Studienarbeit

Survey on generators for Internet topologies at the AS Level

am
Institut fuer Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner
Universitaet Karlsruhe (TH)

Author:
Lin Huang

Betreuer:
Robert Goerke

Tag der Abgabe:
January 14, 2007

Ich versichere hiermit, diese Arbeit bis auf die dem Betreuer bereits bekannten Hilfsmittel selbstaendig angefertigt, alle benutzten Hilfsmittel vollstaendig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unveraendert oder mit Aenderungen uebernommen wurde.

Karlsruhe, den January 14, 2007

_____

Lin Huang

Abstract

As the Internet develops, the simulation of the Internet becomes an urgent request nowadays. To obtain an accurate simulation, we need to model the Internet topology. The modeling of the Internet topology started a from random model to the hierarchical model and then it developed to a scale-free network model. Many characteristics of topology have been analyzed with the corresponding metrics, including the prominent Power-Law distribution (eg. frequency, degree). Modeling the Internet topology is still an important open problem, since an accurate topological model can have significant impact on network research. In this paper we discuss some popular Internet topology models and several relevant Internet topology generators at the autonomous system (AS) level, one of which has been developed at our institute.

Contents

1 Introduction

”What does the Internet look like?” or ”How could I generate Internet-like graphs for my simulations?”. The need for realistic random topologies in simulations has long been recognized. Such needs come from following aspects:

1. Many new applications and experiments are not suitable for being directly applied on the Internet, since some of them are dangerous, for instance, the simulation of worms or viruses on the internet.
2. For some protocols which are dependent on network topology, for example, multicast protocols, more simulated topology environments are needed for experiments or estimation.
3. For national security, for instance, project NMS (network modeling and simulation) from DARPA (Defense Advanced Research Projects Agency).

As we can see, with the developments of the Internet, the simulation of the Internet becomes an urgent request nowadays. To obtain an accurate simulation, we need to model the Internet topology. Modeling the Internet topology is an important open problem, since an accurate topological model can have significant impact on network research. We can design more efficient protocols that take advantage of its topological properties. We can create more accurate artificial models for simulation purposes. And we can derive estimates for topological parameters for analysis and speculations. Moreover, future hardware requirements can more easily be predicted, if a reliable model is known.

In this paper we describe some popular Internet topology models and several relevant Internet topology generators at the autonomous system (AS) level. First, we introduce the Internet topology modeling starting from the random model to the hierarchical model and then to the scale-free model. Second, we present some available AS level Internet topology generators, one of which has been developed at our institute. Third, we also discuss some graph metrics and make some comparisons based on these metrics between the real Internet and the generated Internet graph to show how the generated topologies approximate the actual Internet AS topology. The rest of this paper is structured as follows: We present some previous work on topology and power laws which are topology properties often used in modeling or generators.

2 Previous Work

The Internet is composed of connected subnetworks which are known as domains or autonomous systems. They mainly consist of a large collection of routers and are under separate administrative authorities. Hence the study of Internet topology could be conducted at the router level, where each router is represented by a node or at the AS level, where each AS is represented by a node (see Fig 1). In this paper our study focuses at the AS level, since there are too much nodes (routers) at the router level.

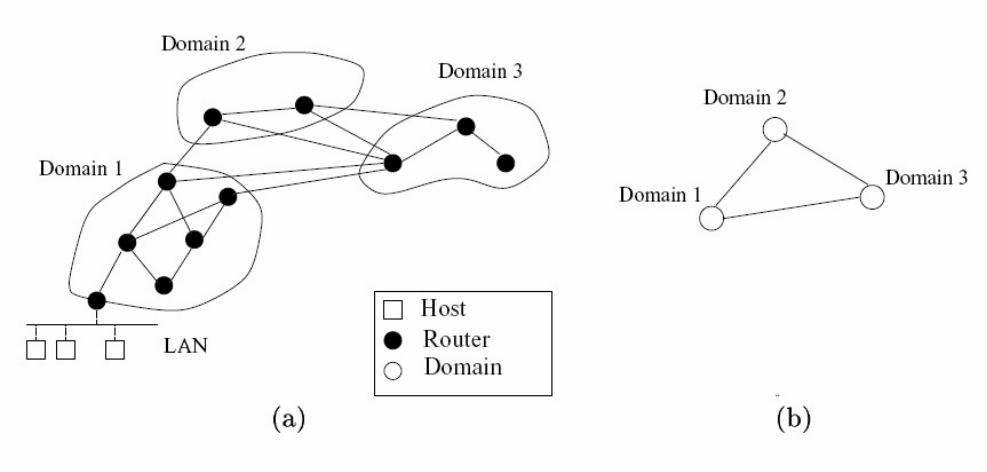


Figure 1: The structure of the Internet at (a) the router level and (b) the inter-domain level

A network topology is usually modeled by an undirected graph where the network devices are modeled by the nodes of the graph and the communication links are modeled by the edges of the graph. Therefore the *outdegree* we will be using later refers to the degree, too. Internet topology models can be divided into two classes: One describes the properties of Internet topology, including Waxman model [JCJ00] (see Section 3.1.1), Tiers [JCJ00] (see Section 3.1.2), Transit-Stub [JCJ00] (see Section 3.1.3) and Power-Laws based models [FFF99] [JCJ00] [WJ]; The other describes the mechanisms of the development of Internet topology properties, including the BA model (Barabasi-Albert model) [BA99] (see Section 3.1.6), ESF (extended scale free model) [AB00] (see Section 3.1.7) and GLP (generalized linear preference) [BT02] (see Section 3.1.8).

Concerning the first class of models, studying the properties of Internet topology actually consists of finding out the metrics that describe the properties. Such a topology model consists of several metrics, the nominal values for these metrics are measured according to the data of the real Internet topology. Within all the discovered Internet topology properties, f_d, the frequency of an outdegree d, is a basic foundation to judge if the topology graph is similar to the Internet topology. In the earlier studies, some researchers consider that the distribution of the outdegree of a node in the Internet is either totally random (Waxman model [BM99] (see Section 3.1.1) or regular (Tiers [JCJ00](see Section 3.1.2)). But with the discovery of power laws [FFF99], it is proved that the Internet topology ranges between both of them. So according to the different characterizations of the frequency of the outdegree, the models in this class could be subdivided into three categories:

1. Random topology. The Internet topology graph is totally orderless. All the nodes are in an equal state. An example of this class is the Waxman model [BM99] (see Section 3.1.1.)

2. Hierarchical topology. The idea of hierarchical topology comes from the knowledge of the Internet structure, which has hierarchical characteristics (like WAN, MAN and LAN in networks). The outdegree of the nodes, which are on the same level, is similar and it is very different if the nodes are on different levels. An instance of this class are the Tiers or Transit-Stub model [JCJ00] (see Section 3.1.2, Section 3.1.3).

3. Power-Laws topology. In such a topology graph, the distribution of the nodes complies with one or more Power-Laws [FFF99]. In the year 1999, Faloutsos et al. analyzed the BGP information of the year 1998 from the National Lab for Applied Network Research (NLANR) and discovered that there are 3 Power-Laws in Internet topology. Power-Laws are expressions of the form $y \propto x^a$, where a is a specific constant of this law, x and y are the measures of interest and $\propto$ stands for "proportional to":

Power-Law 1 (rank exponent): The outdegree, d_v, of a node v, is proportional to the rank of the node, r_v, to the power of a constant, R: $d_v \propto r_v^R$;

Power-Law 2 (outdegree exponent): The frequency, f_d, of an outdegree, d, is proportional to the outdegree to the power of a constant, O: $f_d \propto d^O$;

Approximation (hop-plot exponent): The total number of pairs of nodes, $P(h)$, within h hops, is proportional to the number of hops to the power of a constant, H: $P(h) \propto h^H$, $h \ll \delta$, where δ is the diameter of the graph;

Power-Law 3 (eigen exponent): The eigenvalues, λ_i, of a graph are proportional to their order, i, to the power of a constant, E: $\lambda_i \propto i^E$.

Every node in the graph with a certain outdegree has a rank. The higher the outdegree, the higher the rank is. The frequency refers to the amount of nodes that have the same outdegree. The neighborhood size of a node n within h hops is the number of all the nodes that are reachable to node n within h hops from m. Furthermore, the pair size within h hops is the sum of neighborhood sizes of all nodes within h hops, it thus reflects the connectivity of a graph. The order i is the the order of the eigenvalue, λ_i, in the decreasing sequence of eigenvalues.

