Assembling coherent network topologies using round-trip graphs*

Marino Miculan[1] and Matteo Paier[1,2]

[1] DMIF, University of Udine - marino.miculan@uniud.it,
[2] IMT Scuola Alti Studi, Lucca - matteo.paier@imtlucca.it

Abstract. Discovering the network topology in computer networks is challenging due to limited communication and incomplete information about non-immediately connected nodes. In this paper we address the problem of assembling partial views obtained by discovery tools into a coherent representation, using *round-trip graphs*: labelled bipartite directed graphs representing the communications between hosts, interfaces, and networks. A merge operation is introduced, facilitating compositional and incremental assembly of partial views. This research provides a practical solution for incrementally constructing a comprehensive network topology.

Keywords: Graph theory · Round-trip graphs · Network discovery

1 Introduction

Discovering a network's topology from inside the network itself is notoriously difficult [3]. Tools such as `traceroute` allow for scanning the network from the host on which they are executed, but these scans produce *partial* views of the network, because not all communications are allowed between all nodes (e.g., due to the presence of firewalls) and some information about non-immediately connected nodes may not be observed (e.g., MAC addresses).

To obtain a better understanding of the network, scans can be performed from different starting points, yielding different partial spanning trees of the same network. The problem that arises at this point is: given many partial trees of this kind, how can they be assembled into a coherent view of the entire network?

In this paper, we address this problem in terms of graph theory. More specifically, we introduce *round-trip graphs*, which are bipartite directed (multi)graphs, where nodes represent hosts, interfaces and networks, and labelled edges represent the type of communication allowed between nodes. The output of a scan produces such graphs, more specifically corresponding to directed acyclic graphs (DAGs).

We present a "merge" operation for these graphs, called *amalgamation*. This operation is associative, allowing us to assemble the partial views in a compositional and incremental manner. This operation can be computed in linear time with respect to the size of the input graphs.

* Work supported by the Department Strategic Project of the University of Udine within the Project on Artificial Intelligence (2020-25), and the project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU.

Overall, this research provides a practical solution for incrementally constructing a comprehensive network topology; moreover, the graph models we introduce in this work can be used for the optimal planning of network scans.

We proceed as follows. In Section 2 we recall the basic definitions about labelled graphs, and introduce the notions of *grounding* and *amalgamation*. In Section 3 we introduce round-trip graphs, and provide an example application. In Section 4 we present some conclusions and directions for future works.

2 Directed graphs, grounding and amalgamation

Let $G = (V, E)$ be a graph with vertex set V and edge set $E \subseteq V \times V$. G is a bipartite graph with labelled edges if there exists a partition of V into two disjoint sets V_1 and V_2, such that $E \subseteq V_1 \times V_2 \cup V_2 \times V_1$ [2].

More formally, let N be a fixed set of nodes, L be a fixed a set of labels.

Definition 1. *A* labelled bipartite directed graph *(shortly,* graph*) is a tuple* $G = (V_1, V_2, E_1, E_2)$ *such that* V_1, V_2 *are two disjoint finite subsets of* N*, and* $E_1 \subseteq V_1 \times L \times V_2$ *and* $E_2 \subseteq V_2 \times L \times V_1$*. For a graph* G*, its components are denoted as* V_1^G, V_2^G*, etc. We use the shorthands* $V = V_1 \cup V_2$ *and* $E = E_1 \cup E_2$*.*

Intuitively, we will use nodes to represent parties of a network, while labelled edges denote the types of communication allowed between these parties.

Several operations can be defined on graphs. Let G, H be graphs.

Graph union: $G \cup H = (V_1^G \cup V_1^H, V_2^G \cup V_2^H, E_1^G \cup E_1^H, E_2^G \cup E_2^H)$.

Renaming: A *(node) renaming* is a function $\sigma : N \to N$ (not necessarily injective). It is extended pointwise to sets of names. Moreover, if $E \subseteq N \times L \times N$, then $E[\sigma] = \{(\sigma(x), l, \sigma(y)) \mid (x, l, y) \in E\}$.
The *renaming* of G under σ is $G[\sigma] = (\sigma(V_1), \sigma(V_2), E_1[\sigma], E_2[\sigma])$.

