BRITE: An Approach to Universal Topology Generation *†

Alberto Medina, Anukool Lakhina, Ibrahim Matta, John Byers
{amedina, anukool, matta, byers}@cs.bu.edu
Computer Science Department
Boston University

Abstract

Effective engineering of the Internet is predicated upon a detailed understanding of issues such as the large-scale structure of its underlying physical topology, the manner in which it evolves over time, and the way in which its constituent components contribute to its overall function. Unfortunately, developing a deep understanding of these issues has proven to be a challenging task, since it in turn involves solving difficult problems such as mapping the actual topology, characterizing it, and developing models that capture its emergent behavior. Consequently, even though there are a number of topology models, it is an open question as to how *representative* the generated topologies they generate are of the actual Internet. Our goal is to produce a topology generation framework which improves the state of the art and is based on the design principles of representativeness, inclusiveness, and interoperability. *Representativeness* leads to synthetic topologies that accurately reflect many aspects of the actual Internet topology (e.g. hierarchical structure, node degree distribution, etc.). *Inclusiveness* combines the strengths of as many generation models as possible in a single generation tool. *Interoperability* provides interfaces to widely-used simulation applications such as *ns* and *SSF* and visualization tools like *otter*. We call such a tool a *universal topology generator*.

Keywords: topology generation, graph models, network topology, growth models, annotated topologies, simulation environments.

1 Introduction

During the design phase of an Internet-based technology, extensive simulations are usually performed to assess its feasibility, in terms of efficiency and performance. In general, Internet studies and simulations assume certain topological properties or use synthetically generated topologies. If such studies are to give accurate guidance as to Internet–wide behavior of the protocols and algorithms being studied, the chosen topologies must exhibit fundamental properties or invariants empirically found in the actual extant structure of the Internet. Otherwise, correct conclusions cannot be drawn.

*BRITE is funded in part by NSF grants CAREER ANI-0096045, CAREER ANI-0093296 and ANI-9986397.

†BRITE can be downloaded from http://www.cs.bu.edu/brite.

Unfortunately, achieving a deep understanding of the topology of the Internet has proven to be a very challenging task since it involves solving difficult problems such as mapping the actual topology, characterizing it, and developing generation models that capture its fundamental properties. In addition, the topology of the Internet is a target that is constantly evolving, and it is controlled by a set of autonomous authorities that are not often willing to exchange low-level connectivity information [14].

There are several synthetic topology generators available to the networking research community [18, 5, 3, 11, 9, 1]. Many of them differ significantly with respect to the characteristics of the topologies they generate. Furthermore, existing topology generators fail to produce complete representations of the Internet since they focus primarily on network connectivity or structural characteristics only, and do not attempt to model other properties of the network such as link bandwidths and delays.

Our objective caters to two groups of researchers. On the one hand, there are researchers investigating Internet protocols and algorithms who need topology generation tools to obtain good synthetic topologies for their simulations. On the other hand, there are researchers (like us) investigating the challenges associated with generating accurate synthetic topologies. For both groups it would be very useful to have topology generation tools that allow them to easily evaluate the pros and cons of new generation models.

An attractive scenario is to have a topology generation tool that provides a researcher with a wide variety of generation models, as well as the ability to easily extend such a set by combining existing models or adding new ones. In this paper we discuss the design and implementation of BRITE, the Boston university Representative Internet Topology gEnerator, which is a tool designed to realize this scenario.

This paper is organized as follows. In Section 2 we discuss the challenges that must be tackled to generate accurate synthetic topologies, what the characteristics of an ideal generation tool are and the approach we take to achieve these. In Section 3 we describe the general design of BRITE and some implementation details. Section 4 presents some results obtained using BRITE as the generation tool. Finally, Section 5 presents concluding remarks.

2 Wish List for a Topology Generator

An ideal topology generator should enable the use and development of generation models that produce accurate representations of Internet topologies. The following is a list of desirable characteristics for such topology generator.