Power-Law 1 implies that in the real Internet there is neither such total equality like in the Waxman model [BM99](see Section 3.1.1) nor a strict hierarchy like in Tiers [JCJ00](see Section 3.1.2) and Transit-Stub models(see Section 3.1.3). It suggests a ”loose” hierarchy. Power-laws 1 and 2 reflect that the actual Internet has the character of high irregularity, which means that the minority has greater outdegree, while the majority has smaller outdegree. For example, in the Internet, $R \cong -0.7$, $O \cong -2.2$ at the AS level and $R \cong -0.4$, $O \cong -2.4$ at the router level [FFF99]. The exponent H in the approximation could be used to classify the topology graph. For instance, in the Internet, $H \cong 4.7$ at the AS level and $H \cong 2.8$ at the router level [FFF99]. Power-Law 3 is used to further distinguish two similar graphs of the same kind. An example value of constant E could approximately be -0.4 at the AS level and -0.1 at the router level [FFF99]. All these example of constant values derived from the experiment results on the real Internet [FFF99].

3 Topology Generation Model and Generator

A software tool that creates network topologies is usually called a graph generator or a topology generator. The way it builds network topologies is called a topology model. In this section we will introduce some popular Internet models and some available AS level Internet topology generators, one of which has been developed at our institute.

3.1 Topology generation algorithm

Besides the three topology models we have mentioned before, there are five more topology models which comply with Power-Laws and one map sampling algorithm. The five models could be classified into two parts: one directly decides the outdegree of the nodes according to the Power-Laws, including PLOD (Power-Law outdegree) [PS] and PLGR (Power-Law random graphs) [ACL00]; The other fulfill the Power-Laws by simulating the evolution process of the Internet, including BA [BA99] (see Section 3.1.6, ESF [AB00] (see Section 3.1.7) and GLP [BT02].

3.1.1 Waxman

The Waxman model [BM99] is one of the most popular network models and has been widely used to generate random topologies for network simulations. First all the nodes will be placed uniformly on a 2-dimensional plane and then the model decides if there is an edge to be added between two nodes according to the probability function :

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

where u and v are two nodes, $d(u, v)$ is the Euclidean distance between u and v, α is the average outdegree, β determines the average edge length and L is the maximum Euclidean distance between any two nodes. The value of α and β come from the result of the experiments on the real Internet, $\alpha, \beta \in (0, 1)$ [BM99]. The intuition of this formula is that if tow nodes are far away from each other, then they will not be connected. A random number is generated between 0 and 1. If

$P(u, v)$ is bigger than this random number, then there should be an edge between these two nodes. At last, a spanning tree is created and some necessary edges are added to make sure that the topology graph is connected. The Waxman model works good in representing small networks, because , in my opinion, as the size and complexity of the network raise, just the euclidian distance is insufficient to make the decision.

3.1.2 Tiers

The Tiers model [JCJ00] divides the Internet into three levels: WAN (Wide Area Network), MAN (Metropolitan Area Network) and LAN (Local Area Network). There is only one WAN in every topology. To construct the topology graph, we need to specify the number of LANs and WANs and the number of the nodes per network on every level. On each level the Waxman model is used to generate the topology graph.

3.1.3 Transit-Stub

The Transit-Stub model [JCJ00] is also a hierarchical model. In this model the topology graph has two hierarchical levels: one consists of transit ASs; the other consists of stub ASs. First, a connected random graph of transit ASs is generated. To generate the graph we could use methods: PureRandom, Waxman (recommended) etc. One or more such graphs construct the core of the topology graph. Next, the stub ASs would be connected to transit ASs. Since we use this model in our paper rarely, we would not introduce it very precisely.

3.1.4 PLOD (Power-Law outdegree)

The PLOD model [PS] was carried out by Palmer et al. in the year 2000. After the number of nodes is determined, an outdegree credit for every node will be assigned. The outdegree distribution complies with the appropriate Power-Laws. And then an edge placement loop is executed. It randomly picks two nodes and assigns an edge if they are not connected and each node still has remaining

outdegree credits. Then the credits will be decremented accordingly. The loop continues until there are no more pairs of nodes that fulfill the condition.

3.1.5 PLGR (Power-Law random graphs)

The concept of PLGR [ACL00] was proposed by Aiello et al. in the year 2000. This model is also called Model A. A random graph is produced with the following degree distribution depending on two given values a and b complying with the Power-Laws. Let y be the number of nodes which have been given the degree $x(x > 0)$, then they should satisfy:

$$y = \lfloor e^a / x^b \rfloor$$

where a is basically the logarithm of the number of nodes of degree 1 and b is the log-log rate of decrease of the number of nodes of a given degree. After the degree distribution is defined, a set L will be formed. The set, L, contains $deg(v)$ distinct copies of each node v. Then we choose a random matching of the elements of L. For two nodes u and v, the number of edges joining u and v is equal to the number of edges in the matching of L joining copies of u to copies of v. The graph we get in the end is the Power-Law random graph.

3.1.6 BA (Barabasi-Albert) model

In the year 1999, when Barabasi and Albert from University of Notre Dame researched the scale-free network, they proposed two generic machnisms:

1. **Growth:** networks expand continuously by the addition of new vertices.
2. **Preferential attachment:** new vertices attach preferentially to sites that are already well connected, which means the difference of the connection ability between two nodes are becoming bigger as the network grows, in another word, the rich are getting richer and the poor are getting poorer.

The BA model [BA99], including ESF [AB00] (see Section 3.1.7) and GLP [BT02] (see Section 3.1.8), generates the topology graph by simulating the evolution of

the network according to these two generic mechanisms.

We generate the topology graph by starting with few isolated nodes (n_0 nodes) and periodically a new node with s ($s < n_0$) edges that link to s different in the graph existing nodes, is added to the graph. According to the preferential attachment, the BA model decides for each existing node v whether it is connected to the new node s by *linear preference* $L(v)$:

$$L(v) = \frac{(d_v - c)}{\sum_j (d_j - c)}, c \leq 1$$

Where j is an existing node in the graph, d is the outdegree of a node and c is a constant. The smaller c is, the less preference gives to high degree nodes. As we can see in this formula, if an existing node has a high outdegree, then the probability $L(v)$ is high, which means that the new coming nodes are preferentially connected to this node according to the formula. This linear preference $L(v)$ is also used in ESF [AB00](see Section 3.1.7) and GLP [BT02](see Section 3.1.8). In the BA model c equals 0 [BA99].

3.1.7 ESF (extended scale free model)

The ESF model [AB00] is an expanded version of the above mentioned BA model. It was presented by Barabasi and Albert in the year 2000. In this model the constant c of $L(v)$ equals -1, and there is a new Internet characteristic: In the evolution process of the networks, existing connections might change, which is a process called *rewiring*. Two more constants p, q will be given in advance. The parameters $p(0 < p < 1)$ and $q(0 < q < 1 - p)$ are two probabilities and the values of them are chosen based on Power-Laws.

To generate the topology graph, it also starts with n_0 isolated nodes. Periodically, one of the following three operations is performed:

- With probability p, $n(n \leq n_0)$ new links are added. A node is randomly selected as the start node of the new links and the end node is selected according to the above mentioned $L(v)$. This process is repeated n times.
- With probability q, n links are rewired. First a node is randomly selected. And then a link that is connected to this node is removed and a new link

connected to this node is added. The other node that this new link connects to is decided by $L(v)$. This process is repeated n times.

- With probability $1-p-q$ a new node is added to the graph. This new node has n new links.

3.1.8 GLP (generalized linear preference)

In the year 2002 Tian Bu and Towsley pointed out that there are differences in the value of the characteristic path length and the clustering coefficient between the real Internet topology graph and the topology graphs that are generated by PLGR [JCJ00], BA [BA99] and ESF [AB00]. So they proposed a new topology generation algorithm (model)—GLP [BT02]. In this model the constant c in the linear preference $L(v)$ is not constant any more, but a tunable parameter, $c \in (-\infty, 1)$, that indicates the preference for a new node (edge) connecting to more popular nodes. The algorithm starts with n_0 connected nodes with $n_0 - 1$ edges. Periodically, one of the following operations is performed:

- With probability p, $n(n \leq n_0)$ new links are added. The start node of the link is randomly selected and the end node is chosen by $L(v)$. This process is repeated n times.
- With probability $1-p$, a new node with n new links is added. As always, the end node of the new link is chosen by $L(v)$ in the existing graph.

