the primary use cases of the product being developed. Such activities related to requirements associated with second-order use cases. The smaller connected components (that is, subgraphs that are disconnected from each other) can be viewed as a natural cluster. Fig. 4 showcases the second and third largest connected components from *Graph 1* representing two miniature verification strategies encompassing all four entity types. Fig. 4 (a) shows a closed tree structure with a single document and a single model at either end. Interpreting the component revealed that *model_9* shown in Fig. 4 (a) is a product packaging concerning safety, and it is verified by multiple parameters which in turn results in a generation of the relevant user manual *document_223*. Fig. 4 (b) contains two documents instead (*document_224* as shipping instruction and *document_234* as temperature log). As *model_956* represents shipment packaging, it is assumed that this component is designed to verify temperature safety during packaged shipping. This is the 3rd largest connected component in *Graph 1*; with the 1st largest component accounting for 351 out of 404 nodes, one can assume that *Strategy 1* heavily revolves around a central verification activity.

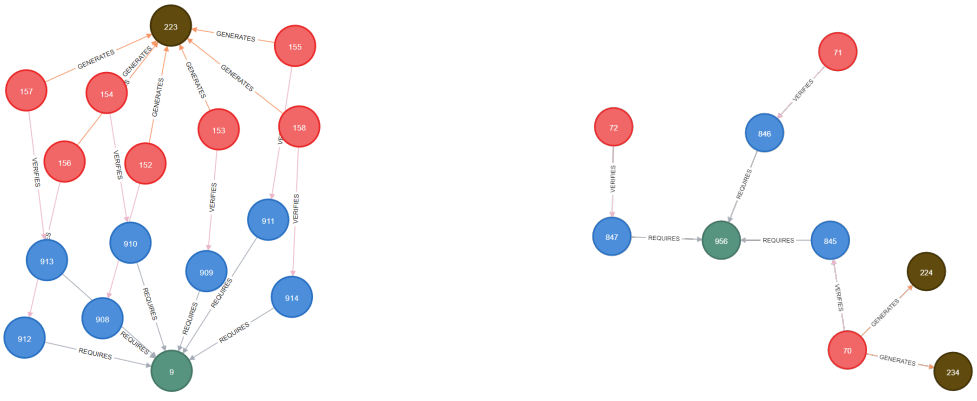


Fig. 4. (a) 2nd and (b) 3rd weakly connected components in *Graph 1*.

### *3.2. For Strategy 2*

Fig. 5 shows the visualization of Graph 2, using the same color code that was used in Fig. 3; the Parameter as blue and the Verification as red. Visual inspection indicates that while Graph 1 and Graph 2 have a different number of node labels and have a large disparity in their sizes, their overall shapes are similar to each other. Graph 2 is also pseudo-tree-shaped, showing naturally occurring clusters in a form of smaller detached components and heavily connected branches. The clusters in Fig. 5 show large condensations of Parameter nodes on the right side, showing predominantly blue clusters. They are the result of the SATISFIES relationship specific to *Graph 2*; high-level parameters behave as hub nodes by being *satisfied_by* numerous, more specific parameters. The largest *Parameter*-focused cluster can be attributed to three such hub nodes. Parameter_559 represents the satisfactory condition for the service provision support, parameter_988 represents the compliance with the technical requirements of relevant national standards, and parameter_1276 investigates cyber-security procedures used during the verification process. As such, the authors can assume that the hub nodes in graphs are indeed high-level entities in the verification strategies governing lower-level entities.

