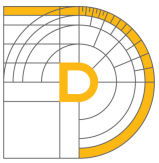
USC Viterbi
School of Engineering
Information Sciences Institute



DETER Technical Report
ISI-TR-684
August 2012

Montage Topology Manager: Tools for Constructing and Sharing Representative Internet Topologies

Authors: Alefiya Hussain and Jennifer Chen

This material is based upon work supported by the Department of Homeland Security, and Space and Naval Warfare Systems Center, San Diego, under Contract No. N66001-10-C-2018.

Any opinions, findings, and conclusions or reccommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Deparment of Homeland Security for the Space and Naval Warfare Systems Center, San Diego.

USC Information Sciences Institute • 4676 Admiralty Way • Marina del Rey, CA 90202

Montage Topology Manager: Tools for Constructing and Sharing Representative Internet Topologies

Alefiya Hussain
USC/Information Sciences Institute
hussain@isi.edu

Jennifer Chen
USC/Information Sciences Institute
jchen@isi.edu

ABSTRACT

Modeling representative topological structures for experimentation has proved to be a challenging task. Our personal experience suggests most experimenters evaluate networking and cyber security systems with small scale topologies, significantly limiting the scientific process and rigor in their experiments. In this paper, we first survey the current topology tools and discuss how the affordances and constraints of these tools influence how experimenters design their investigations. We then present a topology manager that integrates a suite of tools for topology generation, composition, and validation, with an interactive and intuitive graphical interface. These tools provide a collaborative environment where experiments can document, share, and reuse topologies thus building on their work and the work of other researchers. We demonstrate how the tools can be used to develop two non-trivial networking topological structures: an Internet2 topology and the combined Italian power grid and communication network topology.

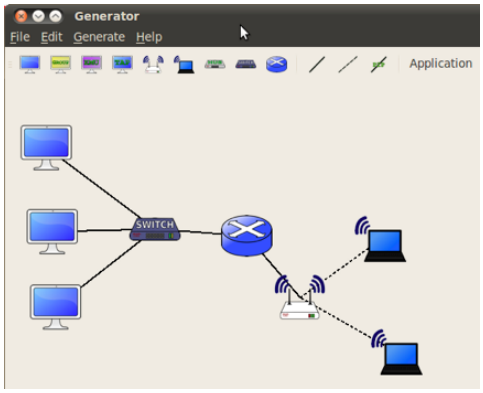
1. INTRODUCTION

Networking and cyber security researcher typically attempt to model the Internet in controlled and repeatable testbed-based environments [6, 9]. Our experience suggests 95% of DETERLab and Emulab testbed experiments are instantiated with less than 10 nodes [17]. While small experiments are more tractable and predictable, they significantly limit the scope and rigor of networking and cyber security evaluations. Scaling testbed experiments requires significant human effort for two reasons. First, the current testbed interfaces are primarily designed around providing access to computation resources, such as, computers and networking substrates to interconnect the computers. They provide simplistic tools for *hand crafting* topological structures. Thus the burden is on the experimenter for constructing, validating, annotating, and providing embedding guidelines for large scale topologies. These interfaces are not designed for documenting and sharing topologies. Second, the amount of effort and sophistication required to support the experiment increases significantly with the scale and complexity of the experiment. There are several recent efforts to develop specialized experiment management workbenches that will support control and repeatability of the experiment through the design, execution, and analysis lifecycles [8, 4, 1].

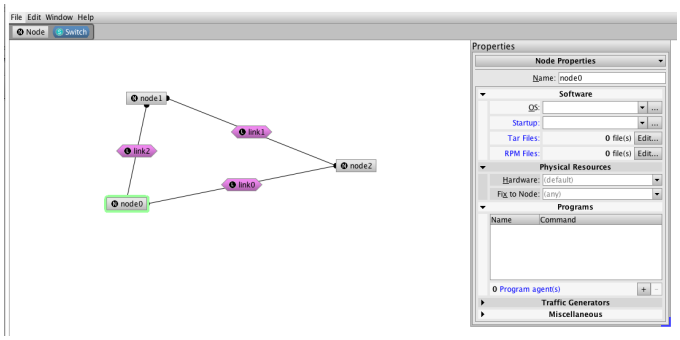
In this paper we discuss the Montage topology manager developed to support topology construction and sharing in a collaborative testbed-based environment. Topology construction is usually the first step in the lifecycle of a testbed-based networking and cyber security experiment. After a topology is constructed, it needs to be embedded on the testbed, instrumented with traffic generation applications, and orchestrated with a workflow. Recent advances in technologies such as federation and virtualization enable embedding large scale topologies on testbeds [7, 18, 23]. The Montage topology manager integrates a wide range of topology management tools into an topology construction and validation workbench. The topological structure of a network is modeled as a graph with nodes and edges.Each node can present a single network entity such as a router, an end host, or a complex network entity such as an autonomous system. The edges represent connections between nodes. In addition to providing tools for generating and graphically manipulating the topology, the Montage topology manager has two other important tools that promote collaboration and reuse.