3.1.9 Map sampling

Map sampling [MP02] was proposed by Magoni and Pansiot in the year 2002. This model only needs to know the number of the nodes and the number of the links of the expected topology graph. And then it randomly extracts a subgraph from the real Internet map with the same amount of nodes and creates a tree by doing the following (also see the example in 3.2.2):

1. A node is randomly selected among all the nodes in the subgraph. All the nodes that are connected to this selected node are stored into a list called

candidate list and the corresponding links are stored in a candidate link list. Step 2 will then be repeated until all the nodes are in the candidate list.

2. A node is randomly picked up in the candidate list and all the other nodes that are connected to this node and still not in the list are added into the list. So are the links, respectively. If there is a node (node n) that is connected to the selected node (node s) and n is already in the list, which means s is also connected to another node in the candidate list, then in the candidate link list, the model has to choose whether to replace the link that n connects to the other node in the candidate list with the link that n connects to s, or to keep the old link and not add the new $n - s$ link to the list. The link that is picked out is going to the redundancy link pool for later use.

Finally the links in the redundancy link pool are picked out and added to the tree to generate the topology graph with the expected amount of nodes and links. In my opinion, unlike other models, map sampling is more like an extraction from the real Internet.

3.2 AS level Internet topology generator

A topology generator is the software realization of the topology model. The design goals of it are as follows:

1. **Representative:** The generated topology graph should precisely reflect as many aspects of the real Internet topology as possible.

2. **Inclusiveness:** The generator is a general topology generator tool that could realize and combine the strength of many different models or algorithms.

3. **Compatibility:** The generator could offer an interface to network simulation application (eg. ns-2) or support other tools.

In this section we will introduce some available AS level Internet topology generator. BRITE [MLMB01] (see Section 3.2.1), Inet [JCJ00] [WJ] (see Section 3.2.3), nem [Mag02] (see Section 3.2.2) are popular used topology generators. The CORE generator [Alb06] (see Section 3.2.4) has been developed by our institute. The general process of the generation of BRITE and Inet is: It places all the nodes on a 2 dimensional plane. Then it assigns the outdegree to each node according to the Power-Law. At last it make all the connections between the nodes. Nem has a special method for topology generation — map sampling, where the graph is generated on the basis of the real Internet topology. The *core* generator uses a totally different idea, the concept of *core*s, to generate the graph.

3.2.1 BRITE (Boston University representative Internet topology generator)

BRITE [MLMB01] is a general topology generator and was developed by Boston University in the year 2001.It generates a topology graph on both the router level and the AS level. The most characteristic feature of it is this ”generality”. It realizes the Waxman model and the Barabasi-Albert model. It could also generate topology graphs by means of the transit-stub (or GT-ITM) hierarchical model in either top down or bottom up way.

Parameter	Meaning	Example values
HS	size of one side of the plane	1000
LS	size of one side of a high-level square	10
NP	clustered node placement	uniform random or pareto
m	number of links added per new node	1,2,3,4,5
PC	preferential connectivity	degree-based only or degree and locality based
IG	incremental growth	enabled

Table 1 BRITE Model Parameters

Table 1 shows the parameters used in BRITE. To generate a topology on a 2-dimensional plane, the plane is first divided into $HS \times HS$ squares. Then some nodes are assigned to each square. The assignment is according to the parameter clustered node placement, NP. The nodes are distributed either uniformly at random or in a bounded Pareto way, in which there are few squares possessing a large number of nodes and many squares with only a few nodes. Now each square is further divided into $LS \times LS$ smaller squares, in which nodes are uniformly distributed. In each of these smaller squares a backbone node is selected to form a spanning tree. Iteratively, new nodes are added in. They are connected to the nodes that are already connected to the backbone. It is the incremental growth that we have mentioned in the BA model (see Section 3.1.6). Each new added node introduces m new links. How these links are added is just like in the BA model (if the BA model is chosen) we have talked about earlier. The new node is always connected to an existing node that has a higher outdegree. However, the authors of BRITE conjecture that not only the outdegree but also locality has impacts on the preferential attachment. This means that the nearest nodes have a higher priority to be chosen. Hence there is a parameter in BRITE, preferential connectivity, PC, deciding whether the locality influences the connectivity when a new links are added.

BRITE can generate a topology graph on both the router level and the AS level. It offers interfaces to many kinds of network simulation applications, including ns-2 [ISI06], OMNeT++, JavaSim etc. It also supports visual tools, like Otter [BHC] from CAIDA. Besides, it can assign a bandwidth and a latency to each connection and has a friendly graphic user interface as shown in Fig 2. The user can set all the parameters by this GUI(see Fig 2: The *topology type* is used to generate the topology at the AS level or at the router level. The *model* is used to generate the graph (eg. Waxman); We can also set the value of HS, LS, α, β (if Waxman is

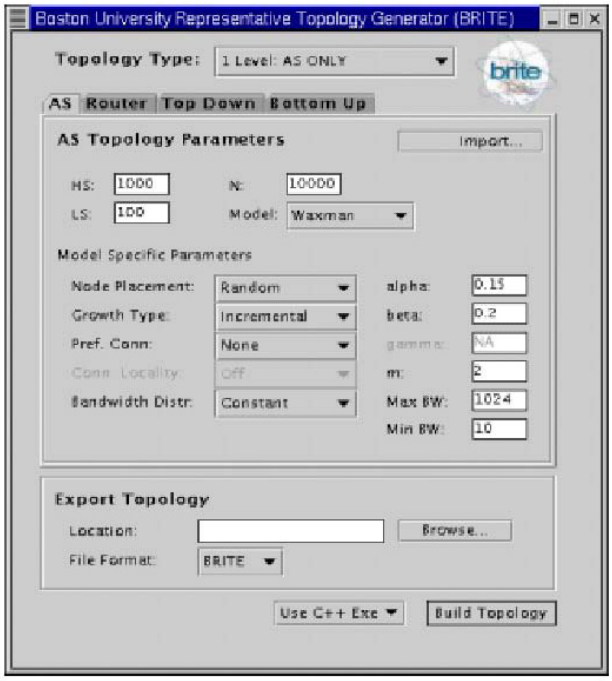


Figure 2: Graphical User Interface of BRITE

used) and the bandwidth and the output file etc.

3.2.2 Nem (network manipulator)

Nem [Mag02] is a topology generator software developed by the University of Louis Pasteur Strasbourg. Nem generates a topology graph on both the router level and the AS level. It can generate a topology graph not only by the method of map sampling, which is an algorithm proposed by themselves, but also with the Waxman model, the BA model, PLOD or model A. We have already talked about how these models work (see Section 3.1.9), so here we focus on map sampling and give out a small example of it.

As it is shown in Fig 3: First a node (node 1) is randomly chosen. Node 2 and 5 are added to the candidate list because they are connected to node 1 and links a

