Limited circulation. For review.

Integration, Navigation and Exploration of Plant Topology Networks Using the Property-Graph Model

David Dorantes Romero, Nina F. Thornhill, *Senior Member, IEEE*

Abstract—Understanding the process topology is essential for many process systems engineering activities. Previous works in this area have explored the extraction of connectivity and causality from different sources by using structural and process data. A few works, however, have focused on how to integrate, store, and visualise connectivity models, which are becoming available in the literature. This paper proposes a method for integration, navigation and exploration of topology models in an efficient way, which can be applied to the study of connectivity and causality in chemical processes.

I. INTRODUCTION

Disturbances in continuous processes commonly travel along the product stream through different propagation paths affecting the process performance. Abnormalities that propagate plantwide include oscillations caused by sticking valves, disturbances produced by interacting controllers, instabilities in the systems such as slugging flows, and vibrations caused by interacting units (for example in a turbine-generator train).

Since large-scale chemical facilities are highly integrated systems containing recycle flows and utility systems, units are not independent anymore. Connectivity and causality have been concepts proposed for describing dependencies between units and variables [1]. Several authors have been applying new methods that make it possible to extract process topology describing causality relationships from process knowledge and process data. The knowledge from process topology has many applications, for instance in the field of plantwide analysis, where the use of process connectivity can enhance the isolation and diagnosis of the root-cause of plantwide disturbances [2], [3]. There is also evidence of use of topology in other areas such as alarm rationalization [4], risk assessment [5] and control structure design [6]. It seems now that mining the data and knowledge about topology (connectivity and causality) is a largely solved problem. What becomes challenging now is to integrate all these insights into a single topology model and make it possible for engineers to use it for exploration, navigation and more effective visualisation.

This work is supported Marie Curie FP7 project ENERGY-SMARTOPS, Contract No: PITN-GA-2010-264940.

David Dorantes Romero is with ABB Technology and Innovation, Norway, Ole Deviks vei 10, david.romero@no.abb.com. He is also part-time PhD student at the Department of Chemical Engineering at Imperial College.

Nina F. Thornhill is with the Centre for Process Systems engineering, Department of Chemical Engineering, Imperial College London, London SW7 2AZ, U.K.

The main contribution of this paper is the proposal of a method to integrate process connectivity and causality information into a plant topology network, making use of interaction and techniques to facilitate its navigation and exploration. Examples will also be given based on a novel prototype called Topoviz™, currently under development.

II. STATE OF THE ART

Information about the topology can be captured from model-based methods or inferred based on data-driven methods. A comprehensive overview of them has been compounded by Yang et al. in [1].

A. Representation of connectivity and causality

Applications from process topology data can be found in the field of performance assessment, or fault diagnosis. In order to understand and analyse the topology, all data is modelled and represented in some way. Commonly, this has been achieved either by using adjacency matrices [7], signed digraphs [8], and multi-level flow modelling (MFM) [9].

Adjacency matrices represent an abstract model of the plant and they are constructed from the parsed information from electronic versions from piping and instrumentation diagrams (P&ID). A reachability matrix can be obtained by linear algebra. This new matrix can be used to determine feasible propagation paths and determine the root-cause. One disadvantage of this method is that matrices are difficult to explore and visualise. Also, only binary relations can be defined (either it is connected or not) and in practice, not all the connections are equally important. Signed digraph approaches are based on the first-principles models that describe the system and convert them to signed digraphs. The resulting digraphs can be used to perform manual fault analysis. One limitation of this method is that digraphs need to be created manually by an expert. Hence they can be expensive to produce and error-prone. MFM methods provide a structured way to represent process knowledge and the different connectivity links via material, energy and information flows. The abstracted objects contain some properties about their functionality. However, this method relies on no standard and practical aspects of its implementations have not been adequately researched. The possibility of expanding structural models coming from the P&ID and adding data to from other types of schematics, electrical diagrams, work orders, alarm lists and maintenance calendars has been explored in [10], [11].

The most common limitation observed from a literature review is that topology models extracted from large-scale facilities are often difficult to visualise because there has been a lack of focus on the human factors, usability and interaction techniques applied to the field of plant topology.

B. Storing: Integrated networks vs. disparate sources

The above review has identified that one gap in the field of process systems engineering is on how to use and store process topology in a way that allows both interoperability and rapid searches. Nonetheless, other areas of knowledge such as bioinformatics and genomics have already dealt with similar problems when it comes to integrating and efficiently storing network models. The proposed approach is based on the notion of modeling topology information as a network, in contrast to having it stored in relational databases. Some reasons for that are that networks are easier to analyse and visualise than multiple tables, they are suitable for fast and powerful querying, and are intuitively object-oriented programming, fastest and optimal in terms of transversions and indexes.