1. *Representativeness.* Produces accurate synthetic topologies. Accuracy should be reflected in as many aspects of the actual Internet topology as possible (e.g. hierarchical structure, node degree distribution, etc.).
2. *Inclusiveness*. Combines the strengths of as many generation models as possible in a single generation tool.
3. *Flexibility*. Generates topologies over a wide range of sizes. Restrictions such as minimum and maximum number of nodes should be reasonably avoided.
4. *Efficiency*. Generates large topologies (e.g. number of nodes $> 100,000$) with reasonable CPU and memory consumption.
5. *Extensibility*. Provides mechanisms that allow the user to easily extend its capabilities by adding new generation models.
6. *User–friendliness*. Follows the usage principles of standard user interfaces. The user should learn the mechanics of the generation tool only once. For each generation model incorporated in the tool, she should only need to learn the functionality associated with the new model.
7. *Interoperability.* Provides interfaces to main simulation and visualization applications. It should be possible to generate topologies that can be processed by widely used simulators such as *ns* [13] and *SSF* [15].
8. *Robustness.* Does not sacrifice robustness in the name of efficiency and includes extensive error detection capabilities.

In Section 2.1 we describe the main topology generators and generation models available, in Section 2.2 we discuss some challenges that must be overcome to develop a universal generation tool satisfying our wish list, and in Section 2.3 we argue about a possible approach to tackling those challenges.

2.1 Available Topology Generators

Available topology models/generators can be broadly classified into two categories [10]. The first category include *ad-hoc models* mostly built based on educated guesses. This includes Waxman models [18], GT-ITM [3] and Tiers [5]. The second category includes models built based on measurements, e.g. power-law distribution of node outdegrees. These *measurement-based models* can be further classified as those that are *causality-oblivious*: models that reproduce the power-law distribution (e.g. Inet [9], PLRG [1]), and those that are *causality-aware*: models that try to model "possible" fundamental/physical causes such as the network growing incrementally, and new nodes preferring to connect to higher-degree nodes (e.g. Barabási-Albert model [2], BRITE 1.0 [11]). With respect to certain properties, such as those studied in [6], measurement-based models seem to be relatively more accurate in generating Internet-like topologies [11].

Briefly, Waxman models are concerned only with general random networks, where the probability of connectivity of node-pairs is based on distance. GT-ITM [3] and Tiers [5] are primarily concerned with the hierarchical properties of the Internet related to how it is organized as levels of service-providers. Barabási-Albert models, BRITE 1.0, Inet, and PLRG are concerned with resemblance to Internet topologies in terms of connectivity properties (e.g. outdegree distribution). Models generating regular topologies (e.g. tree, mesh) have also been used for specific and restricted scenarios.

However, a unified model that considers both hierarchical properties, degree distributions and connectivity properties, and incorporates causal models has not yet been developed.

2.2 Universal Generation Tool: Challenges

Having so many independent generation models and topology generators is disadvantageous in many respects. A researcher in need of synthetic topologies to investigate the correctness and performance of protocols and algorithms is forced to learn the nuances of many of these models/generators. Consequently she may be forced to use the most popular one, the one supported in the simulation environment used, or the easiest one. Analogously, for a researcher investigating the challenges of topology generation and looking for better and more powerful generation models, having so many generators available makes comparative analyses of different models significantly more difficult.

These difficulties are in addition to the inherently hard problems encountered when developing models that accurately capture fundamental properties of the Internet topology. Such a model is usually developed based on measured topological information that is not completely accurate. This lack of accuracy is mainly due to the fact that mapping the Internet topology is a very challenging task [8, 17]. At the Autonomous System (AS) level, available information is richer because it can be obtained or inferred from BGP tables [12, 7]. In contrast, accurate router-level topological information is hard to obtain and until now inferring router-level connectivity has been done by using traceroute or traceroute-like probing mechanisms [8, 4]. Identifying the actual fundamental properties of topologies at the router-level is still an open research question [19]. Most Internet topology studies have approached topology modeling relying only on physical connectivity. However, routing in the Internet is determined by a policy–based routing protocol (BGP) and consequently physical connectivity does not always imply reachability. Customer–provider and peering relationships play a deciding role in determining whether or not traffic can flow between connected nodes. Hence the connectivity of a topology alone does not completely characterize the structural properties of the corresponding routing topology [7]. Even if we knew the actual relationships between ASs, such relationships are continually changing. Therefore, in order to generate accurate representative topologies, the invariants of such relationships across time and size must be discovered.

In short, research in topology generation is in its infancy. New models will be developed as research will expose new and more powerful mechanisms to accurately characterize the topology of the Internet. Our challenges can be concisely put into two issues:

1. How do we develop an adapting and evolving generation tool that constitutes an interface between general Internet research and pure topology generation research? Through this interface, representative topologies developed by the topology generation research community, can be made readily available to the Internet research community at large.
2. How do we design a tool that also achieves the goal of facilitating pure topology generation research? A researcher that devises a generation model should be able to test it readily without having to develop a topology generator from scratch.