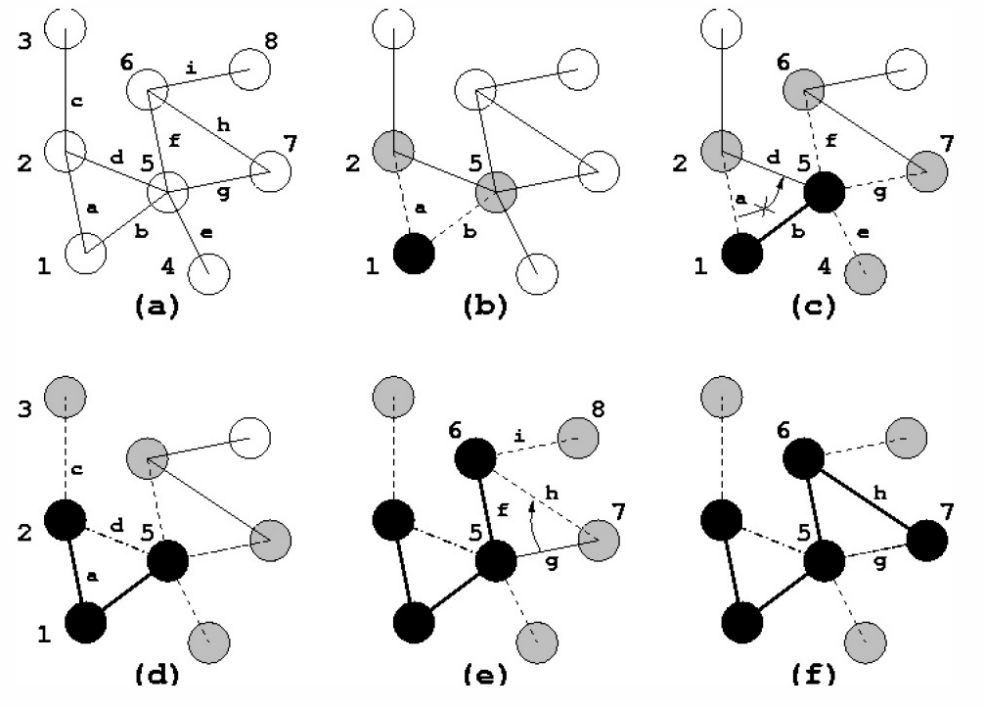


Figure 3: Tree creation process

and b are added to the candidate link list. Therefore nodes 1, 2, 5 are now in the candidate list. Then randomly node 5 is picked out from the candidate list and adjacent nodes 4, 6, 7 are added to the candidate list and links f, g, e to the candidate link list respectively. Now we can see, that node 2 is also connected to node 5, but it is already in the candidate list. So either link a or link d could be kept in the candidate link list and the other goes into the redundancy link pool. This is done randomly. In our example, link a is chosen. Now this process is repeated until all the nodes are in the candidate list. Node 2 is selected, then put node 3 and link c into the appropriate lists, link d is still not selected this time. Node 6 is selected and node 8 and links i are going into the lists. Link g is decided to be replaced by link h and added to the redundancy pool. Until now all the nodes have been through and the tree is completed. The links will be added from the redundancy link pool to the tree to complete the topology graph.

Unlike BRITE, nem has no graphical user interface. So to use nem to generate a topology graph, we need to write a specification file by ourselves. This file includes all the values of parameters that nem needs to create a topology graph. It

has an extension *.specific. The following is an instance of how to write a specification:

// prefix used to generate the network filename(s) (default: graph)
name_prefix nem_4000
// number of nodes in the generated graph (default: 1000)
nodes_nb 4000
// number of nodes in the generated graph (default: 3000)
edges_nb 8000
// network level of the generated graph (default: router_level)
network_level route_level
// graph generation method (default: magoni_pansiot_sampling)
generation_method magoni_pansiot_sampling

Then we write a process file (as follows), which has an extension *.process:

myFile.specific C1

where myFile is the specification we write and C1 is the type of the output file where the generated topology file should be stored. So that we could specify, which kind of output file we need to use. For example we could generate a topology graph in xml format, if we specify the output type as C5. The number that stands for the type of the file are listed in table 2.

Network file format	Extension	Input	Output	Number
nem	*.nm	Yes	Yes	1,2,3
ns-2	*.tcl	No	Yes	4
OPNET Modeler (xml format)	*.xml	No	Yes	5
ITM (alternate format)	*.alt	Yes	Yes	6
H3Viewer	*.h3	No	Yes	40
Tiers (generic format)	*.tiers	Yes	No	
BRITE	*.brite	Yes	No	
Inet2.x	*.inet	Yes	No	
Mercator (native format)	*.mercator	Yes	No	
Mercator (anonymized format)	*.mercator_anonym	Yes	No	
ASmap (route-views format)	*.as_map	Yes	No	
ASconnlist (route-views format)	*.as_conn_list	Yes	No	

Table 2 Network file formats in nem

The process file is executable. If we give a command like:

nem processFileName

then a topology graph will be generated as described in the specification file.

Besides the topology generation this software has two more main functions: Converting network files from one format to another and analyzing the topology of networks.

As we can see from Table 2, nem can load many other types of topology graph such as BRITE or Inet. It provides interface to network simulation applications like ns-2, OPNet etc.

Nem divides the characteristics of the network topology into five categories: Forest (tree), distance, connectivity, number of shortest paths of node pairs and class. To analyze a topology graph, nem will calculate all these properties of the graph and store it in a file whose extension is *.analysis.

3.2.3 Inet

Inet [JCJ00] [WJ] is an Internet topology generation software. It was developed by University of Michigan during the years 1999 to 2002. It generates a topology graph only at the AS level. It applies the PLGR model and preferential connectivity to comply with Power-Laws. Through the research on a large amount of BGP information on route-views.oregon-ix.net during the years 1997 to 2002, the values of the topology properties in Inet are pretty precise and reliable.

Inet-1.0. The first version places all the nodes on a 2-dimensional plane. Each node is assigned an outdegree based on Power-Law 2. Then the top τ nodes which have the highest outdegree construct a full mesh. 25% of all the links whose start

node is the node in the mesh and end node is outside the mesh are connected to the nodes which have outdegree 2. All other nodes are connected to either the nodes in the mesh or to the nodes which have outdegree 2. Then the topology graph is completed. The shortcoming of version 1 is that the research information shows that in the Inet graph 1.5% - 2% of the nodes, which have the largest outdegree, don't comply with Power-Laws.

Inet-2.0. Version two is designed on the basis of version one. 98% of the nodes are assigned an outdegree according to Power-Law 2 , just like in version 1. Meanwhile the 2% of the nodes which cause problems in version 1, are in version 2 assigned with the outdegree according to Power-Law 1. After the outdegree assignment a spanning tree is created with all the nodes whose degree is larger than 1. The nodes with outdegree 1 are connected to the spanning tree. If in the end there is node with outdegree 1 left and in the tree there are no more nodes with a free outdegree for the left nodes to connect to, then the network is not connected and the spanning tree is not valid. If there is no node with outdegree 1 left, this means the spanning tree is successful. Then this tree will be built and other nodes will be connected to the tree in the principle of largest node first (*preferential connectivity*). That means a node is first connected to the node which has larger free outdegree.

Inet-3.0. By the observation of the vertex cover of the real Internet and Inet-2.2 an increasing difference was found. It turns out that the preferential connectivity in version 2 is the cause the this difference. The preferential connectivity says that a node is first connected to the node who has larger free outdegree. For example, between a node with outdegree 4 (free outdegree 3) and a node with outdegree 6 (free outdegree 1), the former will be chosen to be connected to another node. It forces many low outdegree nodes to connect to other low outdegree nodes. But actually in the real Internet map low outdegree nodes are more willing to connect to higher outdegree even if its free outdegree is low. Inet-3.0 makes improvements in this aspect. To choose which node to connect to depends now on the probability $P(i,j)$. The two nodes with the highest $P(i,j)$ are connected.

$$P(i,j) = \frac{w_i^j}{\sum_{k \in G} w_i^k}$$

Where w_i^j ,the weight value of d_j with respect to d_i, is in the form of:

$$w_i^j = \max(1, \sqrt{(\log \frac{d_i}{d_j})^2 + (\log \frac{f(d_i)}{f(d_j)})^2}) \cdot d_j$$

Where d_i, d_j are the outdegrees of node i, node j, $f(d_i)$ is the frequency of the outdegree.

We can take this formula as an improvement of above mentioned linear preference $L(v)$. If i has a higher outdegree, then the value of w_i^j is bigger. As we can see, the bigger the value of w_i^j is, the higher the probability is and if the outdegree difference between two nodes is large, they have a big value of w_i^j. This realizes that low outdegree nodes connect to the highest outdegree node. The value of $f(d)$, in my opinion, has an "even-out" function. It means, that nodes are not connected with nodes who has high outdegree but low quantity.

It is easy to use Inet to generate a topology graph. All we need to do is to give an Inet command with parameters (eg. number of the nodes). It is in the form of:

inet -n N [-d k] [-p n] [-s sd] [-f of]

For instance: "inet -n 5000 $>$ Inet.5000". A topology graph with 5000 nodes will then be created and stored in output file named Inet.5000. Table 3 shows the parameters used in Inet.

Parameter	Meaning
-n N	the total number of nodes in the topology.
-d k	the fraction of degree-one nodes. Default is 0.3.
-p n	the size of the plane used for node placement. Default is 10,000.
-s sd	the seed to initialize the random number generator. Default is 0.
-f of	the debugging output file name. Default is stderr.