Proposition 1. *1. Composition of substitution:* $G[\sigma \circ \sigma'] = G[\sigma'][\sigma]$
2. Renaming distributes over union: $(G_1 \cup G_2)[\sigma] = G_1[\sigma] \cup G_2[\sigma]$

Definition 2. *Let* G *be a graph. A* labelled path *(of length* k*) in* G *is a list* $\alpha = (n_0, l_0, n_1, l_1, \ldots, l_{k-1}, n_k)$ *where for all* $i \in \{0, \ldots, k-1\} : (n_i, l_i, n_{i+1}) \in E^G$*. If the path is all labelled with the same label* l*, it is called* l*-path.*

We say that there is a l/h-round path *from* n *to* m*, denoted as* $n \xrightarrow{l/h} m$ *if there exists a* l*-path from* n *to* m *and a* h*-path from* m *to* n*.*

An unlabelled path *(of length* k*) in* G *is a list* $\alpha = (n_0, n_1, \ldots, n_k)$ *such that for* $i \in \{0, \ldots, k-1\}$ *there exists* l_i *such that* $(n_i, l_i, n_{i+1}) \in E^G$*.*

In the following by "path" we mean unlabelled path, unless differently stated.

2.1 Grounding and Amalgamation

We will use graphs to represent (partial) knowledge about the network. In order to deal with incomplete information we have to distingush between consolidated knowledge, like that directly observable from a node, and hypothetical knowledge,

i.e., guesses about parts of the network not directly observable. This knowledge can be refined, e.g. by further observations on the network. To this end, we fix a set $M \subseteq N$ of nodes, that we call *ground*. Ground nodes are those whose identity is consolidated. Non-ground nodes are those whose existence is assumed or deduced from the context, but still to be verified. This verification happens when merging the knowledge obtained from different points of view, that is, different graphs sharing some ground nodes and paths.

Definition 3 (Ground and suspended paths). *A path α is* ground *if all nodes in it are ground. We denote by $\mathsf{gpath}(G, n, m)$ the set of all ground paths in G between n and m, and by $\mathsf{gpath}(G)$ the set of all ground paths in G.*

Let $n, m \in M$. A path α is suspended between n and m *if the first and last nodes of α are n and m respectively, and all intermediate nodes are not ground (i.e., not in M). Let us denote by $\mathsf{spath}(G, n, m)$ the set of all suspended paths between n and m in G.*

In our application, we aim to build a graph whose ground paths form a spanning tree. Therefore, in order to be consistent, a graph can contain at most one ground path between any two nodes.

Definition 4. *A graph G is* sound *if, for every $n, m \in M$, $|\mathsf{gpath}(G, n, m)| \leq 1$.*

When merging two graphs, we may resolve the uncertain knowledge of suspended paths using ground paths between the same nodes. This operation is called *grounding*, and it is defined next.

Definition 5 (Grounding). *Let $\alpha = (n_0, \ldots, n_k)$ be a ground in a sound graph G, and let β be a suspended path between n_0 and n_k, of the same length k. The* grounding of β on α *is a partial substitution $\theta(\alpha, \beta) : N \rightharpoonup N$ defined as follows:*

$$\theta(\alpha, \beta)(n) = \begin{cases} n_j & \text{if } \beta = (m_0, \ldots, m_k) \text{ and } m_j = n \\ \bot & \text{otherwise.} \end{cases}$$

Let H be another graph. The grounding of H on G *is the graph $H[\sigma_{H/G}]$ obtained by grounding all suspended paths in H using the corresponding ground paths in G. Formally, $\sigma_{H/G} : N \to N$ is defined as follows:*

$$\sigma_{H/G} = \bigcup \{\theta(\alpha, \beta) \mid \alpha \in \mathsf{gpath}(G), \alpha = (n, \ldots, m), \beta \in \mathsf{spath}(H, n, m), |\alpha| = |\beta|\}$$

In other words:

$$\sigma_{H/G}(n) = \begin{cases} n_j & \text{if there exists } \alpha \in \mathsf{gpath}(G), \alpha = (n_0, \ldots, n_k), \\ & \beta \in \mathsf{spath}(H, n_0, n_k), \beta = (n_0, n'_1, \ldots, n'_{k-1}, n_k), n'_j = n \\ n & \text{otherwise.} \end{cases}$$

This definition is well given because G is sound and hence α, if it exists, is unique. An example of path grounding is in Fig. 1.

We are almost ready to provide the core definition of our theory, i.e., how two different (sound) views of the same network can be coherently merged, yielding a new (sound) view of the network. We call this operation *amalgamation*.

