UNIVERSITY OF CALIFORNIA, SAN DIEGO

Mechanisms for Generating Realistic Annotated Internet Topologies

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in
Computer Science

by

Priya Mahadevan

Committee in charge:

Professor Amin Vahdat, Chair
Professor Kimberly Claffy
Professor Bill Lin
Professor Ramesh Rao
Professor Stefan Savage
Professor Alex Snoeren

2007

DEDICATION

To my parents for always believing in me

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The past few years have provided me with opportunities to work and interact with a diverse set of people who have all contributed to making my graduate school experience interesting and enjoyable. First and foremost, I would like to thank my advisor, Dr. Amin Vahdat, for his support, guidance, constant feedback and positive outlook over these years. Amin has been instrumental in helping me identify and understand research problems and how to address them. I have learnt a lot from him on how to communicate research ideas better by way of effective writing and presentation skills. I would especially like to thank him for his constant encouragement to work towards goals that I would not have otherwise targeted.

I am fortunate to work with my committee members - Dr. Stefan Savage, Dr. Alex Snoeren, Dr. k claffy, Dr. Bill Lin and Dr. Ramesh Rao. They have provided valuable feedback that has helped make this dissertation better. I would also like to thank Dr. Geoff Voelker and Dr. George Varghese for providing insightful comments and feedback during my time at UCSD. Dr. Jeff Chase, Dr. Carla Ellis and Dr. Alvy Lebeck helped shape my initial years in graduate school at Duke University and I am ever grateful to them.

I would like to thank Dr. Dmitri Krioukov for his guidance and critical feedback. I would also like to thank Bradley Huffaker, Dr. Marina Fomenkov and other researchers from CAIDA for their help. Colleen Shannon and David Moore have been very supportive and have given useful feedback on numerous paper drafts and practice talks.

Dr. Joann Ordille has been a great mentor throughout my graduate school days and has been a constant source of encouragement. I am grateful to Dr. Jeanne Ferrante for being a wonderful motivator and for providing me the opportunity to attend the Grace Hopper Conference. The administrative staff at both UCSD and Duke University have been extremely helpful, in particular Julie Conner, Michelle Panik and Diane Riggs.

I would like to thank my office mates and colleagues who have made my stay in graduate school fun - Kashi Vishwanath, Jeannie Albrecht, Alvin Auyong, Barath Raghavan, Diwaker Gupta, John McCullough, Chip Killian, Dr. Dejan Kostić, Qing Zhang, Kiran Tati, Ranjita Bhagwan, Alper Mizrak, Jay Chen, Sangmin Lee, Dr. Flavio Junqueira, Dr. Renata Texeira, Dr. Adolfo Rodriguez, Dr. Lipyeow Lim, Lisa Cowen, Roshni Malani, Patrick Verkaik, James Anderson and other sysnet folks. My special thanks to Priyanshu Jain, Reena Mathew and Kavitha Kannan for their wonderful company, constant support and providing the much needed distractions outside of work.

None of this would have been possible without the support of my family. The values of hard work, dedication and sincerity that my parents have inculcated in me have stood me in good stead during my days in graduate school. Their unconditional support, love and belief in me have made me what I am today. My brother Prashant has been motivating in his very unique way. His eternal optimism has helped me see the brighter side of things. I am deeply indebted to Rahul for always being there for me and helping me overcome all obstacles. No amount of thanks will ever do justice to his love, patience, guidance, motivation and selflessness.

Chapter 2, in part, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Computer Communications Review (CCR), January 2006, Mahadevan, Priya; Krioukov, Dmitri; Fomenkov, Marina; Huffaker, Bradley; Dimitropoulos, Xenofontas; claffy, kc; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Computer Communications Review (CCR), January 2006, Mahadevan, Priya; Krioukov, Dmitri; Fomenkov, Marina; Huffaker, Bradley; Dimitropoulos, Xenofontas; claffy, kc; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Conference, Pisa, Italy, September 2006, Mahadevan, Priya; Krioukov, Dmitri; Fall, Kevin; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapters 5 and 6, in full, are a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Conference, Kyoto, Japan, August 2007, Mahadevan, Priya; Hubble, Calvin; Krioukov, Dmitri; Huffaker, Bradley; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

VITA

2007	Doctor of Philosophy University of California, San Diego San Diego, CA
2003	Master of Science Duke University Durham, NC

PUBLICATIONS

Priya Mahadevan, Calvin Hubble, Dmitri Krioukov, Bradley Huffaker and Amin Vahdat, “Orbis: Rescaling Degree Coorelations to Generate Annotated Internet Topologies.” In *Proceedings of the ACM SIGCOMM Conference*, Kyoto, Japan, August 2007.

Priya Mahadevan, Dmitri Krioukov, Kevin Fall and Amin Vahdat, “Systematic Topology Analysis and Generation Using Degree Correlations.” In *Proceedings of the ACM SIGCOMM Conference*, Pisa, Italy, September 2006.

Priya Mahadevan, Adolfo Rodriguez, David Becker and Amin Vahdat, ”MobiNet: A Scalable Emulation Infrastructure for Ad Hoc and Wireless Networks.” *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, April 2006.

Priya Mahadevan, Dmitri Krioukov, Marina Fomenkov, Bradiley Huffaker, Xenofontas Dimitripoulos, kc claffy and Amin Vahdat, ”The Internet AS-Level Topology: Three Data Sources and One Definitive Metric.” *ACM SIGCOMM Computer Communications Review (CCR)*, January 2006.

Aameek Singh, Arup Acharya, Priya Mahadevan and Zon-Yin Shae, ”SPLAT: A Unified SIP Services Platform for VoIP Applications.” *Wiley’s International Journal of Communication Systems (IJCS) (Special Issue on VoIP - Theory and Practice)*, 2006.

Priya Mahadevan , Adolfo Rodriguez, David Becker and Amin Vahdat, ”MobiNet: A Scalable Emulation Infrastructure for Ad Hoc and Wireless Networks.” *International Workshop on Wireless Traffic Measurements and Modeling (WiTMeMo) in conjunction with MobiSys*, Seattle, WA, June 2005.

Aameek Singh, Priya Mahadevan , Arup Acharya, and Zon-Yin Shae, ”Design and Implementation of SIP Network and Client Services.” In *Proceedings of the 13th International Conference on Computer Communication and Networks (ICCCN)*, Chicago, IL, October 2004

Amin Vahdat, Ken Yocum, Kevin Walsh, Priya Mahadevan , Dejan Kostic, Jeff Chase, and David Becker, ”Scalability and Accuracy in a Large-Scale Network Emulator.” In *Proceedings of 5th Symposium on Operating Systems Design and Implementation (OSDI)*, Boston, MA, December 2002.

FIELDS OF STUDY

Computer Systems

ABSTRACT OF THE DISSERTATION

Mechanisms for Generating Realistic Annotated Internet Topologies

by

Priya Mahadevan
Doctor of Philosophy in Computer Science
University of California, San Diego, 2007
Professor Amin Vahdat, Chair

Researchers involved in designing network services and protocols rely on results from simulation and emulation environments to evaluate correctness, performance and scalability. To better understand the behavior of these applications and to predict their performance when deployed across the Internet, the generated topologies that serve as input to simulated and emulated environments must closely match real network topologies with respect to a wide range of graph metrics proposed in the literature. Values for a particular graph metric may capture a graph's resilience to failure or its routing efficiency. Unfortunately, there are typically no known algorithms to generate graphs matching one or more proposed metrics and there is little understanding of the relationships among individual metrics or their applicability to different settings. Furthermore, the generated topologies must also be annotated with observed network characteristics that, for instance, include latencies for the edges in a router-level graph, and AS membership for the nodes in the graph. Finally, it should be possible to rescale a given topology to a variety of sizes while still maintaining its essential characteristics.

We present a new, systematic approach for analyzing and synthesizing network topologies. We first introduce a unifying series of properties, the dK-series of probability distributions, specifying all degree correlations within d-sized subgraphs of a given graph G. Increasing values of d capture progressively more properties of G and, in the limit, describes the graph in its entirety. Using the dK-series, we construct random graphs and demonstrate that these graphs reproduce, with increasing accuracy, all important metrics of measured and modeled Internet topologies. The nature of the dK-series implies that it will also capture any future metric that may be proposed. Further, we scale the generated graphs to a wide range of sizes while still preserving the graph's structure. Finally, we propose techniques to annotate these topologies and specifically describe our scheme to annotate Internet router graphs with AS-membership as well as whether a router is peering or internal.

Chapter 1

Introduction

The Internet has revolutionized our computing capabilities and communications like nothing before. From its initial form, ARPANET, that had 4 computers connected to form a network in 1969, the Internet today has grown to several hundred million hosts. While the exact number of hosts in the Internet is unknown, one study [45] observed 434 million hosts in February 2007. It is estimated that 17.2% of the world's population now has access to the Internet [43]. There are approximately 410 million Internet users in Asia alone, followed by 320 million users in Europe [43].

With this unprecedented growth, both in the number of hosts and the number of users, in the past decade, it is only natural that the number of applications and protocols developed for the Internet has also grown exponentially. Some of the early applications included electronic mail, file transfer and remote login. Over the years however, these simple applications have given way to far more complex, large-scale distributed applications, capable of supporting millions of simultaneous users. Examples of large scale distributed systems and services include online shopping applications, services for making flight and hotel reservations, online banking and trading tools, instant messaging applications, audio and video conferencing capabilities and realtime audio and video streaming.

These services are playing an increasingly important role in our daily lives, making their security, reliability, and performance of paramount importance. Therefore, system designers and application developers need to ensure that these applications meet certain scalability, usability, availability and reliability requirements. Having built these applications and services, the developers cannot directly deploy them full-fledged over the Internet. Ideally, they would like to evaluate these applications in a realistic but controlled environment in order to gain more confidence about the behavior of these applications when actually deployed over the Internet.

Unfortunately, the scale and complexity of the Internet makes it nearly impossible to evaluate network services under a range of realistic conditions. These conditions include evaluating application behavior under expected workloads, as well as over other network conditions such as sudden bursts in traffic, lossy links, *etc*. It is also critical to understand how these applications perform when unexpected events occur. For example, one would like to understand application behavior when there is a partition in the network, or when the service is subject to a distributed denial of service (DDoS) attack.