Table 3 Inet parameters

3.2.4 CORE

The generator CORE [Alb06] is a software that has been developed at our institute. It generates a topology graph using a different concept compared with BRITE, Inet and nem.

The concept of *core*s was proposed by Seidman in year 1982, when he was researching the structure of the network. We call a set of nodes a k-*core*, if each node in the set is connected to at least k other nodes in the set. We call a set of nodes a k-*core-shell* if all the nodes in this set belong to k-*core* and not belong to the $(k+1)$-*core*. To get the *core*s of a graph, the following steps are repeated:

1. Putting all nodes with degree i (starting with i=0) into *shell* i
2. Removing them from graph
3. Searching again repeatedly in new graph until no more nodes with degree i are found.
4. Increase i by one.

Fig 4 is an example of k-*core* and k-*core-shell*. The biggest circle is *core* 0; the second big circle is *core* 1; the third big circle is *core* 2 and the smallest circle is *core* 3. Concerning the *core-shell*, the number of the nodes in *shell* 0 is 1 because only the node in *core* 0 and not in *core* 1 belongs to *shell* 0. Accordingly, the number of the nodes in *shell* 1 is 4, in *shell* 2 is 8 and in *shell* 3 is 8.

As we can see, there is some relationship between the *core* and the outdegree of a graph. A node with a low outdegree definitely doesn't belong to a *core* with a high order. However, a node with a high outdegree could belong to a *core* with low order. For instance, a node in *core* 2 could have a infinite outdegree. Actually, the real Internet network has in many cases such a characteristic *core* structure. Let's take a look back at Fig 1. The AS graph has some interesting characteristics, such as its *core* structure. $70-85\%$ of the nodes are in *core* 1 and *core* 2 but not in *core* 3. The *core*s with higher order have fewer nodes. However the maximum *core* number ($k \cong 26$ with 20k nodes in the real Internet) again is very large.

It is not necessary that a *core* or a *shell* is all connected. They can be composed of modules. We can see in Fig 4 that *core* 2 consists of two modules. There are two kinds of modules: active modules and passive modules. We call a module an active module if in this module there is at least one node that connects the node to a higher *core*. Otherwise, the module is a passive module. Furthermore, there are two kinds of links: internal links and external links. We call a link an external link if this link connects two nodes that are in different *core*s. Otherwise, the link

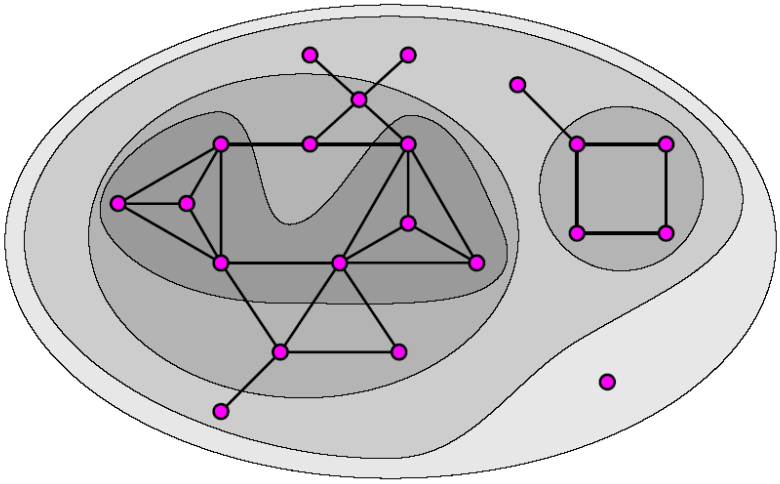


Figure 4: k-core Structure

is an internal link.

To generate a topology graph with CORE we need to know the number of *core*s, the number of the nodes per *core*, and optionally the number of internal links and external links. Meanwhile, there are some restrictions for nodes and links, followed:

For active modules:

- if $n > k : n - 1 \leq m_i \leq \frac{nk-(k^2+k)}{2}$
- if $n \leq k : n - 1 \leq m_i \leq \frac{n(n-1)}{2}$
- and $m_e + m_i \leq nk$

Where n is the number of the nodes, k is the order of the *core*, m_i is the number of internal links and m_e is the number of external links.

For passive modules:

- $\lceil \frac{nk}{2} \rceil \leq m \leq nk - \frac{nk-(k^2+k)}{2}$

where n is the number of the nodes, k is the order of the *core* and m is the number of the links.

From the *core*s we can determine the *shell*s and how many nodes are in one *shell*. In a *shell* the nodes are divided into modules with internal an external links respectively. In a passive module, all the nodes are first connected to a circle and then all the rest of the links are added to the nodes to make sure that every node has the minimal value of the degree to stay in this *core*. Et is also necessary to add the links in the way so that some nodes will not jump to a higher *core*. In a active module, nodes are connected to a circle at first too. And then internal links are added in the same way as in the passive module. Then we can add external links. In an active module there might be nodes that still have not enough links to stay in this *core*. The external links will be first connected with these nodes. And the rest of the external links will be added into the module with the control that no nodes jump to a higher *core*.

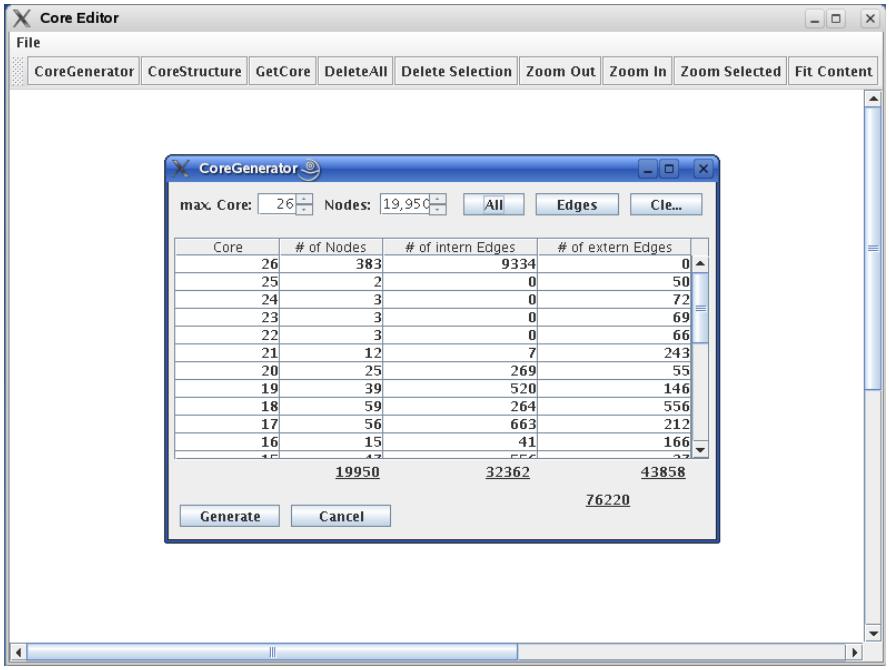


Figure 5: Core Editor

The CORE generator has a friendly graphical user interface (see Fig.5). Especially, it can draw and show the topology graph. It can also analyze a topology graph with all the *core* properties. But we need to get Core statistics from Internet and feed them to the core-based generator.

4 Comparison between different models and generators

In this section, we discuss some graph metrics and make some comparisons on the basis of these metrics between the real Internet and the generated Internet graphs in order to show how the generated topologies approximate the actual Internet AS topology. The comparison is divided into two parts. The first part consists of some comparison results of some early version of generators and models. These comparisons are made by other researchers [JCJ00]. In the second part are some comparison results of latest version of the generators.

4.1 Metrics used for comparison

As we know, the metrics such as minimum, maximum and average values of distance are way too insufficient to fully measure the Internet network nowadays. After the Power-Laws were discovered, many other ideas of topology properties came up, such as frequency, rank etc. Moreover, our institute proposed to use the *core* property, which was applied in the network, to extracted from the Internet network to describe it. In the following we present some metrics we will use in our comparison.

1. **Rank and outdegree Power-Law.** The first Power-Law is $d_v \propto r_v^R$, where d_v is the outdegree of a node v, r_v is the rank of node v on a sorted list in decreasing order of node degree and R is the rank exponent.

2. **Frequency and outdegree Power-Law.** The second Power-Law is $f_d \propto d_v^O$, where d is a value of an outdegree, f_d is the the number of the nodes who have outdegree d and O is the frequency exponent.