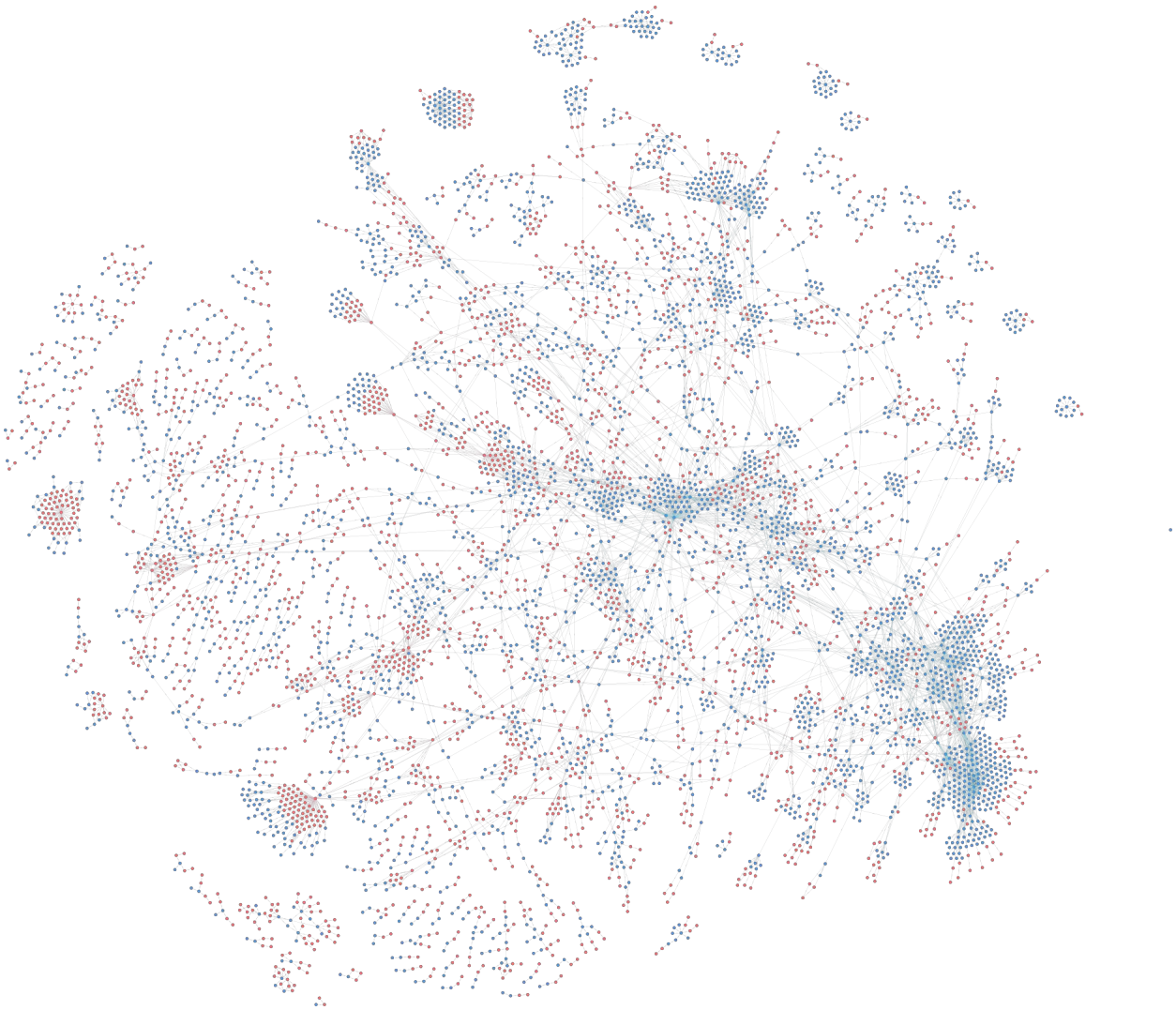

Fig. 5. Visualization of the whole *Graph 2*.