While simulation and emulation tools are available for evaluating network applications, one key limitation of such environments is their inability to capture the rich and complex characteristics of real networks. Typically, one needs to specify as input several important parameters such as the input network topology over which the application is to be evaluated, annotations to this topology such as link latencies, bandwidths and loss-rate for all edges in the topology, queue sizes and drop policies for all the routers, background traffic flows across the topology, congestion levels, network failure models and so forth. In addition, one might also have to supply application specific parameters. For example, while evaluating a web server, application specific parameters include file sizes requested by the web clients and arrival rate of all the requests at the web server. Determining which input parameters to specify is dependent on the application being evaluated and the research questions being investigated [32].

Modeling each of the above components such that they reflect known and observed Internet conditions is difficult and challenging in its own right. In this dissertation, we focus on modeling annotated network topologies. We discuss the limitations of current methodologies in generating network topologies and propose novel techniques for producing realistic random graphs that closely resemble measured network topologies.

1.1 Importance of Topologies

Knowledge of network topologies is crucial for understanding and predicting the performance, robustness and scalability of network protocols and applications. Routing and searching in networks, robustness to random network failures and targeted attacks, the speed with which worms spread in a network, strategies for traffic engineering and network management all depend on the topological characteristics of a given network. In this section, we briefly describe how topologies impact different aspects of networking.

Evaluating and predicting application performance and behavior: Researchers and system developers aiming to improve the performance and reliability of Internet applications and protocols face a daunting task when it comes to evaluating their design. The developer must implement their proposed solution and evaluate it under conditions that match realistic Internet scenarios. There are typically three options for performing such an evaluation: live network deployment, simulation, and emulation.

Live deployment involves running the application across hosts that are directly connected on the Internet. PlanetLab [77] and VINI [6] are examples of such testbeds. Live deployment has the benefit of offering realistic workloads. Additionally, evaluating the application over hosts spread across the Internet also provides confidence that the application will perform well if actually deployed across the Internet. However, results produced in this environment are difficult to repeat as network conditions on the Internet change continuously. Isolating faults that manifest themselves only under certain conditions require running experiments in a tightly controlled manner, which is not feasible in the Internet. Moreover, deploying and administering development software at more than a handful of Internet sites is a strenuous task.

Simulators such as ns2 [73] have been widely adopted by the networking community as an alternative technique to live deployment. Simulators offer more control over the target platform, and allow repeatability of experiments, which makes debugging and fault isolation easier. However, simulators typically have limited scalability and usually require rewriting the software in a language specific to the simulation framework. Rewriting the application for the simulator not only leads to increased development efforts, but also masks key details of real implementation. Further, the effect of deploying the application over a real operating system, sending real packets over a real TCP/IP stack is lost. Evaluating the application in such an environment usually leads to differences in observed behavior as compared to when deployed over the Internet.

Emulators combine the repeatability of simulation with the realism of live deployment. Examples of large-scale emulators include [24, 102] as well as our own effort, ModelNet [98]. Emulators allow users to run unmodified applications on commodity operating systems while subjecting the applications to some user-specified network topology and characteristics. For example, in ModelNet, users specify a target topology over which they would like to run their tests; ModelNet nodes running the application are configured to route their packets through a set of ModelNet core nodes, which cooperate to subject the traffic to the bandwidth, congestion constraints, latency, and loss profile of each hop in the target network topology.

As in all simulators, emulators, and even live deployment testbeds such as VINI [6], the user is required to specify as input the target topology over which the application is to be evaluated along with parameters for latency, bandwidth, *etc.*, for each link in the target topology.

Both the structure of the network, *i.e.* how the nodes are interconnected in the network, as well as its characteristics such as latency and loss-rate play a significant role in application behavior and performance. Even minor variations in network structure and characteristics can impact application and protocol behavior and performance [81, 32, 52, 51]. Consider the case where an application running on a node s is communicating with another application running on node d. If the shortest path between nodes s and d comprises of 5 hops, then the application packets from s have to traverse all of these 5 hops before reaching d. Now, if the topology is modified so that s and d are separated by merely 1 hop, application packets from s to d will have to traverse just this 1 hop and therefore the application behavior in this new topology is likely to be different from its behavior on the old one. As a result, we might not be able to correctly predict application behavior if we run our experiments over non realistic topologies. Therefore, it is crucial to accurately model real observed topologies for realistic evaluation of wide-area applications and services.

Content distribution networks: Many emerging large-scale applications and services depend on accurate information about the topology and its characteristics. For example, content distribution networks such as Akamai [3] and Coral [33] dynamically redirect each client to the replica that yields the best performance based on the underlying network state. Consider the case when one replica is 5 hops away from the client and another replica is 10 hops away. In this case, the ability to determine the closest replica to the client requires knowledge of the underlying topology. With such knowledge, the system can choose the closest replica to service the client, thus improving performance at the client site.

Peer-to-peer systems [93, 84, 21, 49, 15] and overlay networks can also incorporate information from the underlying network in order to better choose their peers. For example, for increasing fault tolerance, a host in a peer-to-peer system can choose peers such that all the paths from the host to all its peers are disjoint.

Traffic Engineering: Topology is also important from the perspective of traffic engineering [4, 28]. Once the network is deployed, an Internet Service Provider must map its customer traffic flows onto the physical topology. Traffic engineering provides the ability to move traffic flows away from the shortest path and onto a potentially less congested physical path across the service provider's network. Such a mapping requires intimate knowledge of the underlying topology. One recent effort [47] uses multiple paths in the ISP's network to meet customer demands, adaptively moving traffic from over-utilized to under-utilized paths.

Routing: Performance of routing protocols critically depend on the network topology [51]. Recent studies have shown that even though Internet-like topologies have the ability to support efficient routing techniques, the current routing architecture does not exploit this ability [51, 9]. Researchers have shown that convergence of routing protocols such as BGP is impacted by the topological structure of the Internet [52, 1]. Thus, a better understanding of the Internet's topology will help researchers and engineers develop more efficient routing architectures.

Network Evolution: In addition to the important role played by topology in the above examples, it is also important to understand the evolution of the network. Based on the current network structure and its evolution over time, the ability to predict the structure of the Internet when it, for example, grows to double its current size will help in developing futuristic technologies and applications. Another important motivation for understanding the topology is to direct network engineering efforts. If it is known that certain connectivity structures lead to better fault tolerant networks, then we can target our engineering efforts to construct topologies with these connectivity properties.

1.2 Requirements of a Topology Generator

For all of the scenarios described in the previous section, one needs to be able to generate a wide variety of graphs that are structurally similar to real observed networks. We note here that it is possible to represent the Internet topology at several levels.

One level is the Internet's AS topology. An Autonomous System (AS) is a collection of networks controlled usually by a single entity and that presents a common routing policy to the Internet. In an AS-level topology, a node represents an AS, while an edge represents a peering relationship between two ASes. A peering-relationship is a contractual agreement to exchange traffic directly between the two ASes. There are two classes of peering relationships: customer-provider and peer-to-peer. In a customer-provider relationship, one AS acts as a customer while the other AS provides the customer with Internet access (usually for a fee). In a peer-to-peer relationship, two ASes mutually agree to connect to each other and thus obtain direct access to each other's network. It is important to note that two ASes can peer at multiple points, *i.e.* they can connect physically with each other at multiple places. However, in an AS-level topology, a single edge corresponds to the peering relationship between two ASes. Measurements of the Internet's AS-level topology include skitter [48], Route Views [82], and WHOIS database [44].

Another representation is the Internet's router topology. Here, a node represents a router, while an edge between two nodes represents a direct physical connection between two routers. DIMES [72], Mercator [95], Rocketfuel [92], iPlane [55] and skitter [48] all provide information on the router connectivity map of the Internet. The type of topology an experimenter needs to use depends on the application under study. For experiments involving intra-AS routing, one requires an AS-level topology. Evaluating a web service in an emulator or simulator requires specifying a router-level topology of the network.

While these measured topologies are publicly available, researchers cannot restrict their experiments to these measured topologies alone. The reasons are manifold. For example, they may want to explore how their application or protocol performs with minor variations in the topology to better understand application sensitivity to the current Internet structure.

Consider the case when an experimenter may have developed a new interdomain routing protocol and wishes to test its sensitivity to a range of possible deployments. While the experimenter may test the behavior of his or her protocol relative to a known Internet AS topology, he or she may also be interested in the sensitivity of the protocol to other potential (random) topologies that follow similar node interconnectivity patterns. If the protocol displays widely varying behavior to these random graphs (that have similar node inter-connectivity), then it might be concluded that the protocol's behavior may not be robust to a range of possible settings. The researcher might also want to experiment with various 'what-if' scenarios, such as what if some of the Tier 1 Internet Service Providers changed their peering arrangements. In this case, the researcher might want to modify the graph specifically to account for the new peering arrangement and again validate the routing protocol over this new graph.

More importantly, the experimenter may be interested in the behavior of the protocol under a different scale. For instance, the new routing protocol may be claimed to be more scalable than existing protocols. In this case, the experimenter would require access to topologies that display the same local connectivity characteristics, but that may be much larger than the measured graph. Similarly, the experimenter may require access to smaller topologies because of scalability limitations in existing simulation and emulation evaluation environments (or because only smaller scale topologies can be configured in future deployment environments such as GENI [35]). This scenario requires the need to evaluate the protocol over random graphs of sizes different from the measured AS topology.

Thus, in addition to available topologies, we also require the ability to generate random graphs that are structurally similar to observed graphs and also of a wide range of sizes. The generated topologies should also support a range of annotations important to higher-level studies. For instance, nodes in the generated router-level topology should be annotated with AS membership information to enable studies that account for routing behavior. Similarly, the business relationships among peering ASes (peering, customer, *etc.*) should be included. Without such annotations, Internet routing would have to default to shortest path between end hosts. Other important annotations include link latencies, loss rates, and capacities for edges in the router-level topology.

Furthermore, the topology generator should stand the test of time. The Internet is heterogeneous and is undergoing change at a rapid rate [74]. Thus, it is likely that the Internet topology (both AS- and router-level) might exhibit different connectivity patterns in a few years. Therefore, all proposed algorithms to generate random graphs matching the Internet's connectivity characteristics today should also be capable of reproducing the graph characteristics of the Internet of the future.

In summary, a topology generator should be able to perform the following tasks:

1. Generate random graphs that are structurally similar to the measured topology, *i.e.* the generated random graphs should faithfully reproduce all metrics of the original graph. Additionally, any graph generation algorithm should also be able to reproduce any metric that may be proposed in the future.

2. Effectively scale the graph sizes while still maintaining the structure of the observed topology.

3. Annotate the generated graphs such that nodes in a router-level topology can be assigned AS membership, while edges in this graph can be annotated with latency, capacity and loss-rate.