3. **Pair size within h hops.** The neighborhood size of a node m within h hops is the number of all the nodes that are reachable to node m within h hops from m. And pair size within h is the sum of neighborhood size of all nodes.

4. **Clustering coefficient.** It is a popular metric used in "small-world" graph. It is used to describe the number of small groupings in the graph. If node v has m_v neighbors, then among all the neighbors there are at most $M =$

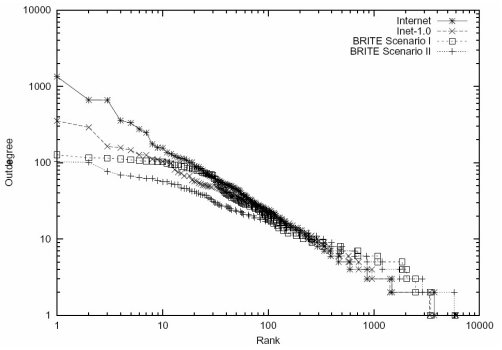
$\frac{m_v(m_v-1)}{2}$ edges, C_v is the actual number of the edges among the neighbors divided by M. The clustering coefficient of the graph is the average of all the C_v. It shows how many of a node's neighbors are adjacent to each other.

5. **Core.** A node is in the k-*core* if it is connected to at least k nodes that are also in the k-*core*. The set of all nodes in the k-*core* is called the k-*core*. We call a set of nodes a k-*core-shell* if all the nodes in this set belongs to k-*core* and not belongs to $(k+1)$-*core*.

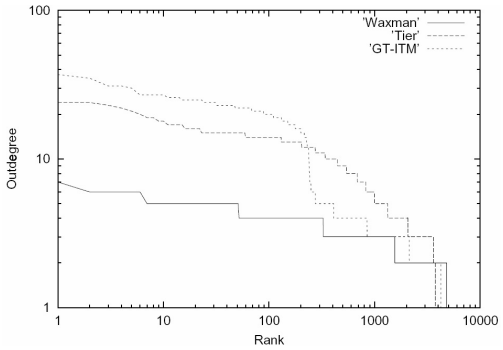
4.2 Comparison results

4.2.1 Part I

In this part there are some comparison results of some early version of generators and models. These comparisons are made by Cheng J. et al. [JCJ00] in the year 1999 when they developed Inet. Six different generators and models generated topology graphs ranging in size from 3000 to 6000 nodes and the results are compared with real Internet topology. The used generators are Waxman, Tiers, GT-ITM, Inet-1.0, BRITE Scenario I (the new link is added only based on outdegree) and BRITE Scenario II (the new link is added based on outdegree and locality). We show the results of 6000 nodes.



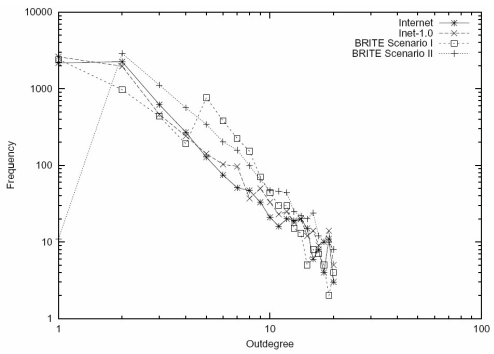
(a) Inet-1.0, BRITE Scenario I, Scenario II and the real Internet with 6000 nodes (rank vs. outdegree)



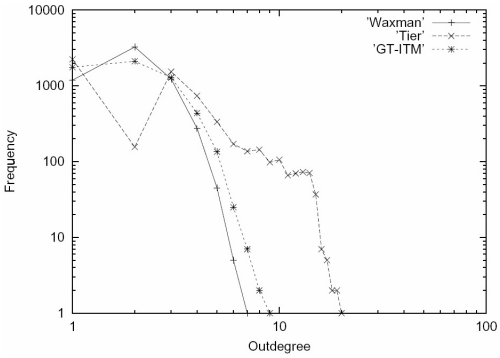
(b) Waxman, Tier, GT-ITM and the real Internet with 6000 nodes (rank vs. outdegree)

Figure 6: Comparison data matching Power-Law(1)

Fig 6 shows that concerning rank-outdegree, Inet and BRITE are quite close. However model Waxman, Tiers and GT-ITM don't comply with the first Power-Laws. In the Waxman model the outdegree of a node is not specified at all. Whether two nodes are connected are decided totally randomly. The nodes are uniformly distributed on the plane and the outdegree of the nodes is pretty uniform. That is why Waxman has a very small maximum outdegree. Inet or BRITE, on the other hand, have some few nodes that have the highest outdegree, just like in the real Internet network.

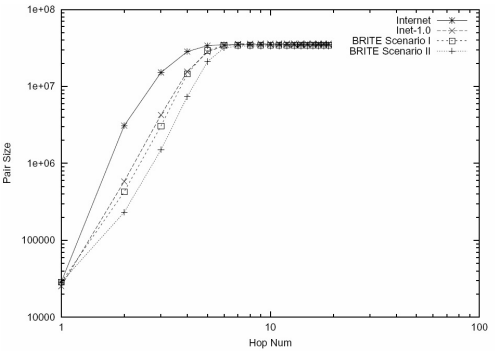


(a) Inet-1.0, BRITE Scenario I, Scenario II and the real Internet with 6000 nodes (frequency vs. outdegree)

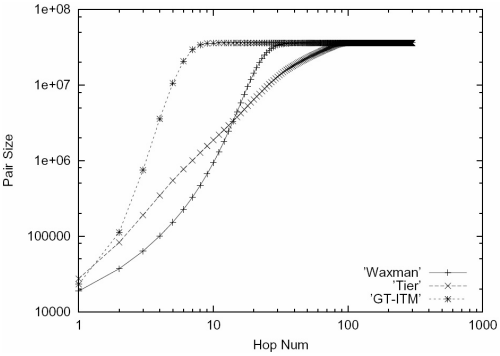


(b) Waxman, Tier, GT-ITM and the real Internet with 6000 nodes (frequency vs. outdegree)

Figure 7: Comparison data matching Power-Law(2)



(a) Inet-1.0, BRITE Scenario I, Scenario II and the real Internet with 6000 nodes (pair size vs. hops)



(b) Waxman, Tier, GT-ITM and the real Internet with 6000 nodes (pair size vs. hops)

Figure 8: Comparison data matching Power-Law(3)

In Fig 7 we can see that concerning outdegree 1 nodes, there is a very big difference between the BRITE scenario II generated topology and the real Internet network. There should be a large amount of nodes that has an outdegree 1 while a few nodes have a very high outdegree. Neither the Waxman nor the Tiers model generate a node that has an outdegree larger than 10. That is also due to the uniform assignment of outdegree of the nodes in the model.

In Fig 8 we can find some discrepancies between topologies generated by Inet and BRITE and the real Internet network when the hop is bigger than 1. For Inet-1.0, the generated topology is due to the generator using only Power-Law 2, but not Power-Law1, in determining node outdegree distribution, which is corrected in the later version. The growth rate of pair size of Waxman and GT-ITM generated topology are much faster than the one pair of real Internet. Only Tiers has a growth similar to the Internet.

4.2.2 Part II

All the comparison results in this part are done by ourselves. We use Inet-3.0, BRITE-2.1 and CORE to randomly generate topology graphs at the AS level with 19950 nodes, whose number is the number of the nodes in the snapshot of a real Internet network on May 31. 2005 (oix-full-snapshot-2005-05-31-2000). Due to the limitation of nem-0.96 that it can only generate graphs with nodes whose amount should be less than one third of the amount of the nodes from the real Internet graph which nem offers in the software, we generate topology graphs with nem in size of 4000 nodes and observe the distribution. For each generator we randomly generate four graphs and get average values of them.

	Inet	BRITE	CORE	nem	Real Internet
num_nodes	19950	19950	19950	4000	19950
num_edges	52406	39897	72460	5200	42682
density	0.000263358	0.000200496	0.00036413925	0.000650163	0.000214492
min_degree	1	2	1	1	1
max_degree	3152	402	1032	734	2407
av_degree	5.25373	3.9997	7.26421	2.6	4.2789
*core*_number	17	2	26	4	26
triples	24580967	574322	4152786	588548	11529606
triangles	57208	176	72701	501	45453
transitivity	0.00698199	0.0026688345	0.05897895	0.005401535	0.0118269
clustering _coefficient	0.49690225	0.0027039775	0.150066975	0.211463325	0.377508

Table 4 Comparison data between Inet, BRITE, CORE and Internet.

