# Graph-Based Manipulation Rules for Piping and Instrumentation Diagrams

Johannes Bayer*, Arka Sinha*

* Smart Data and Services
DFKI, Kaiserslautern, Germany
E-mail: {johannes.bayer, arka.sinha}@dfki.de

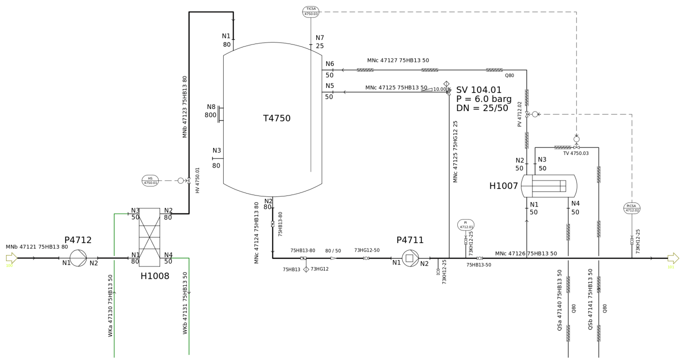


Fig. 1: Sample P&ID Structure [1]

***Abstract*—Piping and Instrumentation Diagrams (P&IDs) are key documents during the construction of chemical plants and other facilities in the area of process engineering. They can be described by a graph structure in which the pipes and signal lines are the connecting edges between individual equipment nodes (e.g. reactors, valves, sensors, controllers, actuators).**

**When digitizing P&IDs from paper-based and similar optical sources, a first step is the extraction of a graph of connected symbols. However, additional enrichment and corrections are needed in order to achieve graph structure that fully captures the given input in a semantically meaningful way.**

**In this paper, an approach in which graph-based rules are used to modify the P&ID-describing graph structures is presented. The objective is to enable the user to conveniently generate customized rules for different scenarios. The approach is evaluated by a survey of sample graphs.**

***Index Terms*—Piping and Instrumentation Diagram, Graph Refinement, Rule Graph**

## I. Introduction

As part of the design process of chemical plants in the process industry, Piping and Instrumentation Diagrams (P&IDs) are used to visualize their structure. They comprise of a set of symbols representing the equipment of the plant like vessels, sensors and actuators. The physical connections between such equipment are also depicted as drawn through lines for pipes or dashed lines for signal propagation, respectively (see fig. 1). Naturally, this structure can be considered a graph.

Historic P&IDs are often stored on paper or in crude digital image formats that lack semantic information. When extracting a graph structure from such visual media, nodes are obtained by matching image parts against a set of known symbols [2] [3], and edges are inferred by evaluating lines optically connecting them. However, further enrichment and corrections are required for an overall semantically meaningful description, which is utilized in a subsequent CAD import. In the paper at hand, this semantic gap is attempted to be bridged by automatic graph manipulation using graph-based rules.

## II. Related Work

### A. Graphs

In this paper, an undirected graph structure $G(N, E)$ is defined as a set of nodes $N$ and a set of connecting edges $E \subseteq N \times N$.

Furthermore, labeling functions for nodes and edges are considered: $f_N : N \mapsto C_N$ and $f_E : E \mapsto C_E$

### B. Subgraph Isomorphisms

Given two graphs $G_1(N_1, E_1)$ and $G_2(N_2, E_2)$, a subgraph isomorphism $f_I : N_1 \mapsto N_2$ is defined as an injective map from the nodes of the first to the second graph so that the graph structure of the first graph is fully contained (eq. 1). Furthermore, restrictions on the node (eq. 2) and edge (eq. 3) labeling are imposed.

$$(n1, n2) \in E_1 \implies (f_I(n_1), f_i(n_2)) \in E_2 \tag{1}$$

$$f_{N_1}(n) = c \implies f_{N_2}(f_I(n)) = c \tag{2}$$

$$f_{E_1}(e) = c \implies f_{E_2}(f_I(n)) = c \tag{3}$$

### C. Graph Refinement

Graph manipulations have been studied for various purposes over the years. Depending on what is represented via graph, the purpose of graph manipulation can vary. Friend connections on social media is often viewed as connected graphs. These platforms are often subject to attacks that can compromise personal data of a particular user and users connected to him/her. Zhou et al., 2008 [4] proposed a method to manipulate the social network graph to anonymize the node information and reducing the number of links as much as possible so that attackers cannot exploit the connections to have access to users private data. Atzmueller et al., 2016 [6], in their HypGraphs approach, used weighted networks to compare graph-based and sequential hypotheses using first order Markov chain models. They analyzed sequential state transitions and modeled them using graphs. Zhao et al, 2015 [7] proposed a method to use graph to represent similarities between labeled and unlabeled images which can be a powerful tool to deal with scarcity of labeled data for training neural networks.