First, it supports the composition of topologies from *partial models* with horizontal and vertical composition tools. These tools allow leveraging models and experiences accumulated by the scientific community over time. The *horizontal* composition tools create a composite topological model by merging several partial models based on intersecting sets of nodes or edges. For example, as shown in Figure 3, an end-user network model is merged with a core network model to create a single topological model. The *vertical* composition tools create a composite topological model by coupling two or more dependent networks based on dependencies represented as edges. For example, as shown in Figure 4, a SCADA logical controller network model is coupled with a communication network model to create a layered topological model by adding edges between the geographically co-located nodes. We discuss composition tools and cataloging tools in more detail in Section 4.

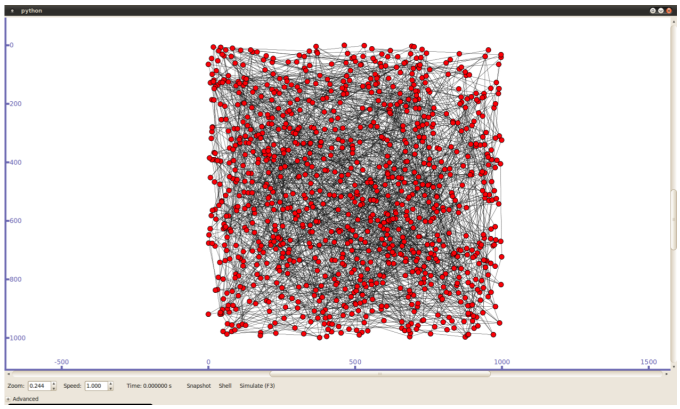
The second tool provides a *validation* mechanism to ensure that the topologies generated or composed meet the *sensibility* and *feasibility* specifications as defined by the ex-



(a) ns-3 Generator



(b) NetBuild GUI



(c) BRITE GUI

Figure 1: The current state of art in graphical topology managers and tools to visualize and manipulate representative Internet topologies

perimenter or the testbed. The sensibility and feasibility specification typically require analyzing the statistical and structural properties of the topology. Additionally, in certain cases, it may also require inspecting the testbed-specific embedding annotations to check if they have been satisfied. For example, the experimenter may specify constraints on particular sets of nodes requiring them to be embedded in specific ways. We discuss both the design and implementation of the various tools in the Montage topology manager in more detail in Section 5.

We believe tools such as the composition and the validation engine are key in promoting reuse and sharing of topology models in experiments. The composition model enables the experimenter to rapidly create variations and derivations of topological structures and share them though the Montage Catalog tool. These derived topological models can then be used to evaluate the systems with corresponding variation on the traffic and workflow definitions. Also traditionally, topological structures are validated using visual inspection methods. The validation engine, based on statistical and structural metrics discussed in Section 5.4, provides a mechanism to validate topological structures based on the specification provided by the experimenter or the testbed [22].

The contribution of this paper is a topology construction manager that enables the construction and documentation of representative topologies for networking and cyber security experimentation in a collaborative testbed-based environment. In Section 5.3 we present two case studies. First, we illustrate how horizontal composition can be used to create an Abeline–inspired Internet2 network topology presented by Li et al. in SIGCOMM 2004 [15]. Second, we illustrate how vertical composition can be used to compose the Italian power-grid network topology studied in several papers including a paper in Nature 2010 [5].

2. TOOLS COMPARISON

This section discusses the Montage topology manager in context with some of the existing topology managers and tools. Specifically, the ns-3 topology editor [19], and the Emulab and DETERLab NetBuild graphical interface [6, 9], and the BRITE topology generator and graphical tools [16]. The tools are compared along the following dimensions; (a) the input and output formats; (b) the capabilities of the topology editor; (c) support for composition of topologies; (d) support for analysis; and (e) visualization support. Each paragraph below introduces the topology tools and discusses the above dimensions in detail.

ns-3 Topology Generator [19], is an open source graphical tool for designing topologies for ns-3. Ns-3 is a wired and wireless simulation and emulation platform for networking experiments. The editor can import and export topology models in an ns-3 specific topology format [14]. It allows construction of both wired and wireless topologies by dragging elements from the palette and dropping them onto the editor canvas as seen in Figure 1(a). It does not have tools to generate regular, random, or hierarchical graphs. Composition is not supported as only a single topology can be imported at a time. There is no support for topology analysis within the current tool. Visualization is limited to a single layout that can be manipulated in the editor.