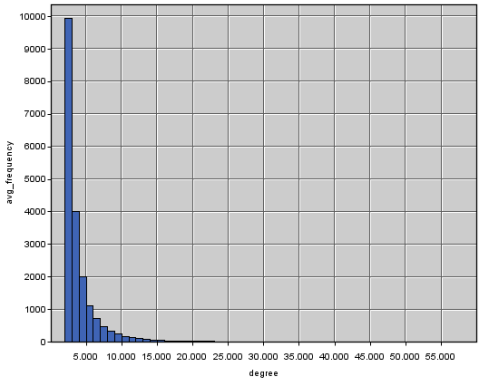
In table 4 we can find that the minimal degree of Internet is 1, but the result from BRITE is 2. It is because that with BRITE we use the BA model and there is a parameter m to give when we use BRITE. It decides how many new nodes with how many new links are added to the graph periodically. The default value of it is 2. That means every once in a while a nodes with outdegree 2 are added the graph. Therefore the minimal degree of BRITE here is 2. In my opinion, we can not set m too big, otherwise there would be too less nodes with small outdegree in this graph, which is not true in the real Internet. To take a look at the clustering coefficient. Inet has a better results then the other two and BRITE seems not good enough. As we have already introduced, clustering coefficient shows how many

of a node's neighbors are adjacent to each other. Concerning the *core*, we can see, except for CORE generator, Inet has the best result.

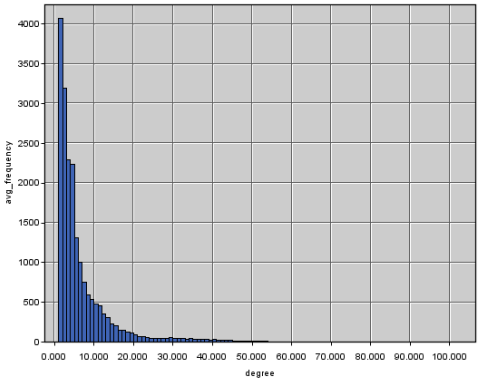
Fig 9 is the results of degree vs. frequency. The distribution of Internet and Inet is almost the same. They all have about 8800 nodes that have outdegree 2 and more than 6000 nodes have outdegree 1. BRITE has no node of outdegree 1. That is because every new nodes to be added has an outdegree 2. In real Internet there are still lots of nodes having only one connection to others. But if we reduce the value of parameter m to 1 in BRITE, other metrics such as clustering coefficient are getting worse. In nem and CORE most nodes have outdegree 1. It is different from the real Internet. CORE's distribution is more average than the real Internet. There are a lot of nodes who have outdegree between 5 and 20. In the real Internet only few nodes have large outdegrees, as described in preferential attachment (see Section 3.1.6).

Fig 10 is the results of rank vs. degree. Inet has also a better result here. The trend of the distribution of ranks in BRITE, CORE and nem are very similar between them and the real Internet, but not as precise as Inet.

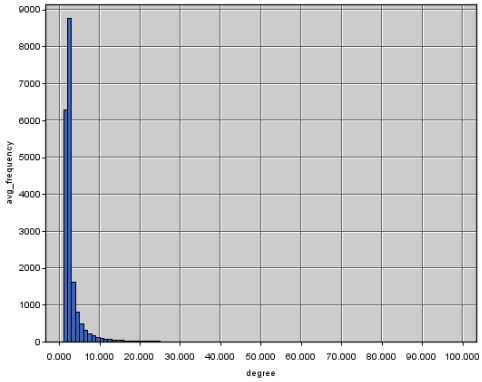
Fig 11 is the pair size with h hops results. Inet here is not very good. First it has only 7 hops in all, while the real Internet has 10 hops. Concerning about this, CORE and nem are better. Except that it takes them 11 hops to go through all the nodes, the distribution are very similar to the real Internet. The first half are almost the same.



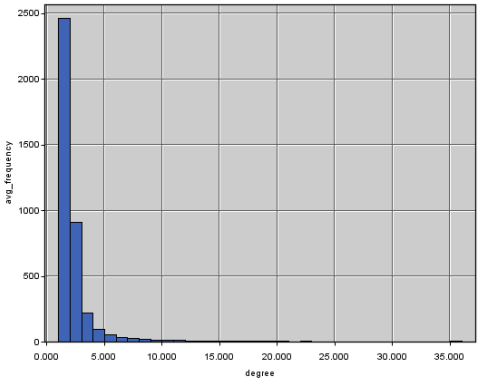
(a) BRITE with 20k nodes



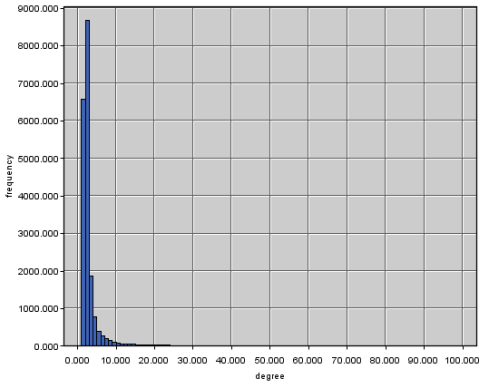
(b) CORE with 20k nodes



(c) Inet with 20k nodes

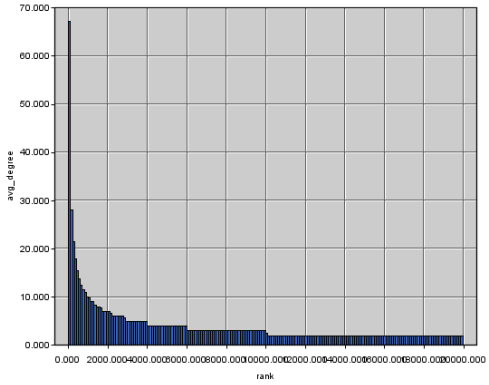


(d) nem with 4k nodes

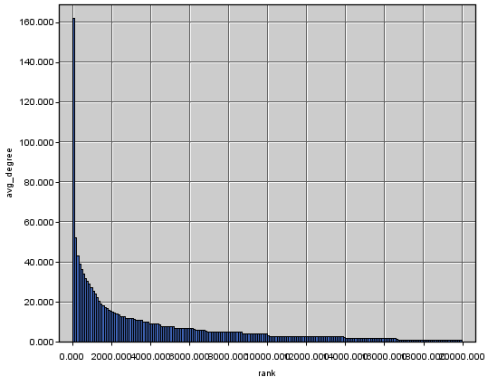


(e) the real Internet with 20k nodes

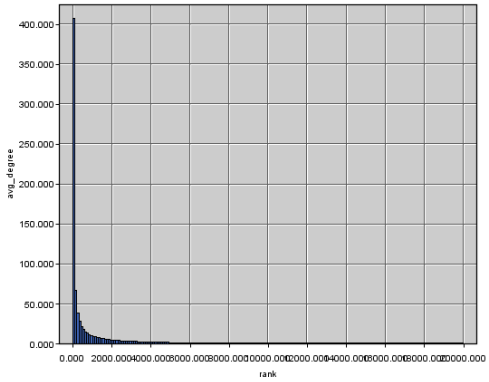
Figure 9: Comparison results of frequency vs. outdegree between BRITE, CORE, Inet, nem and the real Internet