1.3 Limitations of Current Topology Generators

The *state-of-the-art* in network topology research can be broadly classified into the following three categories: (i) developing measurement techniques to map the Internet's current topology, (ii) proposing metrics that characterize key properties of the topology, (iii) developing algorithms to generate graphs that reproduce some of the metric values of real observed networks.

Over the years, a wide variety of network topology metrics have been suggested to measure the similarity between two graphs. Researchers have shown that these metrics have a direct influence on the performance and behavior of applications and protocols. Thus, topology generators aim to produce random graphs that match the metric values of observed networks.

These important metrics include graph spectrum (the largest and smallest non-zero eigenvalues provide tight bounds for critical network characteristics including network resilience [54] and network performance [94]), betweenness (closely related to link value [94] and router utilization [54]), distance distribution (or path-length distribution, a renormalized form of expansion [94]), and clustering (used to compare accuracy of topology generators [12]).

While all the above metrics are individually interesting, there are a number of problems with the current set of available network topology properties. First, they derive from a wide range of studies, and there is no systematic way to determine which metrics should be used in a given scenario. Second, there are no known algorithms to construct graphs with desired values for most of the described metrics. Thus, while it is possible to determine whether two input graphs have, for example, similar distance distribution, it is currently not possible to generate graphs that precisely reproduce a specified form of distance distribution. In the same vein, spectrum and betweenness characterize global graph structure, while known graph-generating algorithms are unable to reproduce these metrics.

Another important limitation of these metrics is that they are essentially incomplete. A future metric may be proposed that might impact a certain class of applications. Identifying such a metric might result in finding that known synthetic graphs do not match this new metric. In this case, one will have to rewrite the topology generation algorithm to account for this new metric and this process could go on forever.

Unfortunately, the current *state-of-the-art* in topology generation does not produce graphs that reproduce the metrics of real observed topologies. Earlier work shows that existing tools do not reproduce the complex structure of router-level topologies [54]. Existing AS-level topology generators produce AS-level graphs, representing entire Autonomous Systems as a single node in the graph with links between ASes representing peering relationships. Furthermore, these tools include either no annotations or use simple heuristics known not to reflect Internet characteristics. Finally, existing techniques typically either cannot perform graph rescaling or do not do so in a manner that reflects known patterns of network evolution.

1.4 Hypothesis and Contributions

Given the requirements of a topology generator for producing representative Internet graphs, the hypothesis of this dissertation is, “It is possible to generate random graphs of a wide range of sizes that match observed topologies with respect to all metrics proposed or that may be proposed in the literature.” If out hypothesis is true, it will be possible to build a topology generator that meets all of the requirements described in Section 1.2 above.

To verify our hypothesis, we propose a novel methodology for generating Internet topologies. First, we identify a series of graph properties, called the dK-series, to unify all the set of topology metrics identified in the literature so far. By using the dK-series as a basis, we propose techniques to generate random graphs that also automatically reproduce all the metrics of the observed topology. Further, we present algorithms to generate graphs of sizes vastly different from that of the original graph, while still maintaining the structural properties of the original graph. Specifically, in this dissertation, we make the following contributions:

1. We present a finite set of enumerable graph properties, the dK-series, to describe and constrain random graphs in successively finer detail. In the limit, these properties describe any graph completely. In our model, we make use of probability distributions on subgraphs of size d in some given input graph. We call *dK-graphs* the sets of graphs constrained by such distributions.

2. We develop and implement new algorithms for constructing random graphs that match the corresponding distributions of the dK-series. We show that our generated random graphs automatically match *all* graph metrics proposed in the literature as well as *any* graph metric that maybe proposed in the future. Validity of the last strong statement follows from the fact that in the limiting case, our synthetic graphs converge to the original: if two graphs are the same, then *all* their metrics are the same too. This unique convergence feature makes our approach

stand separately from all the previous topology research and concludes the quest for a single set of definitive topology metrics.

3. We present algorithms to generate graphs of different sizes from the original. There are a range of possible techniques for performing rescaling; we propose considering historical Internet connectivity data to inform such rescaling.

4. We present a top-down technique for generating router-level topologies annotated with AS membership. We further classify routers in this topology based on whether they are peering or internal routers. We compare our randomly generated, annotated router topologies to observed Internet router topologies and find close matches for a range of graph metrics proposed in the literature.

Our topology generator will serve as valuable input to a range of research studies discussed in Section 1.1. We outline some of the cases here. Studying routing protocol scalability and convergence requires knowledge of both topology and AS relationships and hence our work can serve as input to such work. Many studies on congestion control protocols employ simple "dumbbell"-style topologies. While such simple topologies are appropriate starting points, it will be valuable to consider more complex topologies, for instance with more variable round trip times and multiple, changing bottlenecks. Many overlay and peer-to-peer systems attempt to create application-level logical topologies that match the characteristics of the underlying network. Similarly, developing network coordinates [20, 71, 87] and geo-localization [14, 105] has recently become an important research area. Our topology generator can supply a range of inputs and potential deployment scenarios in support of such studies.

Emerging network testbeds such as VINI [6] and GENI [35] will enable network topology configuration for deployed systems running across the wide area. Once again, running with a range of topologies, scaled to fit available resources, will allow more accurate conclusions to be drawn for emerging network architectures. Our generator will be equally applicable to producing future Internet graphs. Multiple aspects of network security efforts, including defenses against denial of service attacks and large-scale worm outbreaks depend on network topology. The ability to experiment with a range of random graphs that match Internet characteristics to understand the sensitivity of particular techniques to network topology (and to variations in network topology) will be of significant value.

Additionally, since all of our graph generation algorithms do not depend on the type of the topology (whether AS- or router-level) and the evolution of the network, all of our graph analysis and synthesis techniques are applicable to non-Internet graphs as well. Our topology generator will also benefit researchers in biology, sociology, physics, mathematics and economics where graph synthesis and analysis is important. As an example, biologists use protein networks to study interaction among different kinds of proteins. Economists use graphs to study trade patterns among different countries, sociologists use graphs to study interaction amongst people. All of these studies will benefit from our topology generator.

1.5 Organization

The rest of this dissertation is organized as follows. Chapter 2 describes *state-of-the-art* in network topology research. We discuss all important graph metrics proposed in the literature as well as efforts to produce graphs mimicking values of these metrics. In order to identify our unifying property series, we first analyze Internet AS-level topologies extracted from three commonly used data sources. Our analysis helps understand the interplay among all the graph metrics and identify the property that appears fundamental to the graph. We describe our study and its implications in Chapter 3. In Chapter 4, we define the unifying property series, the dK-series, and present algorithms to generate random graphs matching the properties of the dK-series. We also present empirical evidence that our generated graphs reproduce all known metric values for comparatively complex Internet AS-level and router-level topologies. In Chapter 5, we present algorithms to generate graphs of sizes vastly different from that of the original graph. Chapter 6 describes our techniques to annotate router-level topologies with AS-membership information for each router. Finally, in Chapter 7, we discuss a number of scenarios that could benefit from our topology generation techniques, we present our conclusions and outline avenues for future work.

Chapter 2

Background and Related Work

In the previous chapter, we emphasized the importance of topologies in networking and the need to accurately model the topology of the network under study. The Internet's topological properties and their evolution are cornerstones of many practical and theoretical network research agendas. Evaluation environments that will enable development, reliable testing, and performance evaluation of new applications, protocols, and routing architectures that require, as input, annotated network topologies that closely resemble the structure and characteristics of real measured networks. Researchers have been focusing on various aspects of network topology study for the past couple of decades. In this chapter, we take a look at the *state-of-the-art* in network topology research.

Research involving network topology, particularly Internet topology, generally investigates the following two questions:

1. Generation: Can we efficiently generate ensembles of random but "realistic" topologies by reproducing a set of simple graph metrics?

2. Evolution: What are the forces driving the evolution (growth) of a given network?

In Figure 2, we illustrate the methodologies used to answer these questions in the left, bottom, and right parts, respectively. Common to these methodologies is a set of practically important metrics used for analyzing and comparing sets of graphs

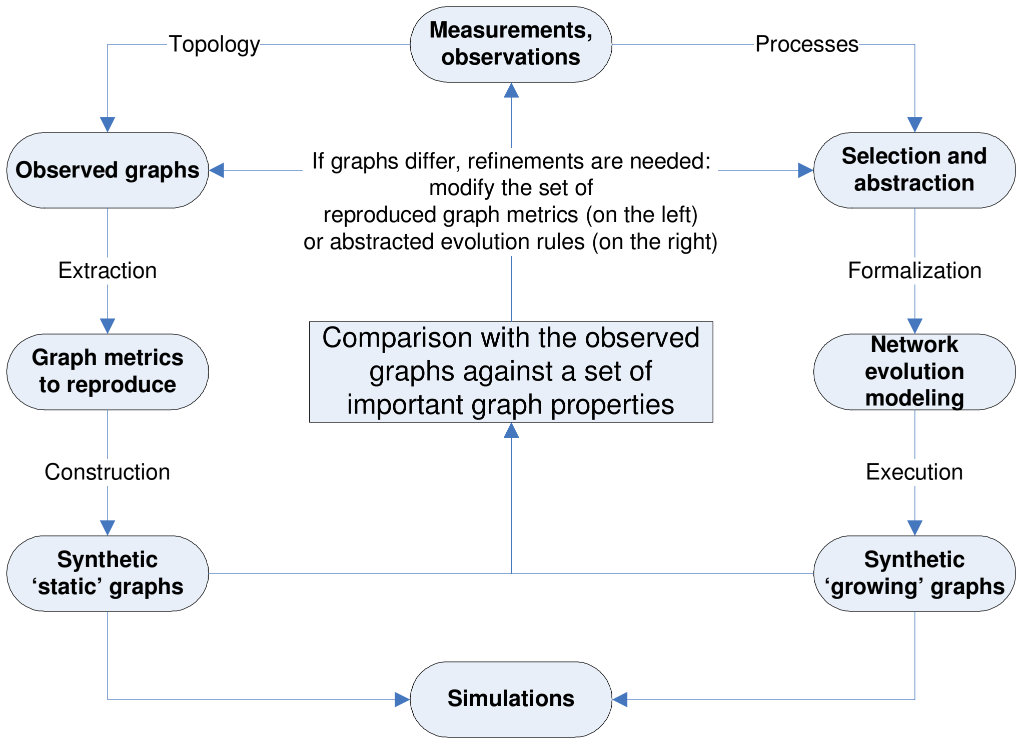

Measurements, observations
Topology
Processes
Observed graphs
Selection and abstraction
If graphs differ, refinements are needed: modify the set of reproduced graph metrics (on the left) or abstracted evolution rules (on the right)
Extraction
Formalization
Graph metrics to reproduce
Comparison with the observed graphs against a set of important graph properties
Network evolution modeling
Construction
Execution
Synthetic 'static' graphs
Synthetic 'growing' graphs
Simulations

Figure 2.1: Methodologies of network topology research.

as shown in the center box of the figure. Researchers have explored and defined many such metrics in the literature. All of these metrics have been shown to directly influence the performance of the network applications and protocols. When two graphs have similar values for these metrics, we say that both the graphs are structurally similar. As we underlined in Chapter 1, we need to deploy and evaluate the application over a variety of random graphs that are structurally similar to real observed topologies. The rationale behind this assumption is that applications display similar behavior when run over structurally similar topologies.