Storing graphs in database and later able to query them is a powerful way to search and manipulate graph structures. Chau et al.,2008 [8] in their work demonstrated how to effectively store graphs and using query to find patterns. In their GRAPHITE system, they used G-Ray algorithm to find subgraph within a graph. Their system's interface lets user to intuitively draw a subgraph which is then used for matching. Later Pienta et al., 2017 [9] went further by creating a system called VISAGE which facilitates users to not only query a graph through subgraph matching, but to also help them build up the query from abstract to specific through a technique which they termed as graph-autocomplete. Both the work however, was limited to pattern matching.

All of these approaches inspired us to use the potential of using graph matching and push it further to manipulate and refine the subgraph part for our use case. Using graphs for P&IDs has been investigated in the past (for example Atzmueller et al., 2016 [6] in their HypGraphs approach). Our aim was to use this concept of transforming paper-based P&IDs in to graphs and later using graph matching and graph manipulation for enhancement and corrections.

## III. Methodology

### A. Modeling P&IDs as Graphs

The basic concept of modeling all domain-specific symbols of an P&ID as nodes and their connecting lines as edges needs to be refined slightly for a practical graph representation. For decreasing complexity, each line segment is modeled as an independent edge. Two adjacent line segments are therefore connected via *junction* nodes.

As of now, node and edge labels have been considered simple elements of a set of possible labeling. However, in the application scenario nodes and edges usually carry multiple different properties. These include domain-specific parameters (like the operating temperature of a vessel or the diameter and material of a pipe) as well as layout information such as position and size and most importantly the symbol type of a symbol. Therefore, the labels of individual nodes and edges are considered lists of *attributes*.

### B. Modeling Modification Rules as Graphs

In the following, the P&ID-describing input graphs (obtained by a symbolic graph extraction) are referred to as *original graphs*. The proposed *rule graphs* are applied to them as described below. A graph generated during the application of a rule graph to an original graph is referred to as *resulting graph*.

A rule graph is based on two disjoint node sets: the *isomorphic part* represents parts of the original graph. The existence of a subgraph isomorphism from this part to the original graph is considered to be the prerequisite for a successful application of the rule. The *alternation part* describes the modifications introduced (see fig. 2, fig. 3 and fig. 4):

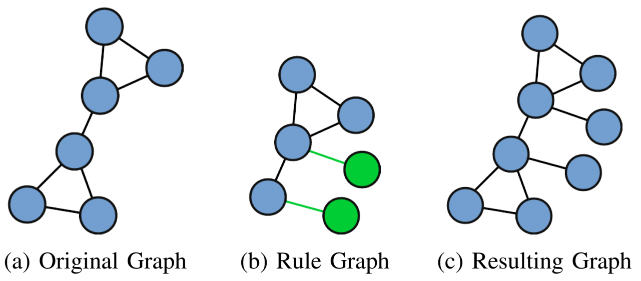

(a) Original Graph (b) Rule Graph (c) Resulting Graph

Fig. 2: Sample Application of a Rule for Adding Nodes and Edges

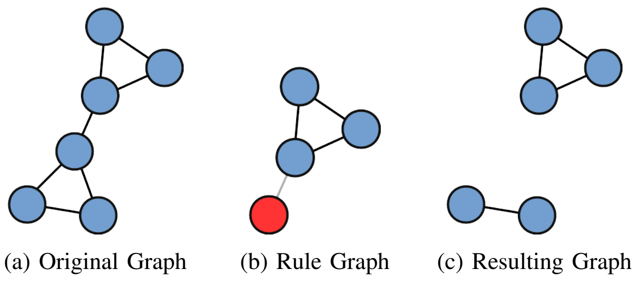

(a) Original Graph (b) Rule Graph (c) Resulting Graph

Fig. 3: Sample Application of a Rule for Deleting a Node.

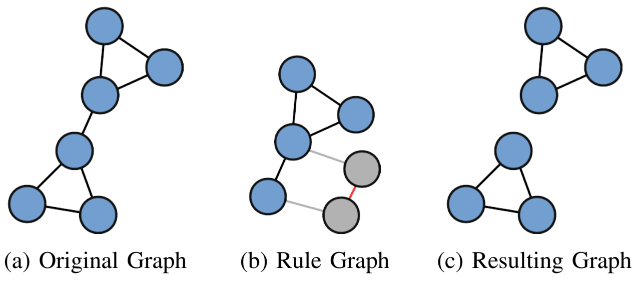

(a) Original Graph (b) Rule Graph (c) Resulting Graph

Fig. 4: Sample Application of a Rule for Deleting an Edge

*1) Original Node:* Node representations of the original graph (blue circles).

*2) Original Edge:* Edge representations of the original graph (black lines), only allowed between original nodes.

*3) Transition Edge:* Edges between original nodes and substitution nodes (gray lines).

*4) Insertion Node:* Nodes to be added to the original graph (green circles).

*5) Substitution Node:* Nodes that should remain in the original graph structure, but are either altered in labeling or connect inserted/substituted/deleted edges (gray circles). An unlabeled substitution nodes indicates no changes in the original node it relates to.