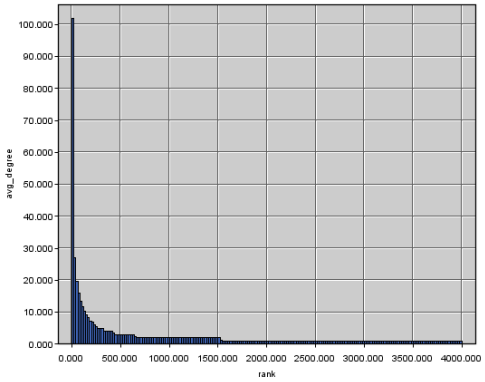
(a) BRITE with 20k nodes



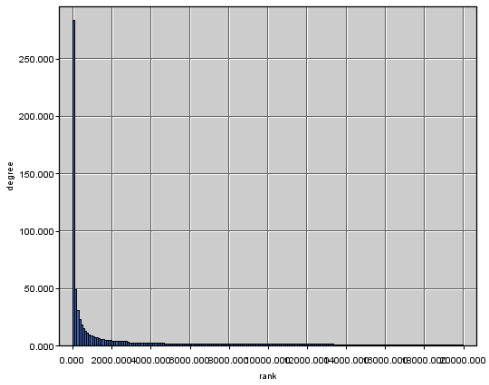
(b) CORE with 20k nodes



(c) Inet with 20k nodes

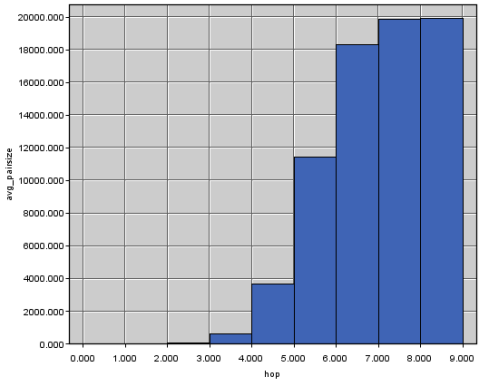


(d) nem with 4k nodes

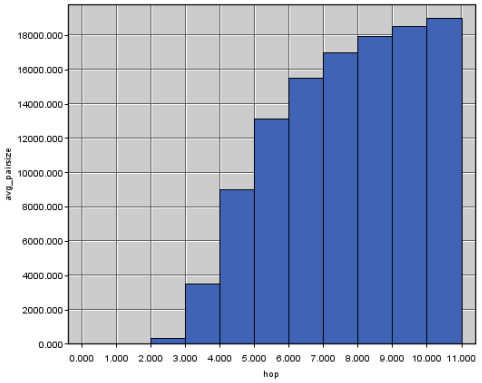


(e) the real Internet with 20k nodes

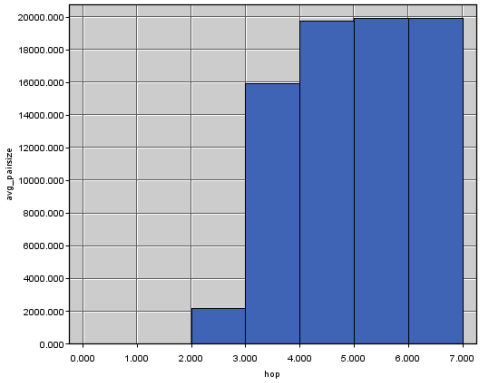
Figure 10: Comparison results of outdegree vs. rank between BRITE, CORE, Inet, nem and the real Internet



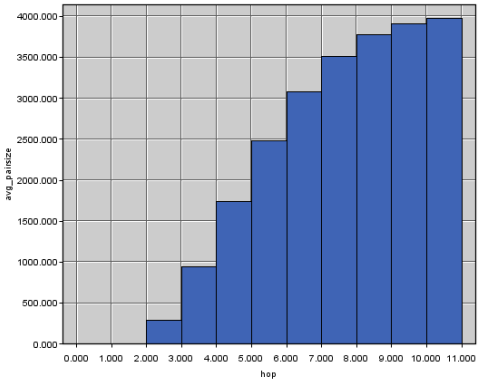
(a) BRITE with 20k nodes



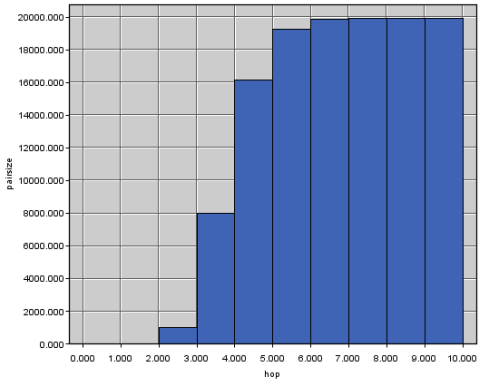
(b) CORE with 20k nodes



(c) Inet with 20k nodes



(d) nem with 4k nodes



(e) the real Internet with 20k nodes

Figure 11: Comparison results of frequency vs. outdegree between BRITE, CORE, Inet, nem and the real Internet

5 Conclusion and Acknowledgement

Through more and more research we find that the Internet is not only hardware and software. It is somehow an independent system with a metabolism. Lacking about the knowledge of Internet is now not only a problem in computer science areas, but also a problem of deficiency of scientific methods for characterizing complex network in general. Simulating the real Internet is very necessary for application, security and research reasons. Therefore in this paper we discuss some popular Internet topology models and several relevant Internet topology generators at the autonomous system (AS) level and include a model based on a metric of (*cores*) that could be an important property of the Internet. We also make some comparisons between the real Internet and the topology graphs that were generated by all the generators described in this paper.

In my opinion, to generate a topology graph at the AS level, generator Inet might be a better tool to use. It is more precise and reliable. All the values of metrics for Inet are more close to the values of the real Internet, including the metric *core* proposed at our institute. As the comparison results shown above, generator CORE is still not good enough to simulate the real Internet. However, results of CORE might be able to get improved by setting many more parameters to fit the real Internet. For example, in our comparison results we only specify the *core* number to generate a graph, it could be more precise if we specify the number of internal and external links according to value of the real Internet. Generator nem has its limitation when map sampling is used — it could only generate graphs with small amount of nodes and there is still some bugs to generate topology graphs at the AS level. Nevertheless, on the other hand, it affords additionally analysis and conversion functionalities that make it easy to observe the real Internet.

In addition, the main aspects for researching the Internet topology could be as follows:

1. Developing new topology measurement techniques so that we can discover more new important properties, that characterize the Internet.
2. Combining graph theory and data mining to find new characteristics from all the data that we already have.
3. Looking up the research results of other complex networks, since there are

general principles among all the complex networks. This could help to discover the internal mechanism of Internet topology development.

At the very end, I just want to appreciate all the help offered by my supervisor, Robert Goerke and Professor Wagner.

References

[AB00] R. Albert and A. L. Barabasi. Topology of evolving networks: local events and universality. volume 85, pages 5234–5237, December 2000. http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve\&db=pubmed\&dopt=Abstract\&list_uids=11102229.

[ACL00] William Aiello, Fan Chung, and Linyuan Lu. A random graph model for massive graphs. pages 171–180, 2000.

[Alb06] David Albrecht. Generieren von graphen mit core-hierarchie. 2006.

[BA99] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. volume 286, pages 509–512, October 1999. http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve\&db=pubmed\&dopt=Abstract\&list_uids=10521342.

[BHC] E. Nemeth B. Huffaker and K. Claffy. Otter: A general-purpose network visualization tool.

[BM99] Waxman BM. Routing of multipoint connections. In *SIGCOMM*, pages 251–262, 1999. http://citeseer.ist.psu.edu/michalis99powerlaw.html.

[BT02] T. Bu and D. Towsley. On distinguishing between internet power law topology generators. 2002. http://citeseer.ist.psu.edu/bu02distinguishing.html.

[FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262, 1999. http://citeseer.ist.psu.edu/michalis99powerlaw.html.

[ISI06] The University of Southern California Information Sciences Institute. The network simulator - ns-2. 2006.

[JCJ00] C. Jin, Q. Chen, and S. Jamin. Inet: Internet topology generator. 2000. http://citeseer.ist.psu.edu/jin00inet.html".

[Mag02] D. Magoni. nem: A software for network topology analysis and modeling. In *MASCOTS '02: Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02)*, page 364, Washington, DC, USA, 2002. IEEE Computer Society.

[MLMB01] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. BRITE: Universal topology generation from a user's perspective. Number 2001-003, 1 2001. `http://citeseer.ist.psu.edu/medina01brite.html"`.

[MP02] Damien Magoni and Jean-Jacques Pansiot. Internet topology modeler based on map sampling. In *ISCC '02: Proceedings of the Seventh International Symposium on Computers and Communications (ISCC'02)*, page 1021, Washington, DC, USA, 2002. IEEE Computer Society.

[PS] Christopher R. Palmer and J. Gregory Steffan. Generating network topologies that obey power laws. In *Proceedings of GLOBECOM '2000*.

[WJ] Jared Winick and Sugih Jamin. Inet-3.0: Internet topology generator. `http://citeseer.ist.psu.edu/526211.html"`.