As depicted in the top box in the picture, researchers have undertaken several projects to measure the Internet's topology [44, 48, 55, 82, 95, 92]. From these measurement studies, we can generate graphs that correspond to the Internet's router-level and AS-level topology. The left side of the picture depicts efforts in the research community to build topology generators that output static synthetic graphs that are structurally similar to the observed graphs. In this methodology, we first extract important graph metrics from the observed graph and then develop algorithms to generate random graphs that reproduce these extracted metric values.

The right side of the picture depicts studies to understand the fundamental laws governing the network's evolution. By analyzing these observed graphs over a period of time, researchers attempt to understand the fundamental growth properties and create network models that mimic the evolution process. As in the previous case, the random graphs generated by the evolution models are compared to the original observed graph with respect to the practically important metrics. The closer the match for all the metrics, the more accurate is the evolutionary model under consideration.

In this chapter, we first describe some of the important topology metrics that have been proposed in the literature, as well as the impact these metrics have on application behavior and performance. Next, we summarize efforts undertaken by researchers to generate synthetic graphs that attempt to reproduce the characteristics of real observed networks. We also discuss some of the efforts in the modeling of network evolution.

2.1 Topology Metrics

Given any topology, we need a mechanism to characterize its structure. We also need a technique to compare the structures of two or more graphs. Merely comparing the adjacency matrix of two graphs is not meaningful. The two graphs might be of different sizes, making a comparison of their adjacency matrices impossible. Thus, researchers have proposed various topology metrics in the literature to compare and characterize graph structure. As previously mentioned, all these metrics also directly influence the performance and behavior of various networking applications and protocols.

In this section, we explain the various topology metrics proposed in influential networking papers. We begin with simple metrics that characterize local connectivity in a network. We then move on to metrics that describe global properties of the topology. These latter metrics play a vital role in the performance of network protocols and applications.

Let G be a graph with n nodes and m edges. A variety of complex structures can be represented as graphs. For example, Autonomous Systems in the Internet can be depicted as a graph with each node representing an AS, while an edge connecting the two nodes represents the peering relationship between the two ASes. In a router-level topology, a node represents a router and an edge represents a direct connection between the two routers. Graphs are also widely used in other fields such as biology, economics and sociology. For example, in protein networks, a node represents a protein and an edge represents the interaction between two proteins. While all of the metrics in this section can be applied to all graph types, we discuss the importance and impact of these metrics specifically with respect to the Internet's router and AS-level topologies.

2.1.1 Average degree

Definition. The two most basic graph properties are the **number of nodes** n (also referred to as **graph size**) and the **number of links** m. They define the **average node degree** $\bar{k} = 2m/n$.

Importance. Average degree is the coarsest connectivity characteristic of the topology. Networks with higher $\bar{k}$ are "better-connected" on average and, consequently, are likely to be more robust. Detailed topology characterization based only on the average degree is rather limited, since graphs with the same average node degree can have vastly different structures.

2.1.2 Degree distribution

Definition. Let $n(k)$ be the number of nodes of degree k (k-degree nodes). The **node degree distribution** is the probability that a randomly selected node is k-degree: $P(k) = n(k)/n$ for all distinct degrees present in graph G.

Importance. The degree distribution is the most frequently used topology characteristic. The degree distribution contains more information about connectivity in a given graph than the average degree, since given a specific form of $P(k)$ we can always restore the average degree by $\bar{k} = \sum_{k=1}^{k_{max}} kP(k)$, where k_{max} is the **maximum node degree** in the graph. If the degree distribution in a graph of size n is a power law, $P(k) \sim k^{-\gamma}$, where γ is a positive exponent, then $P(k)$ has a natural cut-off at the **power law maximum degree** [27]: $k_{max}^{PL} = n^{1/(\gamma-1)}$.

2.1.3 Joint degree distribution

While the node degree distribution tells us how many nodes of a given degree are in the network, it fails to provide information on the interconnection between these nodes: given $P(k)$, we still do not know anything about the structure of the neighborhood of the average node of a given degree. The joint degree distribution fills this gap by providing information about 1-hop neighborhoods around a node.

Definition. Let $m(k_1, k_2)$ be the total number of edges connecting nodes of degrees k_1 and k_2. The **joint degree distribution** (JDD), or the **node degree correlation** matrix, is the probability that a randomly selected edge connects k_1- and k_2-degree nodes: $P(k_1, k_2) = \mu(k_1, k_2) \times m(k_1, k_2)/(2m)$, where $\mu(k_1, k_2)$ is 1 if $k_1 = k_2$ and 2 otherwise. The JDD contains more information about the connectivity in a graph than the degree distribution, since given a specific form of $P(k_1, k_2)$ we can always restore both the degree distribution $P(k)$ and average degree $\bar{k}$ by expressions in [27]. A summary statistic of JDD is the **the average neighbor connectivity** $k_{nn}(k) = \sum_{k'=1}^{k_{max}} k'P(k'|k)$. It is simply the average neighbor degree of the average k-degree node. This property shows whether nodes of a given degree preferentially connect to other high- or low-degree nodes. In a full mesh graph, $k_{nn}(k)$ reaches its maximum possible value, $n - 1$. We can further summarize the JDD by a single scalar called **assortativity coefficient** r [69, 26], $r \sim \sum_{k_1,k_2=1}^{k_{max}} k_1k_2(P(k_1, k_2) - k_1k_2P(k_1)P(k_2)/\bar{k}^2)$.

Importance. The assortativity coefficient r, $-1 \leqslant r \leqslant 1$, has direct practical implications. **Disassortative** networks with $r < 0$ have an excess of **radial** links, that is, links connecting nodes of dissimilar degrees. In other words, smaller degree nodes connect to higher degree nodes. Failure or removal of higher degree nodes in these graphs may lead to the smaller degree nodes getting disconnected, consequently leading to one or more partitions in the network. Thus disassortative networks are vulnerable to both random failures and targeted attacks. On a positive note, vertex covers in disassortative graphs are smaller, which is important for applications such as traffic monitoring [11]. The opposite properties apply to **assortative** networks with $r > 0$ that have an excess of **tangential** links, that is, links connecting nodes of similar degrees. Assortative graphs have an excess of tangential links in the core, the connectivity inside the core is richer, and as a consequence, assortative graphs are harder to break (*e.g.* to decompose into similarly-sized disconnected components). Classic examples are social networks [69].

In contrast to the widely studied degree distribution, the network community has recently started recognizing the importance of JDD [103, 46]. In a prominent recent example [54] Li *et al.* define *likelihood* as the product of the degrees each edge in the graph is incident on, summed over all the edges in the graph. Thus likelihood is a non-normalized version of assortativity coefficient. The authors propose to use likelihood as a measure of randomness to differentiate between multiple graphs with the same degree distribution. Such a measure is important for evaluating the amount of order, *e.g.*, engineering design constraints, present in a given topology. A topology with low likelihood is not random; it results from some sophisticated evolution processes involving specific design purposes, whereas a topology with high likelihood is more random and non-engineered.

2.1.4 Clustering

While JDD contains information about the degrees of neighbors for the average k-degree node, it does not tell us how these neighbors interconnect. Clustering partially satisfies this need by providing a measure of how close a node's neighbors are to forming a clique.

Definition. Let $\bar{m}_{nn}(k)$ be the average number of links between the neighbors of k-degree nodes. **Local clustering** is the ratio of this number to the maximum possible number of such links: $C(k) = \bar{m}_{nn}(k)/\binom{k}{2}$. If two neighbors of a node are connected, then these three nodes together form a triangle (3-cycle). Therefore, by definition, local clustering is the average number of 3-cycles involving k-degree nodes. The two summary statistics associated with local clustering are **mean local clustering** $C_{mean} = \sum C(k)P(k)$, which is the average value of $C(k)$, and the **clustering coefficient** C_{coeff}, which is the percentage of 3-cycles among all connected node triplets in the entire graph.

Importance. Clustering expresses local robustness in the graph and thus has practical implications: the higher the local clustering of a node, the more interconnected are its neighbors, thus increasing the path diversity locally around the node. Clustering also dictates the effectiveness of placement policies for a variety of network services. Newman [70] also showed that worm outbreaks spread faster in high-clustered networks, although outbreak sizes are smaller. One can also use clustering for verifying the accuracy of a topology model or generator. Bu *et al.* [12] propose using the clustering coefficient to distinguish various power law topology generators. They show that while most of these algorithms can generate the Internet's AS-level topology with the node degree distribution following a power law, these graphs did not reproduce the clustering coefficient of observed AS-level topologies. Clustering is a basic connectivity characteristic. Therefore, if a model reproduces clustering incorrectly, it is likely to be less accurate for a variety of other graph characteristics.

2.1.5 Rich club connectivity

Definition. Let $\rho = 1 \ldots n$ be the first ρ nodes ordered by their non-increasing degrees in a graph of size n. **Rich club connectivity** (RCC) $\phi(\rho/n)$ is the ratio of the number of links in the subgraph induced by the ρ largest-degree nodes to the maximum possible number of such links $\binom{\rho}{2}$. In other words, the RCC is a measure of how close ρ-induced subgraphs are to cliques.

Importance. The Positive Feedback Preference (PFP) model by Zhou and Mondragon [108] has successfully reproduced a wide spectrum of metrics of their measured AS-level topology by trying to explicitly capture only the following three characteristics: (i) the exact form of the node degree distribution; (ii) the maximum node degree; and (iii) RCC. In a previous work [109], the authors characterize Internet AS topology generators using the RCC. They present evidence that the Internet AS graphs have a rich club, *i.e* a core tier. This agrees well with the intuition that tier 1 ASes peer with each other. They argue that synthetic graph generators that do not model this rich club will necessarily be different from real Internet graphs that exhibit this particular characteristic.

One can show that networks with the same JDDs have the same RCC. The converse is not true, but given a specific form of RCC, one can fully describe all possible JDDs that would yield the specified RCC.