C. Transforming and visualising topology models

The methods for querying topology information depend on the computer language used for its typification. In the case of ontology models such as OWL and RDF, some authors have used semantic query languages, such as SPARQL [12]; in contrast, LINQ to XML have been used [13] when exchange formats for plant data are used (IEC 62424 / ISO 15926). Methods to convert XML to RDF have been also investigated in the past [14]. Furthermore, tools are being developed for the web technologies as an attempt to enable the user to browse and navigate large and complex ontologies (millions of terms and multiple relationships). The utility and performance of a few ontology viewers was tested, including OLSVis [15] and OntoViz [16], in addition to other feature-rich software packages for network visualisation that are freely available, such as Sci2 [17], Gephi [18] and Cytoscape [19]. Despite the availability of such software, none of them fulfills all functional requirements because (a) they lack native support to industrial XML standards and (b) they do not provide an application programming interface (API) that would allow their use as visualisation layer for Topoviz™.

III. DEFINITIONS

A. Networks (graphs)

There are many definitions of the concept ‘network’ that are available. Most of them try to define the concept as a “group or system of interconnected entities.” Networks are inherently single-relational graphs. Examples of graph networks and their applications in other fields are given:

- Internet: describes the connectivity of routers, websites and network elements
- Transport: defines how places are connected over by air and road routes
- Biochemistry: explains how biological components interrelate (genes, proteins and pathways)

Most of the research in graph and network theory has looked at analyzing single-relational networks, in which links are of the same type. For example, the edges may either represent “collaboration, or “kinship”, but not together.

A. Plant topology networks

Plant topology networks are composed of actors (plant items or instrumentation) linked by relationships (for example material, electrical, and information flows). As plant topology networks are graphs, their analysis is closely related to the exploration of graphs in general. Relevant research in the topic is looking at industrial standards such as AutomationML [20] and ISO 15926 (XMpLant) which can integrate topology, geometry and control logic into a single XML file. The applicability and global use in the industry is still an open question.

B. The property-graph model

The main inputs to produce them are process flow diagrams (PFD), ‘smart’ P&IDs, or system control diagrams (SCD). Almost any plant that can be modelled in computer-readable format can be represented as ‘smart’ schematics. Smart’ schematics are XML files containing all metadata from the actual drawing, and they are easier to parse. There are several different graph data models that can be used for this integration, including property-graphs, hypergraphs, and triples. However, due to the following features it is argued that the property-graph model fits better: it is formed by vertices and edges, vertices have properties (key-value pairs), edges are named, have directionality, weight, connected vertex; and edges may also have properties. Figure 1 gives an overview of these elements.

Fig. 1. Overview of elements and attributes in a property-graph

IV. INTEGRATION OF PLANT TOPOLOGY

The integration of plant engineering data is essential so several domain experts of various engineering disciplines can interact with the data. In order to make this possible, analytical techniques and algorithms for processing graph are required. Graph theory and its extensions are mature and well-understood and suitable for direct application to the problems described in this paper.

Fig. 2. Visual representation of the idea for integrating connectivity from different sources into a plant topology network. Adapted from [12]

Figure 2 represents the overall methodology behind the Topoviz™ tool. The dots inside the bubbles represent process equipment and variables contained inside a data source (process schematic, electrical drawing, first-principles models, and cause and effect logic, among others), and the lines indicate the connections between them. An integrated graph would mean that all links can be stored and classified inside the same model so that the interactions between process-mechanical-electrical interfaces can be simply visualised.

A. Computational aspects

The problem of knowing if connectivity exist between two vertices in a graph can be efficiently solved by using a search algorithm. In general, it is simple to determine programmatically how the graph is connected, as well as to calculate the number of connected items. A simple pseudo-code of an algorithm that can calculate connectivity in the graph is presented below:

1. Initiate at any random vertex of the graph, G
2. Continue from that vertex via depth- or breadth-first search, keep a counter of all vertices reached.
3. Once G has been traversed, if the number of vertices counted is identical to the count of vertex of G, the graph is connected; else it is disconnected.

B. Implementation

1) Pure graph languages

There are two data models that are widely used in the field of network analysis, one is based on eXtensive Markup Language (XML) and the other in JavaScript Object Notation (JSON). GraphML is an XML-based file format for graphs. This format resulted from a joint effort from the graph drawing community that was lacking a standard format for exchanging graph structures.