While much larger in size, *Graph 2* also showed an even lower average *degree* of 1.35 compared to *Graph 1*. None of the 5,961 nodes were in cliques of size 4 or greater. All six centrality metrics in Table 5 show that the centrality values are heavily skewed to the lower side as they did in *Graph 1*, indicating that this could be a common trait among verification strategies. Similar to *Graph 1* in the previous section, *Graph 2* also harbors a few extremely central hub nodes hinted by the sudden increase in the *betweenness* centrality as well as *articleRank* and *PageRank* values from p95 and max. Apart from the size differences, *Graph 2* differs in its cause of isolated nodes; the majority of them are a result of information-less cells in the source dataset. For example, a *Parameter* without the *reqId* attribute would become a graph node as it has a distinct description but was unable to connect to any other nodes in the graph as the identifier is empty. Detecting such entities with ease helps the verification engineers to review their strategies; unnecessary processes and artifacts can be shown in a form of nodes isolated from the main strategy. On the other hand, the detected nodes could represent a series of overlooked data entries and connections that were necessary but missing in the strategy as well. Isolated subgraphs in *Graph 2* tend to be targeted toward a specific aspect of product development, such as *functionalities in emergency or repair mode*. As such, different approaches taken by the engineers in the two companies can be observed by analyzing how verification strategy entities are (dis)connected to each other.

Table 5. Statistical summary of the centrality in *Graph 2*

| measure | min | p50 | p75 | p90 | p95 | max | mean |
|---|---|---|---|---|---|---|---|
| *articleRank* | 0.1500 | 0.1569 | 0.2042 | 0.2779 | 0.3862 | 4.0876 | 0.2053 |
| *betweenness* | 0.0000 | 0.0000 | 1.0000 | 1.0000 | 2.1250 | 29.7500 | 0.4410 |
| *closeness* | 0.0000 | 0.6667 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 0.4800 |
| *degree* | 0 | 1 | 1 | 2 | 4 | 66 | 1.3538 |
| *eigenvector* | 0.0000 | 0.0000 | 0.0007 | 0.0007 | 0.0013 | 0.4652 | 0.0014 |
| *pageRank* | 0.1500 | 0.1575 | 0.2775 | 0.5134 | 0.7027 | 8.8802 | 0.2754 |

Fig. 6 showcases two connected components from *Graph 2* representing subsets of verification activities. Fig. 6 (a) is rotated 90 degrees for better presentation. Compared to Fig. 4, Fig. 6 shows much larger connected components. The first two largest components are not shown here due to their large sizes, each having 3,724 and 214 nodes. Fig. 6 (a) illustrates a parameter-heavy graph, where the majority of the parameters are not directly *verified*. They are however verified indirectly, as all five hub-parameter nodes (*parameter_1308,1309,1312,1313,1314*) have direct connections with respective *Verification* nodes. Fig. 6 (b) shows a highly star-shaped subgraph centered around a single *Verification* node that has no direct connection to *Parameter* nodes. Instead, it *supports* many other *Verification* nodes which then *verify* several *Parameter* nodes together. This is likely for an independent verification task with an underlying assumption to be met. It is worth noting that not only the size of connected components but also their relative ratio are greater in *Graph 2*. 86.88% of the nodes were a member of the largest component in *Graph 1* while the two largest components only account for 66.06% of the nodes in *Graph 2*. This implies that *Strategy 2* utilizes slightly more independent tasks during the verification.

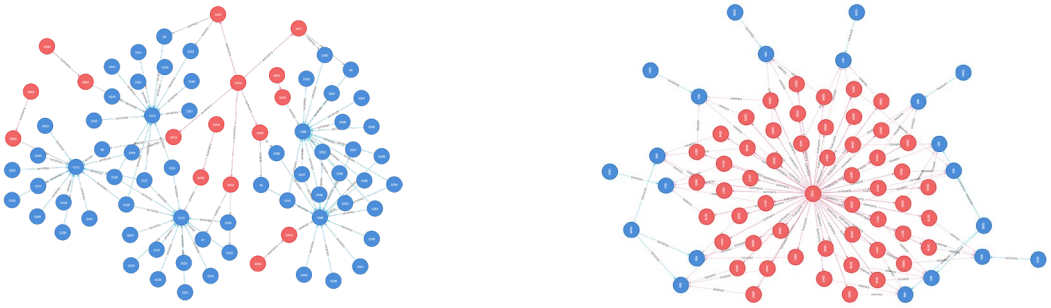

Fig. 6. (a) 3rd and (b) 4th weakly connected components in *Graph 2*.