2.1.6 Distance

Definition. The shortest path length distribution or simply the **distance distribution** $d(x)$ is the probability that a random pair of nodes are at a distance x hops from each other. To define the distance distribution more formally, let $n_i(x)$ be the number of nodes at distance x from node i. We assume that $n_i(0) = 1$ (node i is at distance 0 from itself). Note that $n_i(1) = \deg(i)$. The distance distribution for node i is $d_i(x) = n_i(x)/n$ and its average distance $\bar{d_i} = \sum_{x=0}^{D} x d_i(x)$.

The **distance distribution** for the entire graph is $d(x) = \sum_{i=1}^{n} d_i(x)/n$; the **average distance** is $\bar{d} = \sum_{x=0}^{D} x d(x)$. σ is the **standard deviation** of the average distance and is typically called the distance distribution width since the distance distribution observed in the Internet (and in many other networks) has a characteristic Gaussian-like form. The maximum distance D is called the graph **diameter**.

Importance. Distance distribution is important for many applications, the most prominent being routing. A distance-based locality-sensitive approach [76] is the root of most modern routing algorithms. As shown in [51], performance parameters of these algorithms depend mostly on the distance distribution. In particular, short average distance and narrow distance distribution width break the efficiency of traditional hierarchical routing. They are among the root causes of interdomain routing scalability issues in the Internet today.

Distance distribution also plays a vital role in robustness of the network to worms. Worms can quickly contaminate a network that has small distances between nodes. Topology models that accurately reproduce observed distance distributions will benefit researchers developing techniques to quarantine the network from worms [86].

We note that *expansion*, identified in [94] as a critical metric for topology comparison analysis, is a renormalized version of the distance distribution. Expansion $E(h)$ is the average fraction of nodes in the graph that fall within a ball of radius h centered at a node in the topology [94]. For a node v in the graph, Tangmunarunkit *et al.* first compute the number of nodes that can be reached within h hops. They compute the above value for each node in the graph, average the result and then normalize it by the total number of nodes in the graph. Thus, by renormalizing the distance distribution, we get the expansion of the graph.

2.1.7 Betweenness

Although the average distance is a good node centrality measure–intuitively, nodes with smaller average distances are closer to the graph "center,"–the most commonly used measure of centrality is betweenness. It is applicable not only to nodes, but also to links.

Definition. Betweenness measures the number of shortest paths passing through a node or link and, thus, estimates the potential traffic load on this node/link assuming uniformly distributed traffic following shortest paths. Let σ_{ij} be the number of shortest paths between nodes i and j and let l be either a node *or* link. Let $\sigma_{ij}(l)$ be the number of shortest paths between i and j going through node (or link) l. Its **be-**

tweenness is $B_l = \sum_{ij} \sigma_{ij}(l)/\sigma_{ij}$. We note here that there might be multiple shortest paths of the same length between nodes i and j, all of which are considered in the betweenness calculation. The maximum possible value for node and link betweenness is $n(n-1)$ [22], therefore in order to compare betweenness in graphs of different sizes, we normalize it by $n(n-1)$.

Importance. Betweenness is important for traffic engineering applications that try to estimate potential traffic load on nodes/links and potential congestion points in a given topology. Betweenness is also critical for evaluating the accuracy of sampling the topology using tree-like probes (*e.g.* skitter and BGP). As shown in [22], the broader the betweenness distribution, the higher the statistical accuracy of the sampled graph.

We note that *link value*, used in [94] is directly related to betweenness. Tangmunarunkit *et al.* use link value to characterize hierarchy in the graph. They argue that if some level of hierarchy exists in the topology, then certain links will be used more often than the others. Link value is very similar to betweenness as the authors use shortest-path routing between all source-destination pairs to compute the number of times a link is traversed.

Finally, we note that *router utilization* [54] used to measure network performance is also directly related to betweenness. Router utilization computes the total traffic flow through each router in the topology. This metric computes how many times a router is traversed, and is thus essentially node betweenness.

2.1.8 Spectrum

Definition. Let $\mathcal{A}$ be the adjacency matrix of a graph. This $n \times n$ matrix is constructed by setting the value of its element $a_{ij} = a_{ji} = 1$ if there is a link between nodes i and j. All other elements have value 0. The scalar λ and vector v are the eigenvalue and eigenvector respectively of $\mathcal{A}$ if $\mathcal{A}v = \lambda v$. The **spectrum** of a graph is the set of eigenvalues of its adjacency matrix.

Another closely related definition of the graph spectrum is the spectrum of the eigenvalues of its Laplacian, $\mathcal{L} = \mathcal{T}^{-1/2}(\mathcal{T} - \mathcal{A})\mathcal{T}^{-1/2}$, where $\mathcal{T}$ is the diagonal matrix with t_{ii} equal to the degree of node i. This definition is a normalized version of the original definition, in the sense that for any graph, all the eigenvalues of its Laplacian are located between 0 and 2.

Importance. Spectrum is one of the most important *global* characteristics of the topology. Spectrum yields tight bounds for a wide range of critical graph characteristics [19], such as distance-related parameters, expansion properties, and values related to separator problems estimating graph resilience under node/link removal. The largest eigenvalues are particularly important. Most networks with high values for these largest eigenvalues have small diameter, expand faster, and are more robust, and thus less susceptible to random failures and targeted attacks.

Two specific examples of significant metrics proposed by networking topology researchers are directly related to spectrum. First, Tangmunarunkit *et al.* [94] defined network *resilience*, one of the three metrics critical for their topology comparison analysis, as a measure of network robustness under link removal, which equals the minimum balanced cut size of a graph. By this definition, resilience is related to spectrum since the graph's largest eigenvalues provide bounds on network robustness with respect to both link *and* node removals [19].

Second, Li *et al.* [54] define network *performance*, one of the two metrics critical for their HOT argument, as the maximum traffic throughput of the network. By this definition, performance is related to spectrum since it is essentially the network conductance [36] which can be tightly estimated by the gap between the first and second largest eigenvalues [19].

Beyond its significance for network robustness and performance, the graph's largest eigenvalues are important for traffic engineering purposes since graphs with larger eigenvalues have, in general, more node- and link-disjoint paths to choose from. The spectral analysis of graphs is a powerful tool for detailed investigation of network structure, such as discovering clusters of highly interconnected nodes [100], and possibly revealing the hierarchy of ASes in the Internet [37].

So far, we have described the important metrics that are used in a wide variety of topological studies to compare and characterize the structure of different graphs. In the next section, we briefly summarize various topology generation efforts in the networking community.

2.2 History of Topology Generators

Synthetic graphs resembling actual Internet-like topologies are critical to a wide variety of studies including evaluating network protocols and applications, understanding and optimizing routing algorithms, network management and traffic engineering. For the past decade or more, researchers have been focusing on building topology generators that attempt to produce representative Internet-like graphs. Most of these generators focus on generating random graphs that reproduce specific aspects of real observed networks such as the observed hierarchy amongst the nodes or the observed degree distribution.

One of the earliest topology models was proposed by Waxman [101], and is based on the classical Erdős-Rényi random graphs [30]. In the Waxman model, an edge from node u to v is added with the probability given by $P(u, v) = \alpha e^{-d/(\beta L)}$, where $0 < \alpha, \beta \leq 1$ are parameters of the model and L represents the maximum distance between any two nodes in the graph. While the Waxman model accounts for some network characteristics such as node placement and geographic distance between the nodes, it fails to capture the hierarchy believed to exist in the Internet and was abandoned in favor of other models such as GT-ITM [106] and Tiers [25].

GT-ITM incorporates the Transit-Stub (TS) model to reproduce the hierarchical structure of the Internet. The TS model has two kinds of nodes - transit and stub. Transit nodes represent the backbone of the Internet. The stub nodes, representing the edge systems in the Internet, connect to the transit nodes. This specific form of connectivity gives rise to a tiered structure in the graph.

The Tiers model is another example of generators that model structural hierarchy. Tiers reproduces a three-level hierarchy and models Wide-Area, Metropolitan-Area and Local-Area networks within the Internet. Seminal work [31] in 1999 presented evidence that the degree distribution of Internet ASes followed a power law. The widely held belief that an organized hierarchy existed among the ASes in the Internet was also disproved by researchers [94] who showed that topologies derived from structural generators such as GT-ITM and Tiers that incorporated hierarchies of AS tiers did not have much in common with topologies obtained from real observed data. The smooth power law degree distribution indicates that there are no organized tiers among ASes. The power law distribution also implies substantial variability associated with degrees of individual nodes. Later, topology generators such as PLRG [2], Inet [103], and BRITE [61] also focused on reproducing the observed power law degree distribution.

The popular BA model [5] incorporates growth of the network using preferential attachment. In this model, the network is grown by adding a new node and the newly added node connects to the existing nodes with a probability that is biased towards degrees of the nodes. In other words, newly added nodes connect to existing high degree nodes in the graph, leading to the 'rich get richer' effect. The degree distribution resulting from the BA model is scale free and follows the power law.

PLRG relies on generating random graphs where the node degree distribution follows the power law. The input to the PLRG model includes the number of nodes, n, in the new topology and the value of the power law exponent γ. Nodes in the graph are first assigned degrees with a probability given by $P(k) \sim k^{-\gamma}$, where k is the degree. Nodes are then randomly connected to form a graph, such that their assigned degree is respected.

Inet is specific to the Internet's AS-level topology and is also based on the preferential attachment model. Inet also assigns degrees to nodes based on the power law distribution and ensures that the resulting graph is connected by first creating a spanning tree using nodes of degree greater than two. Nodes with degrees one and two are connected to nodes in the spanning tree and new links are added to the graph according to various rules to ensure that degree of every node is fulfilled.

While Inet, BRITE and PLRG are commonly used in a wide variety of networking research, the graphs from these generators do not match real observed topologies with respect to the wide range of metrics that we discussed in Section 2.1. Additionally, the networking community is focusing on another approach to modeling topologies, especially router-level topologies. Li *et al.* [54] consider router capacity constraints as well as likelihood to model router-level topologies and advocate understanding the evolution of networks in order to accurately model router-level graphs. For AS-level topologies, recent work [17] considers the technological, economic, and political considerations behind whether pairs of ASes peer with one another. Thus, this and related efforts consider the driving evolutionary forces behind the growth of particular topologies. However understanding the evolutionary forces for a network's growth is a difficult problem. To date, researchers have not had much success in inferring these evolutionary principles.