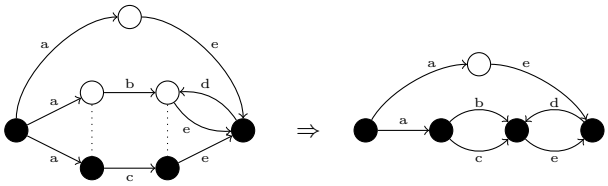


Fig. 1: Grounding of suspended paths. Black nodes represent ground nodes. The dotted mapping represent σ. Notice that suspended paths cannot be grounded on ground paths of different lengths.

Definition 6. *Two graphs G_1, G_2 are* compatible *if for all $\alpha = (n, \ldots, m) \in \mathsf{gpath}(G_1)$ and $\beta = (n', \ldots, m') \in \mathsf{gpath}(G_2)$, if $n = n'$ and $m = m'$ then $\alpha = \beta$.*

Lemma 1. *For G_1 and G_2 two compatible sound graphs, $G_1 \cup G_2$ is sound.*

Definition 7 (Amalgamation). *Let G_1, G_2 be two compatible sound graphs. The* amalgamation *of G_1 and G_2 is the graph obtained by grounding the union of G_1 and G_2 with the information of both: $G \amalg H \triangleq (G \cup H)[\sigma_{G \cup H / G \cup H}]$.*

Proposition 2. *Let G_1, G_2 be two compatible sound graphs. Then:*

1. *Soundness: $G_1 \amalg G_2$ is sound;*
2. *Neutral element: $G_1 \amalg 0 = G_1$ (where 0 is the empty graph);*
3. *Symmetry: $G_1 \amalg G_2 = G_2 \amalg G_1$;*
4. *Associativity: $(G_1 \amalg G_2) \amalg G_3 = G_1 \amalg (G_2 \amalg G_3)$.*

This result allows us to construct incrementally partial views of the network: we start with a partial view (i.e., the result of a scan), and every time we obtain a new (partial) sound view of the network, we can add it to the current graph.

Proposition 3. *The amalgamation of two compatible sound graphs G_1, G_2 can be computed in $O(|E^{G_1} \cup E^{G_2}|)$.*

Proof. Follows from the fact that union and grounding are $O(|E^{G_1} \cup E^{G_2}|)$.

3 Round-trip graphs

We now define round-trip graphs, i.e., labelled bipartite directed graphs designed for representing the information that can be observed on computer networks using tools like `traceroute` [5].

Definition 8. *A* round-trip graph *is a labelled directed graph G where*

ground nodes *are of four kinds: CPU nodes (*_cpu); Layer nodes (*_layer3); Interface nodes (*_eth); Network nodes (*.in-addr.arpa).*

non-ground nodes *are as above, but with a ? in the name (e.g., A_eth?);*

edges *can connect only: Layer nodes to CPU nodes and to interface nodes (and vice versa); interface nodes to network nodes (and vice versa);*

edge labels *are from the set {icmp0, icmp8}.*

It is easy to see that round-trip graphs are bipartite: one side are CPU and interface node, the other is layer and network nodes. Hence we can readily apply the theory developed in the previous Section. In particular:

Definition 9. *A* round-trip *from n to m is a icmp0/icmp8-round path from n to m.*

Tools like `traceroute` (normally) produce round-trips of minimal length, where only a few nodes at the beginning of the path and the final one are ground; all the other nodes in between are not ground. By amalgamating all the round-trips from a given starting point in the network, we obtain a graph similar to a DAG.

As an example, let us consider a network with six hosts A, B, C, D, E, F, connected via four networks, as represented by the round-trip graph in Fig. 2d. The scans from A produce the round-trip graph in Fig. 2a, while from B we can construct the graph in Fig. 2b. Nodes with light color are not ground. Amalgamating these two graphs, we obtain the graph in Fig. 2c: comparing with the actual one (Fig. 2d) we see that the only suspended paths are those which have not been observed neither from A nor from B.

4 Conclusions

In this paper we have shown how to merge network graphs, in order to create incrementally a coherent view from partial views of the same network.

As future work, we plan to consider that the starting views could be not accurate (e.g., due to accessibility policies) and thus the resulting amalgamation could be incomplete. To this end, a hierarchical model of the network could be useful; hence we plan to extend the theory of this paper from directed graphs to *directed bigraphs* [1, 4]. Finally, we would like to implement some tools that use this amalgamation notion to reconstruct topologies of real-world networks, as it will prove useful to sys-admins and network security experts.