2.3 Approach to Universal Generation

We address the challenges described above by establishing a differentiation between **model-oriented** topology generators, and a **universal** topology generator. *Model-oriented* generators are designed and implemented with a specific set of models in mind. All the generators described above fall in this category. In contrast, a *universal generator* should not be tied to a specific set of models. Instead, this generator should be *extensible*, allowing the addition of new models in an easy way. This characteristic makes a universal topology generator *flexible* and *adaptable*, generating representative topologies to be used in different simulation scenarios.

3 BRITE Design and Implementation

BRITE was designed to be a flexible topology generator, not restricted to any particular way of generating topologies. As such, it supports multiple generation models. In this section we describe how this design goal was approached and how BRITE is implemented. Figure 1 depicts a schematic view of the structure of BRITE as it is being used at Boston University. The different components are labeled (1)–(4).

BRITE reads the generation parameters from a configuration file (1) that can be either hand written by the user or automatically generated by BRITE's GUI (see Section 3.8). BRITE provides the capability of importing topologies (2) generated by other topology generators (GT-ITM [3], Inet [9], Tiers [5], BRITE 1.0 [11]) or topological data gathered directly from the Internet (NLANR [12], Skitter [4]). Note that we include BRITE

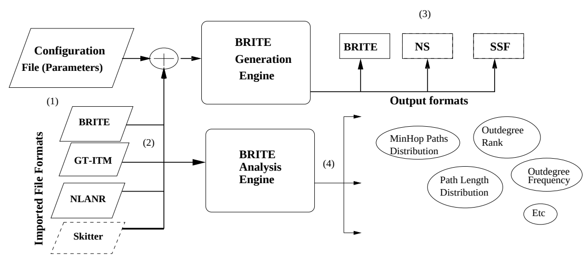

Figure 1: Schematic structure of BRITE

in the imported file formats, because it is possible to generate topologies using BRITE and then reuse them to generate other topologies by combining them with BRITE models or other imported formats. In the current distribution, BRITE produces a topology in its own file format (3), and is capable of producing topologies that can be used directly by the Network Simulator (NS [13]) and the Scalable Simulation Framework (SSF [15]) simulator.

We also developed the BRITE Analysis Engine or BRIANA (4). BRIANA provides a set of analysis routines that may be applied to any topology which can be imported into BRITE. If we need to analyze a new topology, we just add a parsing procedure to BRITE for that new format, and once that is done, BRIANA's analysis routines can be used on the new topology.

3.1 BRITE Architecture

In BRITE, a topology is represented by a class Topology (see Figure 2). This class contains a Model (1) and a Graph (2) as data members, and among others, a set of exporting methods and function members (3). Currently, BRITE topologies can be exported to SSF's DML [15] format as well as visualized using CAIDA's otter tool [4].

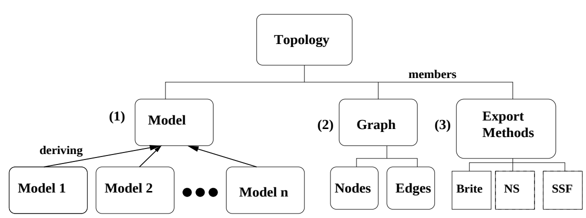

Figure 2: A Topology as seen by BRITE

The Model class is an abstract base class from which multiple specific generation models are derived. Each specific topology generated by BRITE can use a single instance of one of the available generation models if the generated topology is flat, or more than one instance if the topology is a combined hierarchical topology (Section 3.2). The Graph data member (2) is a Graph class with the minimal functionality required by the generation models. Should more capabilities from the Graph component be required, this class may be extended or replaced with minimum effects on the remaining code.

3.2 How BRITE Works

The specific details regarding how a topology is generated depend on the generation model being used. We can broadly think of the generation process as a four-step process:

1. Placing the nodes in the plane
2. Interconnecting the nodes
3. Assigning attributes to topological components (delay and bandwidth for links, AS id for router nodes, etc.)
4. Outputting the topology to a specific format.

This of course is not a clear-cut division that will fit every generation model but conceptually reflects what happens when a topology is being generated. Also, several models may share specific steps during the generation process, while other models differ significantly on the individual steps. In the next section we will discuss these steps in the context of particular models provided in the current distribution of BRITE.