Unfortunately, the current *state-of-the-art* in topology generation fails to meet our requirements specified in Chapter 1. Ideally, one would like a topology generator to produce random graphs that capture important metrics such as distance distribution, betweenness, clustering, spectrum, *etc.* of real measured topologies. The generator must also have the ability to output random graphs of a variety of sizes, as well as annotate the graphs with information that includes AS membership and link latency. All the topology models and generators discussed in this section are unable to reproduce important metric values of real observed topologies such as spectrum and distance distribution. Further, they include either no annotations or use simple heuristics known not to reflect Internet characteristics. While these generators can generate graphs of a variety of sizes, these different sized graphs again are structurally different from the observed topologies.

A first step to building a topology generator that can produce graphs structurally similar to observed Internet topologies is to understand the structural characteristics of these topologies. One needs to know the values of the mostly commonly used graph metrics. Consequently, we study Internet AS-level topologies extracted from various data sources in the next chapter. In addition to analyzing these graphs, we also examine the interplay amongst the various metrics. The question we attempt to answer is: Can we identify a single metric that is fundamental to the graph structure? Is there any one property that also automatically captures all the other graph metrics? Identifying such a metric will simplify our task in building a topology generator as there will be fewer graph characteristics to specify as inputs to the generator.

2.3 Acknowledgments

Chapter 2, in part, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Computer Communications Review (CCR), January 2006, Mahadevan, Priya; Krioukov, Dmitri; Fomenkov, Marina; Huffaker, Bradley; Dimitropoulos, Xenofontas; claffy, kc; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Understanding Interplay Among Metrics

In the previous chapter, we described some of the most important and commonly used metrics in topological research. These metrics are sufficiently diverse and derived from a wide range of studies. More importantly, these metrics directly influence the performance of network protocols and services. For example, convergence of routing protocols is dependent on the distance distribution and average distance of the graph. Studies analyzing the performance of current routing protocols as well as developing new routing architectures must be evaluated over Internet-like topologies that have similar distance distribution as real measured Internet graphs. If this is not the case, any conclusion reached about the routing architecture beforehand may not be valid if the same routing protocol is deployed over the Internet.

Past efforts to generate random graphs that match observed topologies with respect to these metrics have not met with much success. Given our goal to build a generator capable to producing random graphs that match observed networks with respect to all the proposed metrics (and any future metric), one of our first tasks is to understand the structure of Internet topologies. Computing the values for all the important metrics and studying the interplay among the metrics will provide us a better intuition on how to generate random graphs that match the observed graphs with respect to these metrics.

We analyze the structure of the Internet's AS-level topology as these measurements are more readily available than router-level graphs. There are a number of sources of AS topology data, obtained using different methodologies that yield substantially different topological views of the Internet's ASes. We, therefore, extracted our AS-level topologies from three commonly-used data sources: (1) traceroute measurements [48]; (2) BGP [83]; and (3) the WHOIS database [44]. We characterized these topologies with respect to the metrics discussed in Section 2.1. In the rest of this chapter, we discuss our methodology for constructing these AS graphs; further we reveal the peculiarities of each data source and the resulting interplay between artifacts of data collection and the key properties of the derived graphs.

While we study AS-level topologies, our choice of this specific type of topology does not impact our understanding of the interplay amongst the topology metrics. The key insights from this study that we summarize at the end of this chapter, laid the foundation for developing Orbis, our generic network topology generator, that can output any topology including AS-, router-, and even enterprise-level graphs.

3.1 Constructing AS Graphs

We used the following data sources to construct AS-level graphs of the Internet: traceroute measurements, BGP data, and the WHOIS database.

BGP (Border Gateway Protocol) [82] is the protocol for routing among ASes in the Internet. RouteViews [83] collects BGP routing tables using 7 collectors, 5 of which are located in the USA, 1 in the UK and 1 in Japan. Each collector has a number of globally placed peers (or vantage points) that collect BGP messages from which we can infer the AS topology.

RouteViews archives both static snapshots of the BGP routing tables and dynamic BGP data in the form of BGP message dumps (updates and withdrawals). Therefore, we derive two types of graphs from the BGP data for the same month of March 2004: one from the static tables (**BGP tables**) and one from the updates

(**BGP updates**). We create the BGP tables graph using data from the collector *route-views.oregon-ix.net* as it gathers data from the largest number of peers—68. For the BGP updates graph, we choose the collector *route-views2.oregon-ix.net*, which uses 40 peers to collect data, since at the time of this research *route-views.oregon-ix.net* did not collect BGP updates. The data contains AS-sets [82], that is, lists of ASes with unknown interconnection structures. For both BGP tables and updates graph, we discard AS-sets and private ASes [41] because they create false links in the graph. We then merge the 31 daily graphs of March 2004 into one graph for each BGP data source.

Table 3.1: Comparison of graphs built from different data sources. The baseline graph G_A is the BGP tables graph. Graph G_B is the other graph listed in the first row.

	BGP updates	skitter	WHOIS
Number of nodes in both G_A and G_B ($\lvert V_A \bigcap V_B \rvert$)	17,349	9,203	5,583
Number of nodes in G_A but not in G_B ($\lvert V_A \setminus V_B \rvert$)	97	8,243	11,863
Number of nodes in G_B but not in G_A ($\lvert V_B \setminus V_A \rvert$)	68	1	1,902
Number of edges in both G_A and G_B ($\lvert E_A \bigcap E_B \rvert$)	38,543	17,407	12,335
Number of edges in G_A but not in G_B ($\lvert E_A \setminus E_B \rvert$)	2,262	23,398	28,470
Number of edges in G_B but not in G_A ($\lvert E_B \setminus E_A \rvert$)	3,941	11,552	44,614

We show the overlap statistics of our graphs in Table 3.1. This table uses the BGP-table graph as the baseline and compares it with the BGP-updates graph in the first column. Between the two BGP-derived graphs, we note the similarity in the sets of their constituent nodes and links. Given minor differences between node and link sets of the BGP table- and update-derived topologies, we find the graph metric values calculated for these two topologies to be nearly identical for all characteristics that we consider. Therefore, we present characteristics of the static BGP table graph only and refer to it as the *BGP graph*.

Traceroute [97] captures the sequence of IP hops along the forward path from the source to a given destination by sending either UDP or ICMP probe packets to the destination. CAIDA has developed a tool, *skitter* [48], to collect continuous traceroute-based Internet topology measurements. *skitter* maintains a target destination list that comprises approximately one million IPv4 addresses. CAIDA collects these addresses from various sources such as existing destination lists, intermediate addresses in *skitter* traces, users accessing CAIDA website. The goal is to find one responding IP address within each routable $/24$ segment, to provide representative coverage of the routable IPv4 address space. The destination list is updated once every 8 to 12 months to ensure the addresses stay current and to maximize reachability. Skitter uses 25 monitors (traceroute sources), strategically placed in the global Internet: 15 monitors in North America, 6 monitors in Europe, 3 monitors in Japan and 1 in New Zealand. Each monitor sends probe packets to destinations in the target list and gathers the corresponding IP paths.

Using the core BGP tables provided by RouteViews, CAIDA maps the IP addresses in the gathered IP paths to AS numbers, constructs the resulting AS-level topology graphs on a daily basis and makes these graphs publicly available at [13]. For this study, we started with daily graphs for each day of March 2004, *i.e.*, 31 daily graphs. Mapping *skitter*-observed IP addresses to AS numbers involves potential distortion, *e.g.,* due to multi-origin ASes, that is, the same prefixes advertised by multiple ASes [58], AS-sets, and private ASes. Both multi-origin ASes and AS-sets create ambiguous mappings between IP addresses and ASes, hence we filtered them from each graph. In addition, we filtered private ASes as they create false links. Unresolved IP hops in the traceroute data give rise to indirect links [13], which we discarded. The total discarded and filtered links constitute approximately 5 percent of all links in the initial graph. We then merged all the daily graphs to form one graph, which we call the *skitter graph.*

Comparing the skitter graph with the BGP graph, we notice that there is exactly 1 node seen in the skitter but not in the BGP graph. This node is AS2277 (Ecaunet). Since we use BGP table dumps to map IP addresses to AS numbers in constructing the skitter graph, we expect the number of nodes present in the skitter but not in the BGP to be 0. The one node difference occurs because different BGP table dumps were used to construct the BGP table graph and to perform IP-to-AS mapping in the skitter graph on the day when *skitter* observed this IP address in its traces.

WHOIS [44] is a collection of databases with AS peering information useful to network operators. These databases are manually maintained with little requirements for timely updates of registered information. Of the public WHOIS databases, RIPE's WHOIS database contains the most reliable current topological information, although it covers primarily European Internet infrastructure [88, 16].

We obtained the RIPE WHOIS database dump for April 07, 2004. We were interested in the following types of records:

```
aut-num:  ASx
import:   from ASy
export:   to ASz
```

This record indicated the presence of links between ASx-ASy and ASx-ASz. We constructed an AS-level graph (here after referred to as *WHOIS graph*) from these records and excluded ASes that did not appear in the `aut-num` lines. Such ASes are external to the database and we cannot correctly estimate their topological properties, *e.g.*, node degree. We also filtered private ASes.

Both Table 3.1 (column 3) and the topology metrics we considered in Section 2.1 show that the WHOIS topology differs significantly from the other two graphs. Thus, the following question arises: Can we explain the difference by the fact that the WHOIS graph contains only a part of the Internet, namely European ASes? To answer this question we perform the following experiment. We consider the BGP tables and WHOIS topologies narrowed to the set of nodes present both in BGP tables and WHOIS, *i.e.*, the 5,583 nodes present in the intersection of BGP tables and WHOIS graphs (Table 3.1) and compute the various topological characteristics for these reduced graphs. We then compare the properties of the original BGP and WHOIS graphs to their reduced graphs respectively and find that the reduced graphs preserve the full set of the properly normalized topological properties of the original graphs. In other words, the reduced BGP graph, consisting only of ASes found in the intersection of WHOIS and the original BGP graph, has topological characteristics similar to the original BGP graph, while the reduced WHOIS graph has characteristics similar to the original WHOIS graph. Therefore, the differences between full BGP and WHOIS topologies are likely due to dissimilar intrinsic properties of their originating data sources, and not due to geographical biases in sampling the Internet.

Based on the very method of their construction, the three graphs in our study reveal different sides of the actual Internet AS-level topology. The skitter graph closely reflects the topology of actual Internet traffic flows, *i.e.*, the data plane. The BGP graph reveals the topology seen by the routing system, *i.e.*, the control plane. The BGP graph does not reflect how traffic actually travels toward a destination network. The WHOIS graph reflects the topology extracted from manually maintained databases, *i.e.*, the management plane.