NetBuild GUI [6, 9], is a thin client web-based graphical tool for designing topologies for the DETERLab and Emulab testbeds. Once a topology is designed it can be directly instantiated on the testbed. The graphical editor allows construction of wired topologies by dragging node and switch

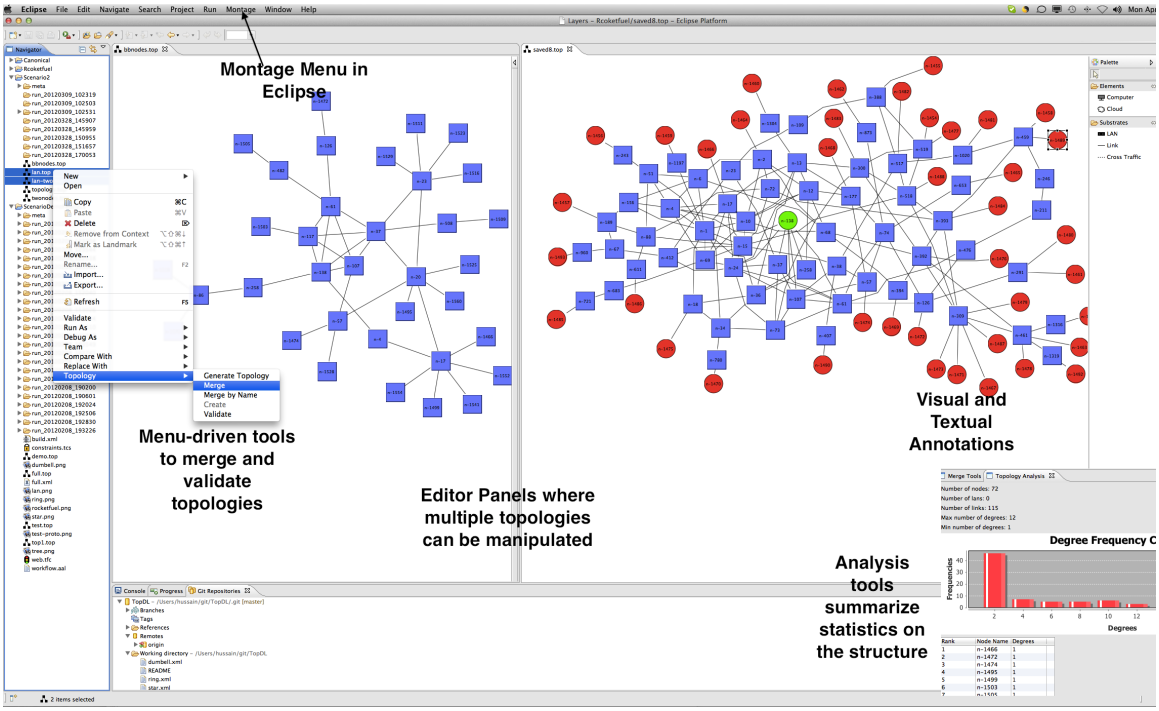


Figure 2: The Montage topology Manager and tools

elements from the palette and dropping them onto the editor canvas as seen in Figure 1(b). Links can be created by clicking on the first and second node in sequence. Once an element is on the canvas, the element can be annotated with different characteristics. For example, the bandwidth and delay values can be specified for each link and the type of operating system and hardware requirements can be specified for each node. When a topology is instantiated, the requested resources are configured as per the hand crafted topological structure. The editor does not support importing existing topologies and manipulating them. The editor also does not have the ability to generate,import, or merge different topological structures. Visualization is limited to a single flat layout that can be manipulated in the editor.

BRITE GUI [20], is an open source framework for generating representative networking topologies. BRITE, stands for Boston university Representative Internet Topology gEnertator, and was originally developed more than a decade ago in 2000-2001. BRITE has extensive support for generating graphs from a wide range of statistical models discussed in detail in the next section. It has the ability to export a topology in two main formats, a BRITE format and the ns-3 simulation format. Additionally there are scripts that allow converting the topology into different testbed and emulation topology languages. The BRITE framework has two types of interfaces; a command line interface and a graphical user interface. The graphical user interface provides a form–based frontend to parameterize different models. It has the ability to hierarchically compose both statistical and empirical models and expand them to a final topological structure. It has the ability to provide simple statistics on the final topological structure. The BRITE framework has limited support for visualizing the resulting topological structures and no support for interactively manipulating the topology. As seen in Figure 1(c), a topology of 1000 nodes is not rendered in a form that represents the underlying structure. Additionally, there is no mechanism to manipulate the nodes. There is no mechanism to textually and visually annotate the nodes and hence severely limits the way the researcher can use this framework.

Figure 2 displays a screenshot of the Montage Topology Manager (abbreviated as MTM) and discusses a couple of tools available within the framework. MTM significantly extends the current state of art in generating and manipulating representative topological structures in a collaborative environment. We discuss them in detail in the following sections.

3. TOPOLOGICAL STRUCTURES

In this section we describe three ways topological structure can be constructed for experimentation. The first is through algorithm based graph generation methods to create regular and random structures. The second class of methods are based on empirical measurements, and lasltly, topologies can be constructed by merging two or more partial models.

3.1 Graph-based Models

There are three main categories of topological structures that are generated from algorithmic models;

Regular graphs are often used in performance, reliability,

and stability studies as their structure makes them tractable and predictable. For example, the dumbbell topology has been widely used to study TCP congestion control [13, 2]. Other examples include ring, trees, stars, lines, meshes, and fully connected graphs. In Section 5, we demonstrate how such topologies can be generated with the Montage topology manager.