The syntax is based on XML and supports most graph structures including directed, undirected, property-graphs, and hypergraphs. GraphSON is its counterpart based on the JSON notation. The idea behind the syntax follows the same principles as in the previous one. Both approaches are convenient for developers to work with as due to its simplicity when creating and parsing.

2) Graph libraries for Windows applications

Table I lists some libraries for graph analysis that are available as software tools. The important criteria to look at is whether it is possible to implement the framework in Windows applications, or in Windows Presentation Foundation (WPF), if it supports search and layout of the networks, and if it supports the modelling using the property-graph. Graphviz is a framework developed by AT&T Labs Research for drawing graphs indicated in DOT language scripts. DOT is a plain text graph description language, more information about this format can be found at [21]. Microsoft Automatic Graph Layout offers graph generation and dynamic layouts. Although MSAGL has been used in many other applications with a need to represent complex directed graphs (business management and manufacturing), the framework cannot fully produce a property-graph. Graph#, on the other hand is a graph visualisation framework built on top of an efficient library which provides generic directed or undirected graph data structures and generic algorithms from graph theory. It was decided that Graph# was the best alternative because it was the only WPF solution that allowed customized graph structures and supported the needs for search and layout.

TABLE I. LIBRARIES FOR DRAWING GRAPHS PROGRAMMATICALLY

Library	Capabilities			
	WPF	Search	Layout	Property-graph support
Graphviz		x	x	
MSAGL			x	x
Graph#	x	x	x	x

3) Analysis of topology networks in WPF

To be able to analyze topology networks modelled as property-graphs it is necessary to traverse the graph structure. From the implementation point of view, there are two approaches to follow in order to achieve the traversing algorithms. One is to use a domain-specific language such as Gremlin or Cypher (topology network needs to be stored in a Neo4j graph database), and the second one is to use custom libraries and scripts in WPF such as QuickGraph (topology network can be stored in memory).

Fig. 3. Proposed framework for transforming XML to topology networks

While it is true that graph-based programming languages are more powerful than WPF implementations, the second option was chosen. The reasons for this are given below: (1) Graph# (library used for graph visualisation) allows direct interoperability with QuickGraph data structures, (2) the size of the graphs used for testing this methodology are easy to handle in memory without the need of a Neo4j database. Graph languages can be used once the quantity of data calls for it.

C. Example network

An example system is presented based on the Imperial College Process Automation Rig model. A proprietary tool for visualisation and analysis for plant topology networks created previously by the authors (Topoviz™) is used in this study. Topoviz™ is a WPF application written in C# which uses Graph# for the integration engine, serialization, deserialization and layout. For more information about Topoviz™ and the Imperial College model please contact the authors.

1) Process Overview

The Imperial College Pilot Rig (Figure 5) comprises of two main process loops. In the first loop, water is circulated from the reservoir into the reactor via a pump. In the second loop, water from the reactor is passed through two electric heaters in parallel. The outlet of the heater will then flow through a short pipe or a long pipe, subject to the action of a 3-way valve, towards a cooler. The stream leaving the cooler is then transferred to a vertical sight glass via a pump, where it will be mixed with compressed air.

Fig. 5. Smart P&ID from Imperial College Pilot Rig

The outlet stream from the vertical sight glass is a two-phase flow, which is then transferred back into the reactor. The air in the mixture will be vented out in the reactor. The physical facility is over instrumented due to its use for teaching purposes. Nevertheless it provides a good example of a laboratory-scale process with a sufficient number of connections to demonstrate the methods.

2) Structure

Different schematics, electrical drawings, and cause and effect table have been input into Topoviz™ as XML. The internal engine of Topoviz's™ uses algorithms for parsing industrial XML and CSV files into the graph model, which can be easily serialised into GraphML (in memory) or SQL database for storage.

Figure 4 shows the resulting plant topology network. The network consists of 112 nodes and 130 edges. The graph has 32 strongly connected elements, 20 sinks, and 9 sources.

Fig. 4. Graph network rendered in Topoviz™.

V. EXPLORATION OF PLANT TOPOLOGY NETWORKS

Exploration in this context is understood as a query-type search, which allows the engineer to find feasible propagation paths as well as analyse in- and out- edges from each node.