References

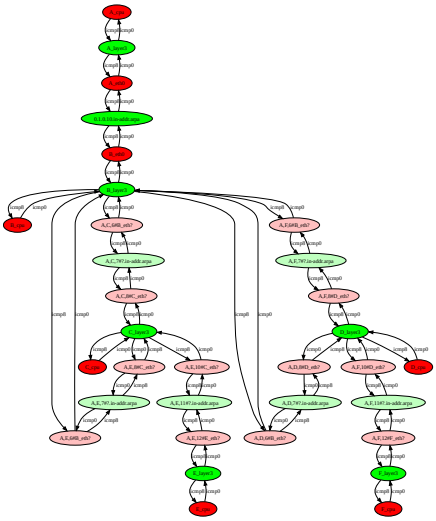
[1] G. Bacci, D. Grohmann, and M. Miculan. Dbtk: A toolkit for directed bigraphs. In *Proc. CALCO 2009*, volume 5728 of *Lecture Notes in Computer Science*, pages 413–422. Springer, 2009.

[2] J. Bang-Jensen and G. Z. Gutin. *Digraphs - Theory, Algorithms and Applications, Second Edition*. Springer, 2009.

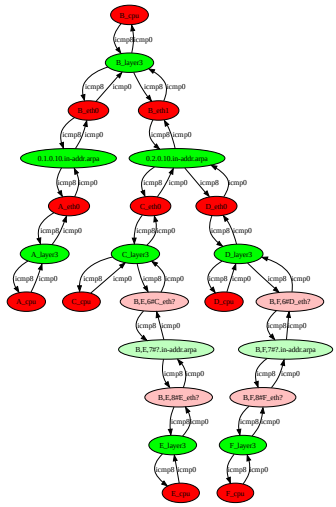
[3] Z. Beerliova, F. Eberhard, T. Erlebach, A. Hall, M. Hoffmann, M. Mihal'ak, and L. S. Ram. Network discovery and verification. *IEEE Journal on selected areas in communications*, 24(12):2168–2181, 2006.

[4] D. Grohmann and M. Miculan. Directed bigraphs. In *Proc. MFPS 2007*, volume 173 of *ENTCS*, pages 121–137. Elsevier, 2007.

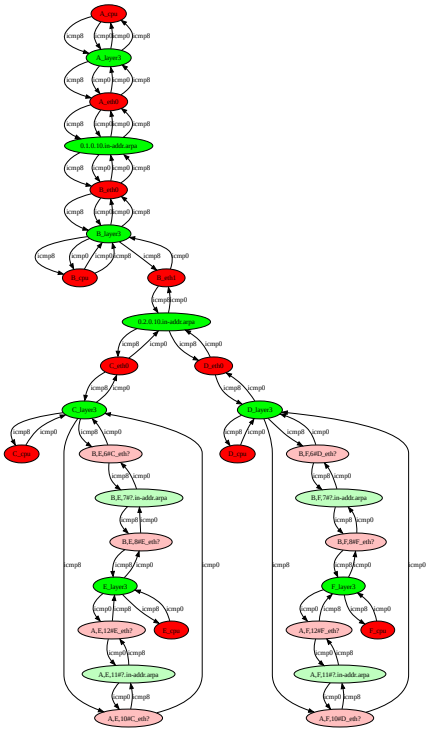
[5] G. S. Malkin. Traceroute Using an IP Option. RFC 1393, Jan. 1993. DOI: 10.17487/RFC1393. URL: https://www.rfc-editor.org/info/rfc1393.



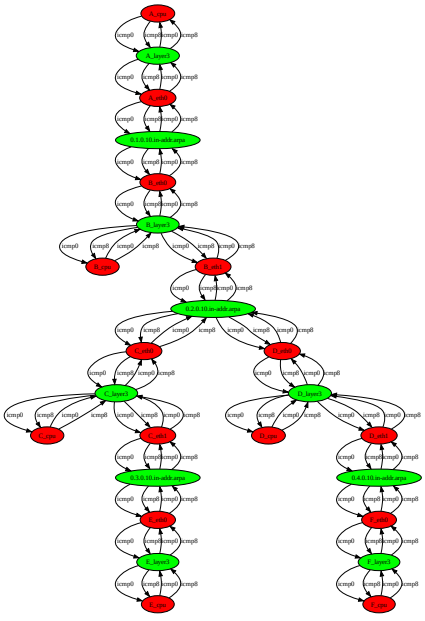
(a) Graph induced by traceroutes from A.



(b) Graph induced by traceroutes from B.



(c) Amalgamation of the graphs in Figs. 2a and 2b.



(d) The actual network.

Fig. 2: Amalgamation of round-trip graphs induced by traceroutes