3.3 Models

BRITE's architecture is centered around the Model class. As we can see in Figure 3, the current distribution of BRITE contains eight different generation models. Some of them are very similar and share implementation code, and others are completely different and share no functionality. Every model has a *Generate* method which returns a graph containing the generated topology. In the next subsections, we describe each of the available models.

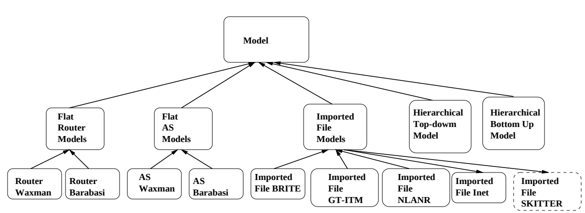

Figure 3: Model class and its deriving classes

3.4 Flat Router-level Models

BRITE contains a class *RouterModel* derived from the Model class. The idea of having such a class is to separate models that generate router-level topologies, from models specific to other environments (AS, LANs, etc.). Keep in mind that the intrinsic details of any of the provided models do not represent a limitation with respect to the flexibility offered by BRITE. If none of the available individual models satisfy the requirements of a specific simulation environment, one could combine existing models or create a completely new model and integrate it into BRITE.

The router-level models currently provided with BRITE are called *RouterWaxman* (Section 3.4.3) and *RouterBarabasiAlbert* (Section 3.4.4). These models share certain functionality. Specifically, both models place the nodes in the same way into the plane, and once the topology has been fully generated, they both assign bandwidth attributes to the links in the same way. They mainly differ in the network growth model and the node interconnection method used.

3.4.1 Placing the Nodes

BRITE separates the placement of the nodes from the process of growing the topology and interconnecting the nodes. By placing a node we mean selecting a location in the plane for it and creating and initializing the data structures for the node in the graph. This phase does not mean that the nodes already belong to the topology because the specific joining time of a node to the topology will depend on the growth model employed.

The class *RouterModel* provides a method called *PlaceNodes* that places the nodes on the plane in one of two ways: random or heavy tailed. The motivation behind providing heavy-tailed distributions is explained in [10]. When node placement is random, each node is placed in a randomly selected location of the plane. When the placement is heavy-tailed, BRITE divides the plane into squares. Each of these squares is assigned a number of nodes drawn from a heavy-tailed distribution. Once that value is assigned, then that many nodes are placed randomly in the square. Again, this placement mechanism can be modified or overridden by particular models.

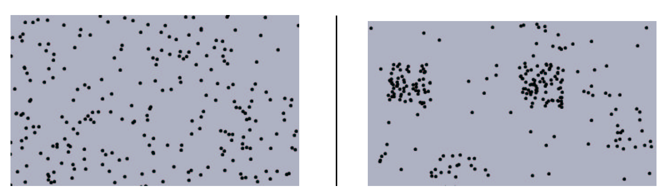

Figure 4: Snapshot of random node placement (left) and heavy-tailed node placement (right)

Figure 4 shows the difference between random and heavy-tailed node placement. The clustering provided by heavy-tailed placement can be used for specific generation models [11].

3.4.2 Assigning Bandwidths

Once the topology has been completely generated, both router-level models invoke the *AssignBandwidth* method of the *RouterModel* class. New router-level models can override this method or choose not to call it at all.

BRITE assigns bandwidths to links according to one of four possible distributions. The user specifies in the configuration file passed to BRITE, which distribution is going to be used *($BWdist$)*, along with a minimum *($BWmin$)* and maximum *($BWmax$)* values for possible bandwidths that can be assigned. BRITE assigns a bandwidth to each link that is either:

1. **Constant**: the value specified by $BWmin$ (equal for all links in the topology).
2. **Uniform**: a value uniformly distributed between $BWmin$ and $BWmax$.
3. **Exponential**: a value exponentially distributed with mean $BWmin$.
4. **Heavy-tailed**: a value heavy-tailed distributed (Pareto with user-specified shape) with minimum value $BWmin$ and maximum value equal to $BWmax$.

Note that the user's choice of $BWdist$, $BWmin$ and $BWmax$ drives BRITE's bandwidth assignment. BRITE treats bandwidth values as unitless. Users interpret the meaning of bandwidth units according to their needs.