A. Graph-search algorithms

Topoviz™ implements several classic algorithms for traversing the network, for example Hoffman-Pavley, Dijkstra, A* and Bellman-Ford for shortest- and k -th shortest path. Additionally, as well as algorithms for topological sort using Tarjan methods (strongly connected elements and least common ancestor) are also implemented. The output is shown both as a text in a display box as well as in a custom diagram containing a sub-graph with only the resulting nodes from the search algorithm for easier visualisation.

B. Example

Assume that there are indicators of oscillations in the measurements from the tags “H-209” and “R-002”. It would be useful to know (a) if there are feasible propagation paths between these tags, and (b) the carrier tags that will transport the disturbance from “H-209” to “R-002”. Using a query system the user can select the two tags and then choose the appropriate algorithm.

The number of paths, as well as a text description of the shortest path number one is presented by the user interface, as well as a mini-topology drawing with all involved components. Figure 6 shows the in- and out- edges from tag “R-002” which represents a pump in the P&ID. As expected, the algorithm has linked this node to the other elements around it inside the system, such as level indicators and thermocouples (loops T 231, L 122, L 122).

Fig. 6. Exploration of network using the query system in Topoviz™

Fig. 7. Example visualisation of in/out edges from a selected tag

VI. NAVIGATION OF PLANT TOPOLOGY NETWORKS

Information visualisation systems which generate topology networks should be designed so that they represent relational information in an effective way considering potential users [22]. Many algorithms which render graphs have been evaluated according to aesthetic criteria (symmetry, clutter, and edge crossings) and for computational efficiency, as well as the maximization of the human performance. For this design, the layout algorithm computes three relevant parameters: weight, link and unit types.

A. Information visualisation layer

Algorithms for the calculation of adjacencies in graphs and their visualisation are quite robust these days. The path visualisation method applies Dijkstra’s algorithm to all instance of the entities in the selected tags. As a result, the neighborhood is propagated throughout all involved graphs. To analyze a path in detail, it is only necessary to select a tag and scroll/zoom to navigate the path. Color codes and highlighting enhances the understanding of what is shown.

B. Example

The figure below show an example application of Topoviz™ for the navigation of a plant topology network. The shapes and sizes of the nodes are customized to give useful information to the users.

Fig. 8. Example visualisation of shortest propagation path

As visible in Figure 8, the user can get a good overview of the way the disturbance could possibly propagate between two tags. The user can further hover over the links and have detailed information of whether is a process, electrical or logical connection.

In order to add usability to the visualisation, the tags selected as origin and end are semi-highlighted as well as the ‘cheapest’ feasible propagation path between them. present, all weights inside the connectivity models have the same value, but research is being done in order to come with alternatives to distinguish between more- or less relevant links in the plant topology network. One possible way to establish the cost function in the edges can be associated to probability functions based on historical data.

VII. SUMMARY AND CONCLUSIONS

This paper has described a method for modelling a chemical process as an integrated topology network and the possible applications this can have in the field of fault management, such as root-cause analysis, risk analysis (HAZOP), alarm rationalization and automation engineering.

It has been argued that the requirements of process engineers for understanding the interactions between different systems in the plant are: to have an integrated model containing information about process connectivity and causality, and to have means to explore and navigate the data.

The article reports that XML, WFP and Graph# are suitable technologies to meet the requirements. The reasons for these recommendations are that these technologies can be implemented as modular Windows applications, using C# as programming language. Another reason is that there is already a great deal of work on parsing algorithms, and interaction libraries that can make the implementation more effective. Based on these technologies, integration of disparate databases such as process schematics and cause and effect logic can be achieved using the concept of the property-graph.

The presented results have shown the applicability of the integration and visualisation methodologies. Other relevant areas where the manipulation of topology information is significant are the electrical schematics and first-principles models (variable to variable relationships). Some example uses of navigation and exploration techniques based on graph theory were demonstrated in a case study based on a laboratory scale process. In the particular example presented here, it was possible to integrate structural and control connectivity from the P&IDs based on parsing algorithms and to draw inferences about the possible propagation paths between the units in the plant.

The use of standard ontologies for plant data is a fact that will determine to which adept the integration of metadata can be performed automatically. Remaining challenges are the scalability of the visualisation to a large number of systems, as well as the implementation of richer interaction methods and automatic extraction of a complete topology model.

ACKNOWLEDGMENTS

The financial support from the Marie Curie FP7-ITN project 'Energy savings from smart operation of electrical, process and mechanical equipment— ENERGY-SMARTOPS', Contract No: PITN-GA-2010-264940 is gratefully acknowledged. The contributions from Tone-Grete Graven and Trond Haugen are kindly acknowledged.