Flat Random graphs have been widely used to model internetworks both at the router level and at the autonomous system (AS) level. The router level topologies capture the hop by hop path on the network, where as the autonomous system level topologies captures a high level abstraction, where only hops between different domains, such a regional or administration domains, is modeled. A wide range of statistical models have been used to capture network structure. Examples include, the Waxman model [24], ER graphs [10], power-law models [11], and preferential connectivity models such as Barabasi–Albert models [3].

All these random graph generation methods are variations of the same basic method. A set of node vertices are distributed in a plane, and an edge is added between each pair of vertices with some probability as defined by the model. In Section 5, we discuss how the BRITE [16, 26] topology generators are extended to support generating various random graphs that can be visualized and manipulated within the Montage topology manager.

Hierarchical graphs reproduce transit-stub, hub-spoke, or tier-based structural properties of the Internet. For example, the Transit-stub and the Tiers topology generators produce topologies that have well defined multi-level representative hierarchies [27, 12, 16]. There are primarily two types of hierarchies. *Top-down* hierarchical topologies that first define the AS-level topological structure and then each node in the AS-level graph is resolved to a corresponding router-level topology. *Bottom-up* hierarchical topologies that first define the router-level topology and then the routers are segregated into disjoint groups of nodes. Each group uniquely represents an autonomous system. Both types of hierarchies can be generated using the BRITE topology generator that is integrated with the Montage topology manager [16].

3.2 Empirical Models

While graph-based models parsimoniously capture the topological structure in few parameters, experimenters sometimes need to experiment with empirical models of topologies. Empirical or real–world topologies are typically identified and measured at the AS-level, POP-level, router-level, or edge network topologies. For example, each node in an AS-level topology represents one autonomous system (AS). The Montage Catalog has several Rocketfuel topologies that can be imported, visualized, manipulated, or merged with other models to create representative topologies for experimentation [21].

4. TOPOLOGY COMPOSITION

The Internet is a large-scale, highly engineered, and highly complex system. It is characterized by an enormous degree of heterogeneity and undergoes continuos and significant changes over time. There have been several papers recently that critically examine Internet topology models such as the scale-free node degree distributions [15, 25].

We have developed a community-based collaborative environment in Montage that supports composing and validating topology models. A topology model can be constructed from a series of *partial* models, each representing part of the topological network structure that needs to be modeled for the experiment. Each partial model may not meet the overall specification of the topological structure when considered in isolation, however, when combined, they create representative topological structures that can be used for systematic evaluations of networking and cyber security systems. Hence this allows experimenters to iteratively build on their own work as well as the work of others by augmenting and extending such models.

We propose two types of composition methodologies.

Horizontal composition where a composite topological model can be created by combining several individual partial models. These partial models can be developed using graphical-based methods discussed in Section 3.1 or through empirical measurements as discussed in Section 3.2. For example, several measurement-based edge networks and enterprise network models can be merged with a mesh-based core network model to create a single topological structure that can be used to evaluate cyber security incidences in edge networks. Another example, is the Abilene-based network depicted in Figure 3, where the each vertex represents a router, each edge represents a physical connection, and end-user networks are shown as clouds [15]. Starting from this abstract topology model, we want facilitate combining models to create representative topological structural details for each of the clouds.

Vertical composition where a composite topological model is created by coupling two or more complete or partial models of dependent networks. The dependencies between the networks may be, for example, due to shared communication channels, co-location, or other shared resources . These partial models can be developed using graphical-based methods discussed in Section 3.1 or through empirical measurements as discussed in Section 3.2. For example, Figure 4 models a power blackout that affected almost all of Italy for a 12 hour period in 2003 [5]. The model is composed of two complete network models, the Italian high-voltage electrical transmission network (HVIET) and the high

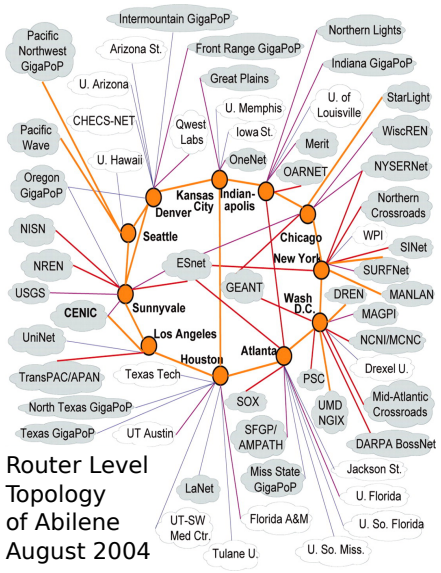


Figure 3: A router-level topology of the Abilene-network [15]. The topological structure of the edge networks can be defined by *horizontal* composition methods.