3.4.3 Router Waxman

RouterWaxman basically refers to a generation model for a random topology using Waxman's probability model [18] for interconnecting the nodes of the topology, which is given by:

$$P(u,v) = \alpha\, e^{-d/(\beta L)}$$

where $0 < \alpha, \beta \leq 1$, d is the Euclidean distance from node u to node v, and L is the maximum distance between any two nodes.

3.4.4 Router BarabasiAlbert

BRITE provides a *RouterBarabasiAlbert* model, which implements a model proposed by Barabási and Albert [2]. This model suggests two possible causes for the emergence of a power law in the frequency of outdegrees in network topologies: incremental growth and preferential connectivity. *Incremental growth* refers to growing networks that are formed by the continual addition of new nodes, and thus the gradual increase in the size of the network. *Preferential connectivity* refers to the tendency of a new node to connect to existing nodes that are highly connected or popular.

RouterBarabasiAlbert interconnects the nodes incrementally. When a node i joins the network, the probability that it connects to a node j already belonging to the network is given by:

$$P(i,j) = \frac{d_j}{\sum_{k \in V} d_k}$$

where d_j is the degree of the target node, V is the set of nodes that have joined the network and $\sum_{k \in V} d_k$ is the sum of outdegrees of all nodes that previously joined the network.

3.5 Flat AS-level Models

For the current distribution of BRITE, the provided AS-level models are very similar to the models provided for generating router-level topologies. The main difference between these router-level and AS-level models is the fact that AS models place AS nodes in the plane and these nodes have the capability of containing associated (router-level) topologies. Note that this does not mean that there are no AS-level and router-level models that differ substantially from each other. The idea of separating router-level from AS-level from the beginning is to allow for the flexibility of developing independent models for each scenario. The two AS-level models provided with the initial distribution of BRITE are **ASWaxman** and **ASBarabasiAlbert**.

3.6 Hierarchical Topologies

Generation models such as *Transit-stub* [3] and *Tiers* [5], are centered around reproducing structural properties of the Internet. In particular, Transit-stub has a well-defined hierarchy representing transit and stub autonomous systems in the Internet. Tiers is based on a three-level hierarchy of the Internet as represented by wide-area, metropolitan-area and local-area networks.

Producing synthetic topologies that possess similar structural characteristics to the Internet is important since such properties reflect how the Internet is engineered. On the other hand, achieving hierarchical similarities should not be accomplished at the expense of accuracy with respect to other properties such as degree distributions. There must be a generation model that strikes a good balance between structural properties and degree-related properties. We are currently developing such unified models.

BRITE currently supports generation of two-level hierarchical topologies. The two-level limit might be overcome by recursively generating a n–level topology in n phases. However, two-level hierarchical topologies are in concordance to the two-level routing hierarchy that has persisted in the Internet since ARPANET evolved into a network of networks interconnecting multiple autonomous systems. We plan to extend BRITE to natively support more than two levels if we find that it would allow for the generation of topologies that actually reflect real-world scenarios.

3.6.1 Top-down Hierarchical Topologies

Top-down is one of the approaches used by BRITE to generate hierarchical topologies. Figure 5 depicts the structure of the top-down approach for generating hierarchical topologies. The main steps are labeled (1)–(3).

Top-down means that BRITE generates first an AS-level topology (1) according to one of the available flat AS-level models (e.g. Waxman, Imported File, etc.). Next, for each node in the AS-level topology BRITE will generate a router-level topology (2) using a different generation model from the available flat

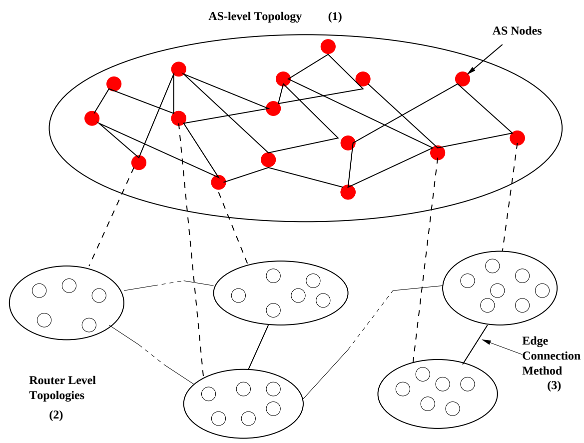

Figure 5: Top-down Approach for Generating Hierarchical Topologies

models that can be used at the router-level. BRITE uses an edge connection mechanism to interconnect router-level topologies as dictated by the connectivity of the AS-level topology. Performing this interconnection of router-level topologies in a representative way is an open research question. BRITE provides four edge connection mechanisms, borrowed from the popular GT-ITM [3] topology generator. The idea is to gradually increase the set of edge connection methods with models that reflect what actually happens in Internet topologies.