## 4. Conclusions and limitations

We have shown in this paper knowledge graphs that capture the verification complexity of two real systems, one *small* medical device, and one defense system. The knowledge graph of the medical device captures the complete set of requirements and verification activities employed in the development of the system. The knowledge graph of the defense system is also comprehensive, although it misses requirements and verification information beyond the third level of decomposition.

The sizes of both verification strategies are several orders of magnitude larger than those of the use cases employed in verification engineering research. The medical device was designed against 129 requirements and underwent 179 verification activities and the defense system was designed against 3,285 requirements and underwent 2,576 verification activities (excluding both normative requirements), while use cases in existing research range up to 20 requirements and 20 verification activities. This provides evidence of research needed to scale state-of-the-art methods for verification planning and assessment.

The visual density of the graphs gives an idea of the complexity of the verification problem, in terms of size and interconnectedness. Some initial, basic graph metrics have been computed, showing a certain level of clustering. Further assessment of the meaning and implications of such properties is planned for future work.

The work presented in this paper is at an initial stage and presents some limitations. First, the dataset has not been curated. No attempt has been made at evaluating the quality of the different elements in the verification artifacts (for example, does a requirement statement really represent a requirement?). This is important to investigate how engineers make verification decisions. Second, we have not considered the content of the different verification sources, so there are limitations in accurately characterizing verification models, adequacy of the links between requirements and verification activities, etc. Third, the graphs have not been interpreted in the context of the engineering content within the verification artifacts. Future research is planned in this area.

## Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. CMMI-2205468.

## References

1. Collopy, P.D., *A research agenda for the coming renaissance in systems engineering*, in *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. 2012.
2. Collopy, P.D., *Report on the Science of Systems Engineering Workshop*, in *53rd AIAA Aerospace Sciences Meeting*. 2015, American Institute of Aeronautics and Astronautics.
3. Collopy, P.D. *Systems engineering theory: What needs to be done*. in *Systems Conference (SysCon), 2015 9th Annual IEEE International*. 2015.
4. Salado, A., R. Nilchiani, and D. Verma, *A contribution to the scientific foundations of systems engineering: Solution spaces and requirements.* Journal of Systems Science and Systems Engineering, 2017. **26**(5): p. 549-589.
5. Salado, A. and R. Nilchiani, *On the Evolution of Solution Spaces Triggered by Emerging Technologies.* Procedia Computer Science, 2015. **44**: p. 155-163.
6. Salado, A. and R. Nilchiani, *The Concept of Order of Conflict in Requirements Engineering.* IEEE Systems Journal, 2016. **10**(1): p. 25-35.
7. Collopy, P.D. and P.M. Hollingsworth, *Value-Driven Design.* Journal of Aircraft, 2011. **48**(3): p. 749-759.
8. Lee, B.D. and C.J.J. Paredis, *A Conceptual Framework for Value-driven Design and Systems Engineering.* Procedia CIRP, 2014. **21**: p. 10-17.
9. Wymore, A.W., *Model-based systems engineering*. 1993, Boca Raton, FL: CRC Press.
10. INCOSE, *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. version 4.0 ed. 2015, Hoboken, NJ, USA: John Wiley and Sons, Inc.
11. Buede, D.M. and W.D. Miller, *The Engineering Design of Systems: Models and Methods*. Third edition ed. 2016, Hoboken, NJ, USA: John Wiley & Sons, Inc.
12. Engel, A., *Verification, Validation, and Testing of Engineered Systems*. Wiley Series on Systems Engineering and Management, ed. A.P. Sage. 2010, Hoboken, NJ: John Wiley & Sons, Inc.
13. Salado, A. and H. Kannan, *Elemental Patterns of Verification Strategies.* Systems Engineering, 2019. **5**: p. 370-388.
14. Salado, A. and H. Kannan, *A mathematical model of verification strategies.* Systems Engineering, 2018. **21**: p. 583-608.
15. Hazelrigg, G.A., *A Framework for Decision-Based Engineering Design.* Journal of Mechanical Design, 1998. **120**(4): p. 653-658.
16. Hazelrigg, G.A., *An Axiomatic Framework for Engineering Design.* Journal of Mechanical Design, 1999. **121**(3): p. 342-347.
17. Buede, D.M., *The engineering design of systems: Models and methods*. 2009: Wiley.