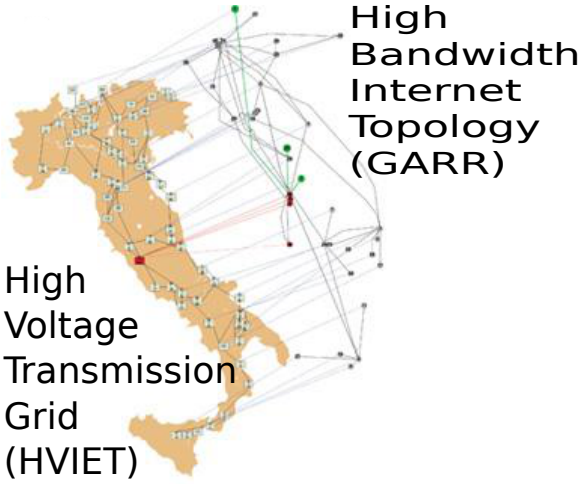


Figure 4: Modeling a blackout in Italy in 2003. The network topology is composed of two interdependent networks that are coupled together using *vertical* composition methods [5]. Reproduced with permission from Nature Publishing Group

bandwidth blackbone of the communication network (GARR) that links Italian universities and research institutions. The two models are geographically coupled by adding edges to represent interconnectedness and dependency relationships.

In Section 5, we discuss how such composite topological models can be created using partial or complete network models.

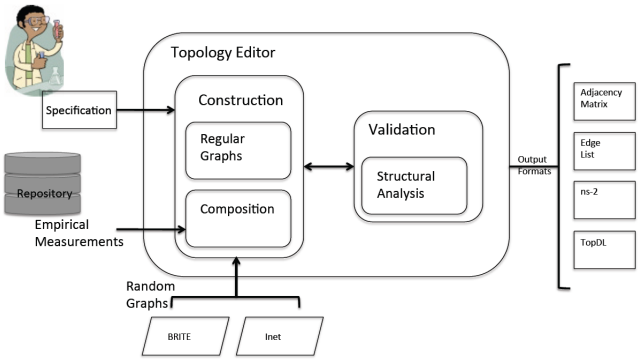


Figure 5: The Architecture of the Montage Topology Manager

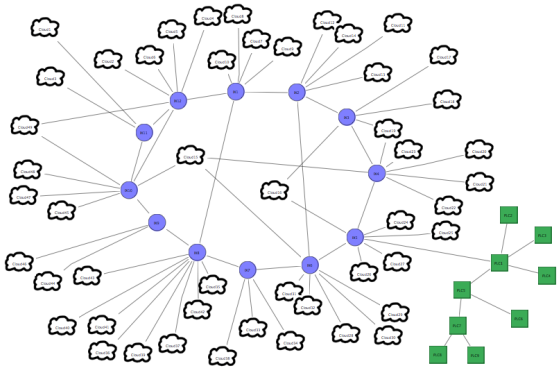


Figure 6: Modeling the Abilene-inspired topology using horizontal composition in the Montage topology manager

5. DESIGN AND IMPLEMENTATION

In this section we first discuss our high-level goals for developing the Montage topology manager. We then discuss the architecture and the implementation of the tools. We finally illustrate how the tools can be used to realize composite topologies.

5.1 Design Wish List

There are several topology generation tools available in the networking community. Our goal is to integrate and develop a wide range of topology management tools to create a collaborative environment to construct large scale topologies for testbed-based experiments. Specifically, we wanted to develop topology construction tools that have the following properties:

- **Interactiveness** Provide an ability to construct non-trivial topological models. In addition to hand crafting topological models, we wanted to be able create a wide range of variations and derivations of topological models through graphical user interfaces. The interac-

tions for the topology construction should be *natural* for the experimenter and maximize the *productivity* of the experimenter.

- **Flexibility** Provide interfaces that support a wide range of experimenter abilities, from a naive student to an astute researcher.

- **Extensibility** Provides a wide range of expressive topology generation models and provides an easy interface to add new models as and when required. The graph based topological structure should be easily extensible. For example, the topological models should be allowed to grow in scale and complexity through an iterative and repeatable process.

- **Interoperability** Provide tools that are interoperable and can be used on a wide range of operating systems. The topological models should be expressed in a wide range of formats so that they can be instantiated on different experimentation environments.

- **Robustness** Provide mechanisms to validate topology models with analysis of the resulting topological structure through interactive and automated mechanisms. The validation mechanism should allow testing for sensibility and feasibility of the topological structure and the variations and derivations.

In the sections below to discuss the architecture of the topology manager, some of the individual tools and methods.

5.2 Architecture

As seen in Figure 2, Montage leverages the extensible Eclipse-based plugin framework for constructing, deploying, and managing experiments. The main components in the Montage topology manager are: tools for importing, exporting, and cataloging topologies, graph-based topological models and extensions to existing topology generators, methods and algorithms for horizontal and vertical composition, validation mechanisms for topologies. We discuss each of these in detail below.

The specific details regarding how the topology is generated are dependent on the generation models discussed in Section 3 , however, we can broadly define the topology construction workflow as a four phase process:

1. Construct a topology either by hand-crafting it, using the available topology generators, or through composition of partial models.