3.1.1 Limitations and validity of our results

All our data sources have some inaccuracies arising from their collection methodology. Since *skitter* methodology relies on answers to ICMP requests, ICMP filtering at intermediate hops adds some inaccuracy to the data. *skitter* also fails to receive ICMP replies in the address blocks advertised by some small ASes. The BGP graph depends on routing table exchanges, and not all peer ASes advertise all their peering relationships; therefore the BGP graph tends to miss these unadvertised links. Various misconfigurations, *e.g.,* announcement of prefixes not owned by an AS, *etc.,* are some of the other causes of errors with the BGP data. The manually maintained WHOIS database is most likely to contain stale or inaccurate information [88]. In fact, the WHOIS graph is likely to reflect unintentional or even intentional over-reporting of peering relationships by some providers. There have been reports about some ISPs entering inaccurate information in the WHOIS database to increase their “importance” in the Internet hierarchy [88].

We limit our data collection to a single month for obtaining the AS-level graphs in this chapter. We believe that the interdependencies between different metrics will hold for data gathered over various periods of time and are not an artifact of the current Internet or our sampling period. We study historical AS-level data for skitter and BGP for the period from 2000 to 2005 and find that indeed the metric values have more or less stayed invariant during this period. We present detailed results from historic AS-level graphs in Chapter 5.

When processing each of our data sets to create the desired graph, we make choices while dealing with ambiguities and errors in the raw data. One example is the detection of “false” links created by route changes in traceroute data. The processing we apply may potentially cause ambiguity in our final graphs.

While all three sources of topology data contain a number of sources of errors and cannot be considered perfect representations of true AS-level interconnectivity, the results of a number of recent studies indicate that the available data is a reasonable approximation of AS topology. The presence of global and strategically located vantage points for both BGP and skitter graphs as well as the careful choice of destinations used by *skitter* lend credibility to traceroute-based measurement studies. There have been some doubts about the validity of topologies obtained from traceroute measurements. Specifically, Lakhina *et al.* [53] numerically explored sampling biases arising from traceroute measurements and found that such traceroute-sampled graphs of the Internet yield insufficient evidence for characterizing the actual underlying Internet topology. However, Dall’Asta *et al.* [22] convincingly refute their conclusions by showing that various traceroute exploration strategies provide sampled distributions with enough signatures to statistically distinguish between different topologies. The authors also argue that real mapping experiments observe genuine features of the Internet, rather than artifacts.

While each of our data sources has its own limitations, an open question is which data source most closely matches actual Internet AS topology, given that each graph approximates a different view of the Internet looking at the data (skitter), control (BGP), and management (WHOIS) planes. Our study does not aim to answer this important question. Rather we focus on the structure of these graphs as revealed by the various topology metrics. Our study shows that despite their different collection methodologies, all the metric values for these three graphs is determined by its average degree ($\bar{k}$) and assortativity coefficient (r), both of which are coarse summary statistics of the *joint degree distribution* (JDD). As will be evident from the results, of all the metrics we considered, the JDD appears to play a central role in determining a wide range of other topological properties.

3.2 Metric Results for the Three Topologies

In this section, we quantitatively analyze differences between the three graphs in terms of various topology metrics. In Table 3.2, we summarize the important metric values for all three graphs – skitter, BGP and WHOIS. The table shows that the skitter and BGP graph are qualitatively similar to each other, while the WHOIS graph is the most different. We now proceed to discuss how each of these graphs differ with respect to the metrics. At the same time, we also discuss the interplay amongst the various metrics.

3.2.1 Average degree

The WHOIS graph has the smallest number of nodes, but its average degree is almost three times larger than that of BGP, and ~ 2.5 times larger than that of skitter (Table 3.2). In other words, WHOIS contains substantially more links, both in the absolute (m) and relative ($\bar{k}$) senses, than any other data source, although the credibility of these links is lowest. The chief reason for WHOIS graph's high average degree lies in its measurement specifics: we have information from every node's perspective in the database, while skitter and BGP graphs are obtained by sampling using tree-like explorations of the Internet's ASes.

We also observe that the number of nodes in the BGP graph is almost twice the number of nodes in skitter. This again can be explained by the measurement techniques of the two data sources: *skitter* relies on responses to ICMP requests sent to IP addresses on its target list of destinations and it may not have any targets in the address blocks advertised by some small ASes. As a result, *skitter* does not see these ASes. The BGP

Table 3.2: Summary of important metrics for skitter, BGP and WHOIS topologies

		skitter	BGP	WHOIS
Average degree	Number of nodes (n)	9,204	17,446	7,485
	Number of edges (m)	28,959	40,805	56,949
	Avg node degree $(\bar{k})$	6.29	4.68	15.22
Degree distribution	Max node degree (k_{max})	2,070	2,498	1,079
	Power-law max degree (k_{max}^{PL})	1,448	4,546	-
	Exponent of $P(k)$ $(-\gamma)$	2.25	2.16	-
Joint degree distribution	Avg neighbor degree $(\bar{k}_{nn}/(n-1))$	0.05	0.03	0.02
	Exponent of $k_{nn}(k)$ $(-\gamma_{nn})$	1.49	1.45	-
	Assortative coefficient (r)	-0.24	-0.19	-0.04
Clustering	Mean clustering (C_{mean})	0.46	0.29	0.49
	Clustering coefficient (C_{coeff})	0.03	0.02	0.31
	Exponent of $C(k)$ $(-\gamma_C)$	0.33	0.34	-
Rich club	Exponent of $\phi(\rho/n)$ $(-\gamma_{rc})$	1.48	1.45	1.69
Distance	Avg distance $(\bar{d})$	3.12	3.69	3.54
	Std deviation of distance (σ)	0.63	0.87	0.80
	Exponent of $d(k)$ $(-\gamma_d)$	0.07	0.07	0.09
Betweenness	Avg node betweenness $(\bar{B}_{node}/(n(n-1)))$	$11 \cdot 10^{-5}$	$7.7 \cdot 10^{-5}$	$17 \cdot 10^{-5}$
	Exponent of $B(k)$ (γ_B)	1.35	1.17	-
	Avg edge betweenness $(\bar{B}_{edge}/(n(n-1)))$	$5.37 \cdot 10^{-5}$	$4.51 \cdot 10^{-5}$	$3.10 \cdot 10^{-5}$
Spectrum	Largest eigenvalue	79.53	73.06	150.86
	Second largest eigenvalue	-53.32	-55.13	68.63
	Third largest eigenvalue	36.40	53.54	62.03

routing tables however contain information about these ASes and thus these nodes are observed in the BGP graph. The extra ASes in the BGP dataset are mostly low-degree (cf. Section 3.2.2) and therefore the BGP graph has a lower average degree than skitter.

Graphs ordered by increasing average degree $\bar{k}$ are BGP, skitter, WHOIS. We call this order the **$\bar{\mathbf{k}}$-order**.

3.2.2 Degree distribution

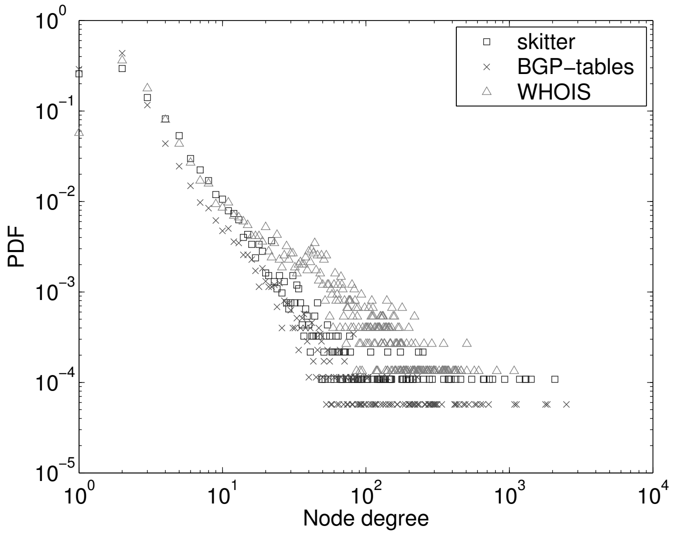
As expected, the degree distribution PDFs and CCDFs in Figure 3.1 are in the $\bar{k}$-order (BGP $<$ skitter $<$ WHOIS) for a wide range of node degrees.

Comparing the observed maximum node degrees k_{max} with those predicted by the power law k_{max}^{PL} in Table 3.2, we conclude that skitter is closest to power law. The power-law approximation for the BGP graph is less accurate. The WHOIS graph has an excess of medium-degree nodes and its node degree distribution does not follow a power law at all. Thus, generators based on reproducing the power-law degree distribution such as PLRG [2] will be unable to produce random graphs structurally similar to the WHOIS topology.

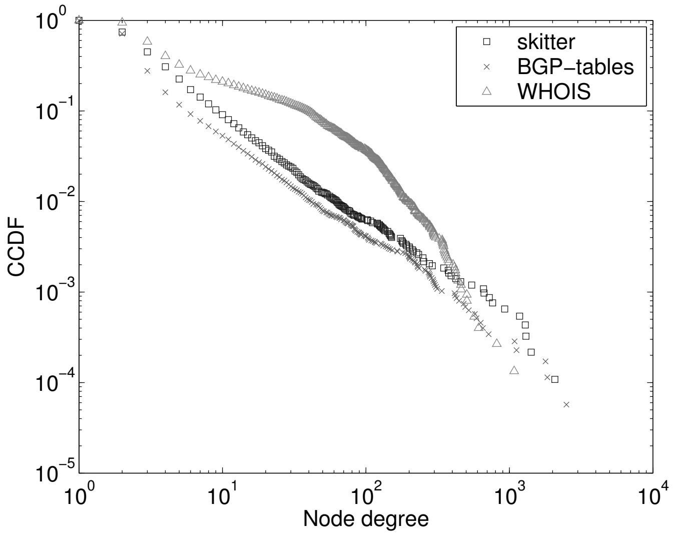
There are fewer 1-degree nodes than 2-degree nodes in all the graphs (Figure 3.1(a)). This effect is due to the AS number assignment policies [41] allowing a customer to have an AS number only if it has multiple providers. If these policies were strictly enforced and if there were no measurement inaccuracies, then the minimum observed AS degree would be 2.