The basic edge connection methods provided with BRITE operate as follows. If (i, j) is a link in the AS-level topology, then pick a node u from the router-level topology associated with AS node i, $RT(i)$, and a node v from the router-level topology associated with the AS node j, $RT(j)$, by one of the following methods:

- **Random**: u is picked randomly from $RT(i)$ and v randomly from $RT(j)$.
- **Smallest degree**: u and v are nodes with the smallest degrees in $RT(i)$ and $RT(j)$, respectively.
- **Smallest degree non-leaf**: u and v are nodes of smallest degree in $RT(i)$ and $RT(j)$ respectively but are not leaves.
- **Smallest k-degree**: u and v are nodes of smallest degree greater than or equal to k in $RT(i)$ and $RT(j)$, respectively.

The final topology is obtained by flattening the hierarchical topology into a router-level topology composed of the individual topologies associated with each node at the AS-level.

The configuration file used by BRITE to generate a top-down topology contains parameters controlling the bandwidth distribution for inter- and intra-domain links. These parameters override the specific parameters for the AS- and router-level topologies. Bandwidths for the generated AS-level topology will be assigned according to the inter-domain distribution. Furthermore, bandwidths for each generated router-level topology are assigned according to the intra-domain distribution. During the flattening process, the links established between different router-level topologies will be assigned the bandwidth associated with the corresponding AS-AS link. This bandwidth–assignment method represents just one possible mechanism. Different assignments can be implemented and added to BRITE.

3.6.2 Bottom-up Hierarchical Topologies

Another viable approach to generate hierarchical topologies is the bottom-up approach. Our preliminary results indicate that this approach is promising. The interesting question to be answered with this approach is: how can we infer topological characteristics at the AS-level from known topological information at the router-level. BRITE provides a model that generates hierarchical topologies following this approach.

In this model, BRITE first generates a router-level topology using any of the available models (router Waxman, Imported File, etc.). Once this topology has been constructed, BRITE assigns to each AS node (level-2 node) a number of routers according to an assignment type specified by the user. With this number of assigned routers to an AS node, BRITE groups that many nodes from the router topology following a grouping method specified also by the user as a parameter to BRITE. We next describe the assignment types and grouping mechanisms provided as a base bottom-up model by BRITE.

Assignment Types: The set of parameters associated with the Bottom-up model include $NumAS$, the number of ASs requested by the user. BRITE then assigns router nodes to ASs in one of the following ways:

- **Constant:** Assign each AS an equal number of router-level nodes, i.e. $NumNodes/NumAS$, where $NumNodes$ is the number of nodes in the whole router-level topology.
- **Uniform:** Pick a value uniformly distributed in $[1, NumNodes]$.
- **Exponential:** Pick a value exponentially distributed with mean $\frac{NumNodes}{NumAS}$.
- **Heavy-tailed:** Pick a value from a truncated heavy-tailed distribution between 1 and $NumNodes$.

BRITE enforces two guarantees on the assigment of routers to ASs, namely, that every router is assigned to an AS, and that no AS in the topology is left with zero routers. Details of the implementation are given in [10].

Grouping Mechanisms: Once the number of nodes for an AS has been assigned, BRITE assigns this number of router nodes to the AS in one of two ways:

- **Random pick:** Pick one node at random at a time and assign it to AS i until it reaches the specified size. Repeat for all ASs.

- **Random walk:** Perform a self-avoiding random walk through the graph, where each step in the walk corresponds to choosing a random neighbor from a given vertex. Each visited node is assigned to AS i until it reaches the specified size. Repeat for all ASs.

Providing several ways to group nodes into ASs is aimed at facilitating the process of experimentation. One could implement an assignment mechanism of routers to ASs that mimics the assignment procedure of [16] to compute an AS overlay on top of a measured router-level topology. Developing representative assignment/grouping models is the subject of ongoing research.

3.7 Imported File Model

As we mentioned before, one of the design goals of BRITE was to combine strengths of available models into a single tool. BRITE incorporates an *ImportedFileModel* class deriving from the base abstract class Model. Figure 3 shows the current structure of the ImportedFileModel.