REFERENCES

- [1] F. Yang, P. Duan, S. L. Shah, and T. Chen, *Capturing Connectivity and Causality in Complex Industrial Processes*. Springer, 2014.
- [2] F. Yang and D. Xiao, "Progress in root cause and fault propagation analysis of large-scale industrial processes," *J. Control Sci. Eng.*, vol. 2012, pp. 1–10, 2012.
- [3] P. Duan, S. L. Shah, T. Chen, and F. Yang, "Methods for root cause diagnosis of plantwide oscillations," *AIChE J.*, vol. 60, no. 6, 2014.
- [4] M. Schlegelburg, L. Christiansen, N. F. Thornhill, and A. Fay, "A combined analysis of plant connectivity and alarm logs to reduce the number of alerts in an automation system," *J. Process Control*, vol. 23, pp. 839–851, 2013.
- [5] L. Cui, J. Zhao, and R. Zhang, "The integration of HAZOP expert system and piping and instrumentation diagrams," *Process Saf. Environ. Prot.*, vol. 88, pp. 327–334, 2010.
- [6] D. B. Alabi, "Automated Analysis of Control Degrees of Freedom," MSc thesis, Imperial College London, 2010.
- [7] J. Thambirajah, L. Benabbas, M. Bauer, and N. F. Thornhill, "Cause-and-effect analysis in chemical processes utilizing XML, plant connectivity and quantitative process history," *Comput. Chem. Eng.*, vol. 33, no. 2, pp. 503–512, Feb. 2009.
- [8] M. R. Maurya, R. Rengaswamy, and V. Venkatasubramanian, "Application of signed digraphs-based analysis for fault diagnosis of chemical process flowsheets," *Eng. Appl. Artif. Intell.*, vol. 17, no. 5, pp. 501–518, 2004.
- [9] J. Petersen, "Causal reasoning based on MFM," in *Proceedings of Cognitive Systems Engineering in Process Control (CSEPC2000)*, Taejeon, South Korea, pp.36-43.
- [10] D. D. Romero and T. G. Graven, "Visual representation of connectivity information for efficient system understanding," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 2013.
- [11] D. D. Romero, N. F. Thornhill, and T. G. Graven, "Towards the development of a tool for analysing plant-wide dependencies," in *ETFA2014*, 2014.
- [12] J. Persson, A. Gallois, A. Björkelund, L. Hafdel, M. Haage, J. Malec, K. Nilsson, and P. Nugues, "A knowledge integration framework for robotics . categories of knowledge used in robotized production exposing the semantics of AutomationML," in *International Symposium on Robotics ISR*, 2010, pp. 1–8.
- [13] M. Barth and A. Fay, "Efficient use of data exchange formats in engineering projects by means of language integrated queries — Engineers LINQ to XML," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 1335–1340.
- [14] I. Miletic and M. Vujasinovic, "Enabling semantic mediation for business applications: XML-RDF, RDF-XML and XSD-RDFS transformations," in *Enterprise Interoperability II*, 2007.
- [15] S. Vercruyssen, A. Venkatesan, and M. Kuiper, "OLSViz: An animated, interactive visual browser for bio-ontologies," *BMC Bioinformatics*, vol. 13, no. 116, 2012.
- [16] M. Sintek, "OntoViz Tab: Visualizing Protege Ontologies," 2003. [Online]. Available: <http://protege.wiki.stanford.edu/wiki/OntoViz>. [Accessed: 03-Jun-2014].
- [17] Sci2 Team, "Science of Science (Sci2) Tool," *Indiana University and SciTech Strategies*.
- [18] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," *ICWSM Meeting*, The AAAI Press, 2009.
- [19] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Scwikowski, and T. Ideker, "Cytoscape: A software environment for integrated models of biomolecular interaction networks," *Genome Res.*, vol. 13, no. 11, pp. 2498–2504, 2003.
- [20] L. Hundt, R. Drath, A. Lueder, and J. Peschke, "Seamless automation engineering with AutomationML," in *International Conference on Concurrent Enterprising*, 2008.
- [21] Microsoft, "The DOT language," 2014. [Online]. Available: <http://www.graphviz.org/content/dot-language>. [Accessed: 24-Mar-2014].
- [22] H. . Purchase, "Effective information visualisation: a study of graph drawing aesthetics and algorithms," *Interact. Comput.*, vol. 13, no. 2, pp. 147–162, Dec. 2000.