2. Document and annotate the topology with visual and experimentation specific information. For example, large scale topologies may require additional information so that they can be correctly partitioned for emulation or testbed-based experimentation.

3. Validate the resulting topological model based on sensibility and feasibility specifications and constraints.

4. Export the topology in a specific format in the catalog or of instantiation on the testbed.

This workflow conceptually reflects what occurs when topological structures are created for experimentation. A specific experimentation workflow may have a different ordering of the above steps or may perform several of the steps multiple times to iterate and generate the final topological structure. In the following sections, we discuss these steps in the context of particular methods and mechanisms provided in the Montage topology manager.

5.2.1 Topology Editor

The topology editor is the primarily interface between the experimenter and the topology generation tools. The editor provides a range of tools that allow the experimenter to construct topological graphs models from scratch using boxes and links or create derivations of the models through importing and manipulations.

The editor provides a wide range of expressive topology generation models for both regular and random graphs. The topology editor tools and menus can be extended easily to add new generation models as required. Additionally, Eclipse-based tools and mechanisms allow maintaining local histories of the topology files such that changes can be easily reverted if required. The editor is also tightly integrated with a range of version control systems.

Topology composition is supported in two primary modes in the editor. First, the experimenter can choose two or more topology files to merge them interactively in the editor. The editor also allows the experimenter to define merging constraints based on node–name mappings. The constraints can be applied on one or more partial topology structures to merge them to create a composite topological structure.

5.2.2 Import and Export

The topology editor can import topologies that are generated by other topology generators in four main formats; as an adjacency matrix, as an edge list, in ns-2 format, and the TopDL format. The TopDL is topology description language developed for federation services at DETERLab and allow topologies to span across several different physical testbeds [23]. Once the topology has been developed it can be exported in all the above listed formats. The first two formats encode only connectivity information about the network graph. The last two formats can encode visual and textual annotations in addition to the connectivity information.

Visual annotations allow the elements to be encoded with color and shape attributes. A vertex or an edge can be annotated with a particular size, shape, edge width, or color.

Additionally, the Montage topology manager has a diverse set of topology models in a git-based Catalog.Figure 8 shows

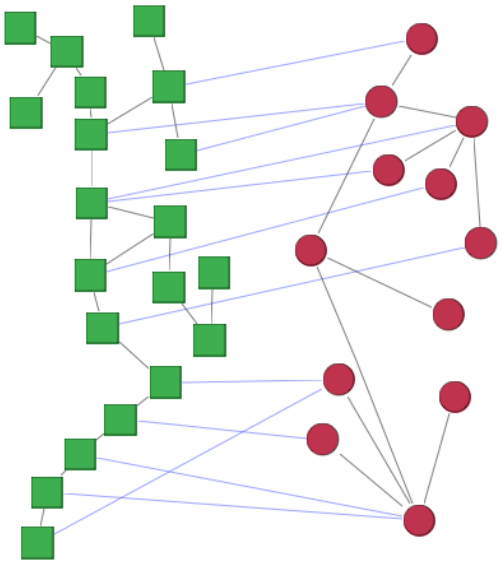


Figure 7: Modeling the Italy blackout topology using vertical composition in the Montage topology Manager

a screen shot of the the Montage topology Catalog. Each topology entry has three main pieces of information: an icon that captures the topological layout, the descriptor annotations that summarize the statistical and analytical properties of the topology, and the meta-data annotations that include authorship and usage history information. The descriptor annotations include information such as the number of nodes, the number of links, and the minimum number of hops between any two nodes in the topology. The meta-data annotations include information such as the name of the contributor, the date, the number of times the topology was downloaded, and possibly a ranking of the contributor. The Montage topology manager has tools and scripts that allow extracting and defining the descriptor and meta-data information. The catalog can be searched used on the annotations and the meta-data.

5.3 Composition

The composition mechanism can create composite topologies from *partial models* with horizontal and vertical composition. The *horizontal* composition tools creates a composite topological model by merging several partial models based on intersecting sets of nodes or edges. We consider two different types of constraints: syntactic and semantic constraints. (a) *Syntactical* constraints support merging based on string matching. For example, node named IXC2011 and node named POP2011 are identical in the two topology fragments. The composite topology model will merge the two fragments by overlapping that node and adjusting the edges. The annotations are preserved from the primary fragment in case of conflicts. (b) *Semantic* constraints support merge based on inherent model properties. For example, the partial model named AS100, is a core autonomous system implying its nodes should not be leaf nodes. This type of merging considers annotations applied to the whole fragment as compared to individual elements. Currently, the topology manager supports merging of fragments based on syntactic constraints. We are currently exploring the various semantic constraints that are representative of networking and cyber security topology models and will be implementing them as future work. Figure 6 illustrates how horizontal composition tools can be used in the Montage topology manager to create an Abilene-inspired network topology presented by Li et al. in SIGCOMM 2004 [15].