The *Generate* method associated with a model derived from ImportedFileModel parses a file in the format of the corresponding imported topology, and loads it into the BRITE Graph data structures. It can now be used as a native BRITE topology. We have applied this approach to combine topologies from existing generators with topologies generated with a variety of other models. There are many useful scenarios where a researcher may benefit from having such a capability. For example, we could generate a top-down hierarchical topology, where the AS-level topology has been imported from NLANR topological data, and the router-level corresponds to Waxman topologies or topologies generated by the GT-ITM generator. Thus, BRITE's architecture allows a researcher to combine topologies incorporating diverse research themes, as well as to create new models specific to certain scenarios. The available models in this category are depicted in Figure 3 and we are currently working on importing topological data from the CAIDA project, such as the data gathered by Skitter [4].

3.8 BRITE GUI

Figure 6 shows a snapshot of the main window of BRITE's GUI. Through this interface the user can drive the generation process by specifying model parameters, input files, and export formats. A detailed explanation and examples of BRITE functionality and parameters can be found in [10].

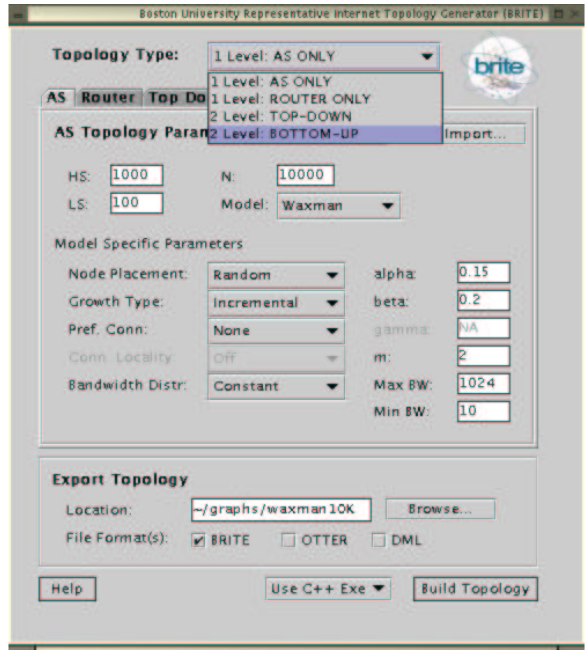

Figure 6: Snapshot of BRITE's GUI main window

4 Comparative Study Using BRITE

In this section we provide a symbolic comparative study of some generation models. The idea of doing this comparison is to illustrate the design principles of BRITE in a "real-world" environment.

Recent empirical studies [6] have shown that Internet topologies exhibit power-laws of the form $y = x^{\alpha}$ for, among other properties, (P1) the outdegree of a node versus rank, and (P2) frequency of an outdegree versus outdegree. The seeming invariance of these properties with respect to size and time suggests they are fundamental properties of Internet topologies. After this discovery, the question to be asked, do the currently used topology generators generate topologies that satisfy these properties [11]?

Using BRITE, we generated topologies according to the *RouterBarabasiAlbert* model and used the *ImportedFileModel* to import GT-ITM flat, GT-ITM Transit-Stub, and NLANR topologies. For each topology, we plot a single property (P1), that is, outdegree versus rank in a log-log plot.

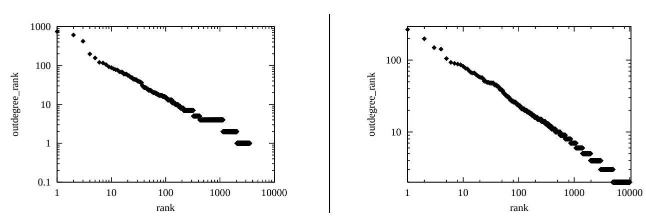

Figure 7: Rank-outdegree for NLANR 04/1998 (left) and BRITE with Barabasi-Albert (right) topologies