18. Tribus, M., *Rational Descriptions, Decisions and Dsigns*. 1969, USA: Pergamon Press Inc.
19. Thompson, S.C. and C.J.J. Paredis, *An Investigation Into the Decision Analysis of Design Process Decisions.* Journal of Mechanical Design, 2010. **132**(12): p. 121009-121009-9.
20. Aughenbaugh, J.M. and C.J. Paredis, *The Value of Using Imprecise Probabilities in Engineering Design.* Journal of Mechanical Design, 2005. **128**(4): p. 969-979.
21. Salado, A. and H. Kannan, *Properties of the Utility of Verification*, in *IEEE International Symposium in Systems Engineering*. 2018: Rome, Italy.
22. ECSS, *Space engineering - Verification*. 2009, European Cooperation for Space Standardization: Noordwijk, The Netherlands.
23. ECSS, *Space engineering - Testing*. 2012, European Cooperation for Space Standardization: Noordwijk, The Netherlands.
24. Hamermesh, D.S., *Expectations, Life Expectancy, and Economic Behavior.* The Quarterly Journal of Economics, 1985. **100**(2): p. 389-408.
25. Smith, V.K., D.H. Taylor, and F.A. Sloan, *Longevity Expectations and Death: Can People Predict Their Own Demise?* The American Economic Review, 2001. **91**(4): p. 1126-1134.
26. Hoppe, M., A. Engel, and S. Shachar, *SysTest: Improving the verification, validation, and testing process— Assessing six industrial pilot projects.* Systems Engineering, 2007. **10**(4): p. 323-347.
27. Barad, M. and A. Engel, *Optimizing VVT strategies: a decomposition approach.* J Oper Res Soc, 2005. **57**(8): p. 965-974.
28. Xu, P. and A. Salado, *A Mathematical Approach to Design Verification Strategies that Incorporate Corrective Activities as Dedicated Decisions.* IEEE Open Journal of Systems Engineering, 2022: p. 1-10.
29. Kulkarni, A.U., et al., *An evaluation of the optimality of frequent verification for vertically integrated systems.* Systems Engineering, 2021. **24**(1): p. 17-33.
30. Messer, M., et al., *Model Selection Under Limited Information Using a Value-of-Information-Based Indicator.* Journal of Mechanical Design, 2010. **132**(12).
31. Engel, A. and S. Shachar, *Measuring and optimizing systems' quality costs and project duration.* Systems Engineering, 2006. **9**(3): p. 259-280.
32. Salado, A., H. Kannan, and F. Farkhondehmaal. *Capturing the Information Dependencies of Verification Activities with Bayesian Networks*. 2019. Cham: Springer International Publishing.
33. Farkhondehmaal, F. and A. Salado, *Efficient Population of the Verification Tradespace Using Bayesian Inference.* IEEE Systems Journal, 2020. **14**(3): p. 3225-3232.
34. Kulkarni, A.U., et al. *Is Verifying Frequently an Optimal Strategy? A Belief-Based Model of Verification*. in *ASME 2020 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. 2020.
35. Kulkarni, A.U., C. Wernz, and A. Salado, *Coordination of verification activities with incentives: a two-firm model.* Research in Engineering Design, 2021. **32**: p. 31-47.
36. Xu, P. and A. Salado. *A Concept for Set-based Design of Verification Strategies*. in *INCOSE International Symposium*. 2019. Orlando, FL, USA.
37. Xu, P., A. Salado, and G. Xie. *A Reinforcement Learning Approach to Design Verification Strategies of Engineered Systems*. in *IEEE International Conference on Systems, Man, and Cybernetics*. 2020. Toronto, Canada.
38. Xu, P., A. Salado, and X. Deng, *A Parallel Tempering Approach for Efficient Exploration of the Verification Tradespace in Engineered Systems.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2022: p. 1-13.