The *vertical* composition tools create a composite topological model by coupling two or more networks based on dependencies represented as edges. The dependencies between the networks may be due to, for example, share communication channels, co-location, or other shared resources. Currently, the vertical composition tool supports two modes: (a) *manual* composition, where edges are manually inserted between the two partial models, or (b) *probabilistic* composition, where edges are inserted between a particular node in the primary partial model and all nodes in the other partial models based on a specified value p. In the future, we will explore other types of vertical composition modes. Figure 7 illustrates how vertical composition tools can be used in the Montage topology manager to couple a SCADA logical controller network model depicted with green squares, with a a communication network model, depicted with red circles. It creates a layered topological model by adding edges between the geographically co-located nodes. The composite model is a representation of the Italian power-grid network topology studied in several papers including a paper in Nature 2010 [5].

5.4 Validation

The *validation* mechanism can be used to ensure that the topologies generated and composed meet the *sensibility* and *feasibility* specifications as defined by the experimenter or the testbed. This step is recommended and important in the workflow of topology construction for several reasons. (a) Manipulations, such as composition, variations, and derivation of the topological structure may alter statistical and structural properties that may not be *representative* or *realistic* for typical networks. (b) Topological structure generated may not directly work on the testbed due to the limited availability or configurations of the physical resources. (c) Further, as the scale and complexity of the topology increases, the properties may no longer be visually apparent or tractable.

The sensibility and feasibility specification typically require analyzing the statistical and structural properties of the topology. The statistical properties include cardinality measures, node degree rank and frequency measures. The structural properties report the path length distribution in the

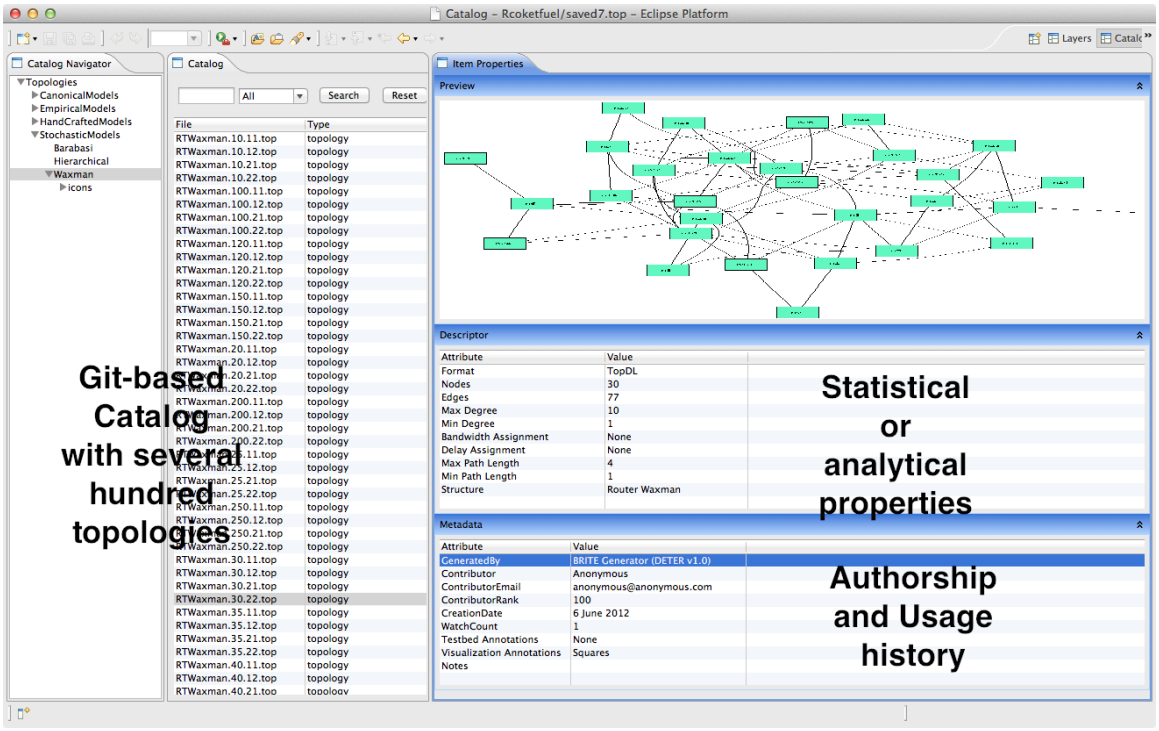


Figure 8: The Catalog tool allows browsing and searching through a large collection of topologies that can be directly imported into the editor

topology.

We consider two different types of cardinalities: absolute and relative. (a) *absolute* cardinality measures the number of elements in a set. For example, the total number of nodes or vertices in a topology or the total number of leaf nodes in a topology. (b) *relative* cardinality measures the ratio of elements in one set as compared to another set. For example, the ratio of high bandwidth links to low bandwidth links. Additionally, the topology manager has tools to report the node degree rank, the node degree frequency, and the path length distribution. Currently, the validation specification allow defining and testing of absolute cardinalities.