The goal of this section is not to make conclusive re-

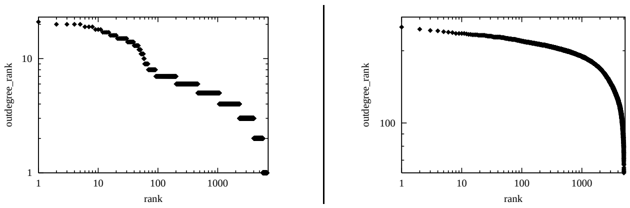

Figure 8: Rank-outdegree for GT-ITM TS (left) and GT-ITM Flat (right) topologies

marks with respect to the differences between the involved models/generators. However, we can see that for the rank of node outdegrees [6], we can establish clear differences between generators aimed at reproducing degree-related properties (e.g. *BRITE/BarabasiAlbert*) and generators aimed at reproducing hierarchical properties (e.g. GT-ITM). In Figure 7 we observe the same type of results obtained in [6] which are reproduced by the *BarabasiAlbert* model implemented in BRITE [11]. In Figure 8 we observe that GT-ITM models lack some characteristic(s) that would allow them to strike a balance between hierarchical properties and degree-related properties.

We want to emphasize that this symbolic comparative study was performed in about 20 minutes using BRITE and BRIANA. Even when the goal was not to reach conclusions from the comparisons, this exercise illustrates how the principles of BRITE and BRIANA translate into an increased efficiency and productivity in the generation and analysis of topologies.

5 Conclusions

Internet research requires good topology generation models that reproduce fundamental properties of the topology of the Internet. It is also a requirement to be able to use such models in simulations in an easy and effective way. In this paper, we have described BRITE, a universal topology generation tool.

We will continue improving the design of BRITE to include multiple inheritance, further import/export formats, and more representative hierarchical models. In the current implementation, the GUI is not extensible and so new models cannot be easily incorporated into the GUI. We also hope to add more analysis routines into BRIANA and incorporate routines that researchers in the networking community have proposed.

A software tool could be said to be successful when it is used for purposes undreamed of by its authors. We hope that new research will shape BRITE and BRIANA and that future releases will incorporate the work of many researchers in the networking community.

References

[1] W. Aiello, F. Chung, and L. Lu. A Random Graph Model for Massive Graphs. In *32nd Annual Symposium in Theory of Computing*, 2000.

[2] A.L. Barábasi and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286:509–512, October 1999.

[3] K. Calvert, M. Doar, and E. Zegura. Modeling Internet Topology. *IEEE Transactions on Communications*, pages 160–163, December 1997.

[4] K.C. Claffy and D. McRobb. Measurement and Visualization of Internet Connectivity and Performance. URL = http://www.caida.org/Tools/Skitter.

[5] M. Doar. A Better Model for Generating Test Networks. In *Proceeding of IEEE GLOBECOM*, November 1996.

[6] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On Power-Law Relationships of the Internet Topology. In *ACM Computer Communication Review*, Cambridge, MA, September 1999.

[7] L. Gao. On Inferring Autonomous System Relationships in the Internet. *IEEE Global Internet*, November 2000.

[8] R. Govindan and H. Tangmunarunkit. Heuristics for Internet Map Discovery. In *Proceedings of IEEE INFOCOM'00*, March 2000.

[9] C. Jin, Q. Chen, and S. Jamin. Inet: Internet Topology Generator. Technical Report Research Report CSE-TR-433-00, University of Michigan at Ann Arbor, 2000.

[10] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: Universal Topology Generation from a User's Perspective. Technical Report BUCS-TR-2001-003, Boston University, 2001. Available at http://www.cs.bu.edu/brite/publications.

[11] A. Medina, I. Matta, and J. Byers. On the Origin of Power-laws in Internet Topologies. *ACM Computer Communication Review*, 30(2):18–28, April 2000.

[12] National Laboratory for Applied Network Research (NLANR). URL = http://moat.nlanr.net/rawdata/.

[13] The Network Simulator (ns). URL = http://www.isi.edu/nsnam/ns/.

[14] V. Paxson and S. Floyd. Why We Don't Know How To Simulate The Internet. In *Proceedings of the 1997 Winter Simulation Conference*, Atlanta, GA, January 1997.

[15] Scalable Simulation Framework (SSF). URL = http://www.ssfnet.org/.

[16] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin. The Impact of Routing Policy on Internet Paths. In *Proceedings of IEEE INFOCOM 2001*, 2001.

[17] W. Theilmann and K. Rothermel. Dynamic distance maps of the Internet. In *Proceedings of IEEE INFOCOM'00*, March 2000.

[18] B. Waxman. Routing of Multipoint Connections. *IEEE J. Select. Areas Commun.*, December 1988.

[19] E. Zegura. Thoughts on router-level topology modeling. Message posted to the end2end-interest mailing list, January 31, 2001.