*6) Deletion Node:* Nodes to be removed from the original graph along with their connecting edges (red circles).

*7) Insertion Edge:* Edges to be inserted to the original graph (green lines).

*8) Substitution Edge:* Edges of the original graph to be relabeled (gray lines).

*9) Deletion Edge:* Edges to be removed from the original graph (red lines).

### C. Comparing and Manipulating Node and Edge Labels

In order to reduce the complexity of rule graphs, their nodes and edges bear labels that only describe the attributes that are either matched against or altered. More precisely, a match is considered successful, if the attribute values present in the isomorphic graph part of the rule graph are present in the original graph. A label in a substitution node or edge contains only the attributes in the label that should be altered.

### D. Multiple Matches and Applications

If there are multiple subgraph Isomorphisms between the original graph and a rule graph, one of them is (randomly) selected for the rule application.

## IV. Experiment

In order to demonstrate the use of the rule graph system, the following scenario is considered: Thermal pipe insulation is often depicted as discrete symbols in P&IDs (see fig. 5a). Therefore, a image-processing based symbol graph extraction pipeline yields graphs in which the insulation is described as nodes (see fig. 5b). However, describing the pipe insulation as an edge attribute is semantically preferable. In order to resolve this situation (see fig. 5d), a rule graph is used that replaces nodes labeled as insulation by edges with the same labeling (see fig. 5c).

## V. Future Work

As of now, only single applications and connected rule graphs are considered. Allowing for multiple applications of sets of rule graphs or single rule graphs of higher complexity (e.g. intermediate states) could allow for more powerful overall modifications.

Currently, the structure of rule graphs is very restricted. For example, expressing the non-existence of elements in the isomorphic rule graph part is desirable. Further investigations and an elaboration of the theoretical foundations of the technique are needed assess and extend its computational power. Nevertheless, the system should remain comprehensible by the user.

In order to make the imposed modifications more comprehensible, further visualizations are needed displaying the subgraph matching of original graph elements and how they are modified.

Apart from that, applying the technique to other domains remains to be investigated.

## References

[1] Data Exchange in the Process Industry: Test Cases Repository, https://gitlab.com/dexpi

[2] Moreno-garcia, C., Elyan, E. & Jayne, C. New trends on digitisation of complex engineering drawings. *Neural Computing And Applications*. **31**, 1695–1712 (2019)

[3] Kang, S., Lee, E. & Baek, H. A Digitization and Conversion Tool for Imaged Drawings to Intelligent Piping and Instrumentation Diagrams (P&ID). *Energies*. **12**, 2593 (2019)

[4] Zhou, Bin, and Jian Pei. ”Preserving Privacy in Social Networks Against Neighborhood Attacks.” ICDE. Vol. 8. 2008.

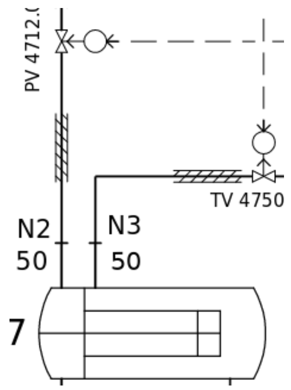
[5] Zhao, Mingbo, et al. ”Automatic image annotation via compact graph based semi-supervised learning.” Knowledge-Based Systems 76 (2015): 148-165.

[6] Atzmueller, Martin, et al. ”HypGraphs: an approach for analysis and assessment of graph-based and sequential hypotheses.” International Workshop on New Frontiers in Mining Complex Patterns. Springer, Cham, 2016.

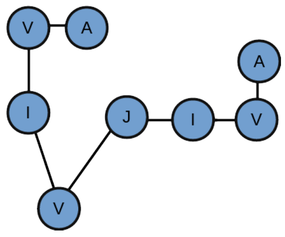
[7] Zhao, Mingbo, et al. ”Automatic image annotation via compact graph based semi-supervised learning.” Knowledge-Based Systems 76 (2015): 148-165.

[8] Chau, Duen Horng, et al. ”Graphite: A visual query system for large graphs.” 2008 IEEE International Conference on Data Mining Workshops. IEEE, 2008.

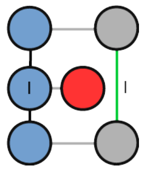
[9] Pienta, Robert, et al. ”Visual graph query construction and refinement.” Proceedings of the 2017 ACM International Conference on Management of Data. ACM, 2017.



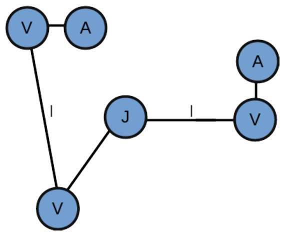
(a) P&ID



(b) Original Graph



(c) Rule Graph



(d) Resulting Graph

Fig. 5: A P&ID is transformed into a graph and refined by a rule