In certain cases, the validation tool may also require inspecting the testbed-specific embedding annotations to check if they have been satisfied. The experimenter may specify annotations on sets of nodes requiring them to be embedded in specific ways. For example, the number and bandwidth of available network interfaces on a physical node in a testbed limit the node degree that can be supported in a topology. Typically, such constraints do not exist for simulation-based topologies.

6. CONCLUSION

Networking and cyber security research requires collaborative environments that allow developing non-trivial topological structures for rigorous scientific evaluations. In this paper we presented the Montage topology management tools that provide extensive support for interactive and intuitive topology generation, composition, and validation. These tools enable the experimenter to develop large scale and complex topological structures and promote documenting, sharing and reuse of topologies. We hope that new research directions will guide the development of the topology tool suite and the future releases will incorporate topology generation tools and workflows from other researcher in the networking and cyber security community.

7. REFERENCES

[1] J. Albrecht, C. Tuttle, C. Snoeren, and A. Vahdat. Planetlab application management using plush. *SIGOPS Operating Systems Review*, 40(1):33–40, 2006.

[2] K. Anagnostakis, M. Greenwald, and R. Ryger. On the sensitivity of network simulation to topology. In *MASCOTS*, 2002.

[3] A. Barabasi and R. Albert. Emergence of scaling random netoworks. *Science*, 286(5439):509–512, October 1999.

[4] T. Benzel. The science of cyber security experimentation: the deter project. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, pages 137–148, New York, NY, USA, 2011.

[5] S. Buldyrev, R. Parshani, G. Paul, E. Stanley, and S. Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464:0125–1028,

2010.

[6] DETER. DETERLab: A Testbed for Cyber Security Experimentation. http://www.deterlab.net, 2004.

[7] J. Duerig, R. Ricci, L. Stoller, M. Strum, G. Wong, C. Carpenter, Z. Fei, J. Griffioen, H. Nasir, J. Reed, and X. Wu. Getting started with geni: a user tutorial. *SIGCOMM Comput. Commun. Rev.*, 42(1):72–77, 2012.

[8] E. Eide, L. Stoller, and J. Lepreau. An experimentation workbench for replayable networking research. In *Proceedings of the 4th USENIX conference on Networked systems design and implementation*, NSDI'07, pages 16–16, Berkeley, CA, USA, 2007. USENIX Association.

[9] Emulab. Emulab.net: A Network Emulation Testbed. http://www.emulab.net, 1997.

[10] P. Erdős and A. Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[11] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *SIGCOMM Comput. Commun. Rev.*, 29(4):251–262, Aug. 1999.

[12] I. Globecom, editor. *A Better Model of Generating Test Networks*, November 1996.

[13] O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmetz. On realistic network topologies for simulation. In *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, MoMeTools '03, pages 28–32, New York, NY, USA, 2003. ACM.

[14] T. R. Henderson, S. Roy, S. Floyd, and G. F. Riley. ns-3 project goals. In *Proceeding from the 2006 workshop on ns-2: the IP network simulator*, WNS2 '06, New York, NY, USA, 2006.

[15] L. Li, D. Alderson, W. Willinger, and J. Doyle. A first-principles approach to understanding the internet's router-level topology. *SIGCOMM Computer Communication Review*, 34(4):3–14, 2004.

[16] A. Medina, A. Lakina, I. Matta, and J. Byers. Brite: An approach to universal topology generation. In *Proceedings of the International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS)*, Cincinnati, Ohio, August 2001.

[17] J. Mirkovic, A. Hussain, and H. Shi. Deterlab usage statistics: Semi-annual dhs review. http://www.deter-project.org, 2011.

[18] K.-H. Nam, M.-K. Shin, H.-J. Kim, and S. Jeong. Federating future internet testbeds: an adaptor-based approach. In *Proceedings of the 6th International Conference on Future Internet Technologies*, CFI '11, pages 50–52, New York, NY, USA, 2011.

[19] ns-3 Generator. ns-3 topology generator. http://www.nsnam.org/wiki/index.php/Ns3Generator, November 2010.

[20] G. Riley and J. Pelkey. ns-3 and BRITE intergration. http://www.nsnam.org/wiki/index.php, March 2011.

[21] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring isp topologies with rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, 2004.

[22] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. Network topology generators: degree-based vs. structural. *SIGCOMM Comput. Commun. Rev.*, 32(4):147–159, Aug. 2002.

[23] F. Theodore. Fedd: Deter federation daemon. http://fedd.deterlab.net/.

[24] B. Waxman. Routing on multipoint connections. *IEEE Journal on Selection Areas on Communications,*, 6(9):1617–1622, December 1988.

[25] W. Willinger, R. Govindan, S. Jamin, V. Paxson, and S. Shenker. Scaling phenomena in the internet: Critically examining criticality. *PNAS*, 99:2573–2580, 2002.

[26] J. Winick and S. Jamin. Inet-3.0: Internet topology generator. Technical Report CSE-TR-456-02, University of Michigan, 2002.

[27] E. Zegura, K. Calvert, and S. Bhattacharjee. How to model an internetwork. In *Proceeding of IEEE Infocom*, San Francisco, CA, 1996.