3.2.3 Joint degree distribution

The JDD, or the node degree correlation matrix, is the probability that a randomly selected edge connects k_1- and k_2-degree nodes: $P(k_1, k_2) = \mu(k_1, k_2) \times m(k_1, k_2)/(2m)$, where $\mu(k_1, k_2)$ is 1 if $k_1 = k_2$ and 2 otherwise. We can summarize this matrix by a single scalar called assortativity coefficient r [69, 26]. Computing the assortativity coefficient for our topologies, we find that all the three Internet graphs built



(a) PDF



(b) CCDF

Figure 3.1: Node degree distributions $\mathbf{P}(\mathbf{k})$ for skitter, BGP and WHOIS graphs.

from our data sources are disassortative ($r < 0$) as seen in Table 3.2. We call the order of graphs with decreasing assortativity coefficient r—WHOIS, BGP, skitter—the **r-order**.

We can explain the r-order in terms of differing topology measurement methodologies. First, we notice that both skitter and BGP graphs are results of *tree-like* explorations of the network topology, meaning that we can roughly approximate these graphs by a union of spanning trees rooted at, respectively, *skitter* monitors or BGP data collection points. As such, both these methods are likely to discover more radial links connecting numerous low-degree nodes, *i.e.,* small ASes, to high-degree nodes, *i.e.,* large ISP ASes, where the monitors are located. At the same time, these measurements fail to detect some tangential links interconnecting medium-to-low degree nodes since many of these links belong to none of the spanning trees rooted at the vantage points in the core. In contrast, WHOIS data contains abundant medium-degree tangential links because it relies on operators to report *all* the links attached to a given AS, *i.e.,* a source of a WHOIS record. This excess of tangential links in WHOIS is thus responsible for its much higher assortativity. Second, we explain that the BGP graph has a slightly higher assortativity than the skitter graph. As discussed in Section 3.2.2, the BGP graph contains the tangential links between low-degree nodes that traceroute probes of *skitter* miss since these links are typically the backup links to smaller secondary providers, while *skitter*'s ICMP packets tend to follow the primary paths to larger primary providers. This small excess of tangential links is responsible for a slightly higher assortativity of the BGP graph compared to skitter.

The interplay between $\bar{k}$- and r-orders underlies Figure 3.2, where we plot the average neighbor connectivity functions for the three graphs. For uniform graph comparison we plot normalized values of the average neighbor degree $k_{nn}(k)/(n-1)$. In the case of skitter and BGP, $k_{nn}(k)$ can be approximated by a power law with the corresponding exponents γ_{nn} in Table 3.2.

Skitter has the largest excess of radial links that connect low-degree nodes (customers ASes) to high-degree nodes (large provider ASes). The highest relative number of radial links is responsible for skitter's highest average degree of the neighbors of

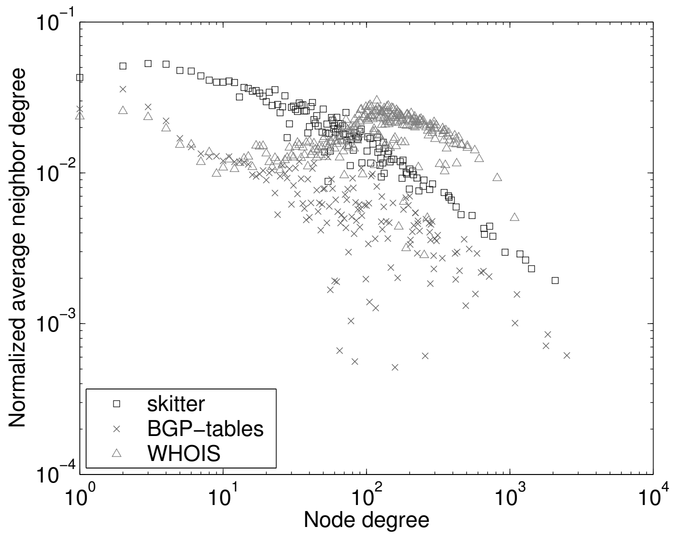


Figure 3.2: Normalized average neighbor connectivity $\mathbf{k_{nn}(k)/(n-1)}$.

low-degree nodes: in Figure 3.2, skitter is at the top in the area of low degrees, while BGP is below and WHOIS is at the bottom (r-order). On the other hand, the greatest proportion of tangential links between ASes of similar degrees in the WHOIS graph contributes to connectivity of neighbors of high-degree nodes; therefore the WHOIS graph is at the top for high-degree nodes ($\bar{k}$-order).

3.2.4 Clustering

The two summary statistics associated with local clustering are **mean local clustering** $C_{mean} = \sum C(k)P(k)$, which is the average value of $C(k)$, and the **clustering coefficient** C_{coeff}, which is the percentage of 3-cycles among all connected node triplets in the entire graph. We first observe that the clustering average values C_{mean} in Table 3.2 are in the $\bar{k}$-order, which is expected: clustering increases with increase in number of links. The values of C_{mean} are almost equal for skitter and WHOIS, but the clustering coefficient C_{coeff} is 15 times larger for WHOIS than for skitter. As shown in [90], orders of magnitude difference between C_{mean} and C_{coeff} is intrinsic to highly disassortative networks and is a consequence of strong degree correlations (JDD) necessarily present in such networks. Skitter, which is highly disassortative, therefore

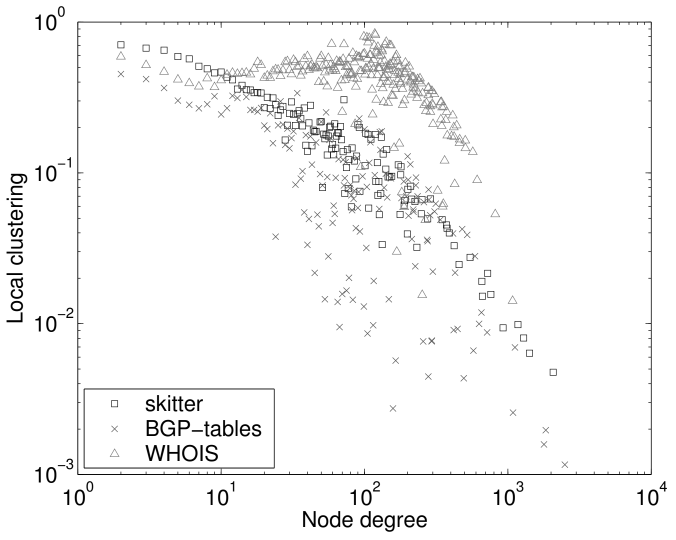


Figure 3.3: Local clustering $\mathbf{C}(\mathbf{k})$.

expectedly exhibits this huge difference between its C_{mean} and C_{coeff} values. The BGP graph, which is again disassortative, also displays this order of magnitude difference between its C_{mean} and C_{coeff} values. The WHOIS graph, on the other hand, is much more random as explained by its assortativity coefficient value of -0.04, and therefore its C_{mean} and C_{coeff} are fairly close to each other. We stress that the JDD plays an important role in dictating the values for C_{mean} and C_{coeff} for all three graphs.

Similar to $k_{nn}(k)$, the interplay between $\bar{k}$- and r-orders explains Figure 3.3, where we plot local clustering as a function of node degree $C(k)$. Skitter's clustering is the highest amongst the three graphs for low-degree nodes because this graph is most disassortative. The links adjacent to low-degree nodes are most likely to lead to high-degree nodes, the latter being interconnected with a high probability. The WHOIS graph exhibits the highest values for clustering for high-degree nodes since this graph has the highest average connectivity (largest $\bar{k}$). The neighbors of high-degree nodes are interconnected to a greater extent, resulting in higher clustering for such nodes. Similar to $k_{nn}(k)$, $C(k)$ also can be approximated by a power law for skitter and BGP graphs (exponents γ_C in Table 3.2).

3.2.5 Rich club connectivity

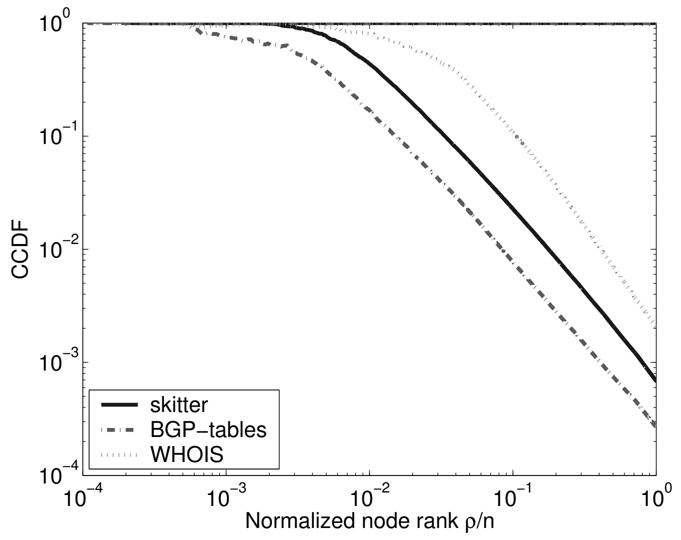


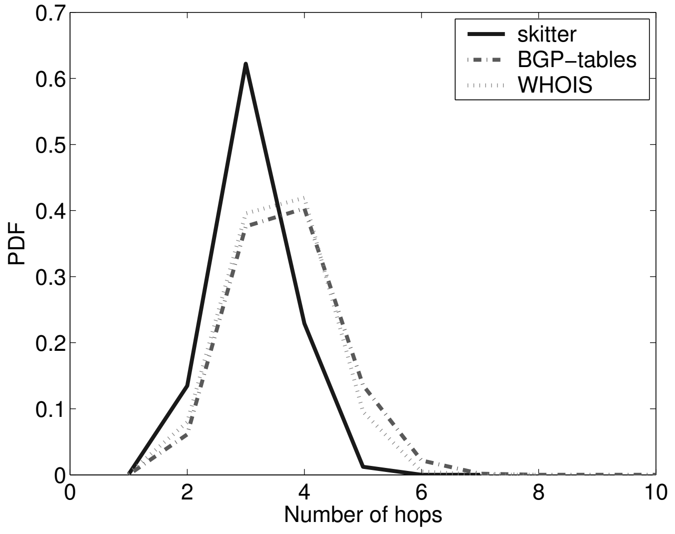
Figure 3.4: Rich club connectivity $\phi(\rho/\mathbf{n})$.

The RCC $\phi(\rho/n)$ is the ratio of the number of links in the subgraph induced by the ρ largest-degree nodes to the maximum possible number of such links $\binom{\rho}{2}$. As expected, the values of $\phi(\rho/n)$ in Figure 3.4 are in the $\bar{k}$-order with WHOIS at the top: more links result in denser cliques. RCC exhibits clean power laws for all three graphs in the area of medium and large ρ/n. The values of the power-law exponents γ_{rc} in Table 3.2 result from fitting $\phi(\rho/n)$ with power laws for 90% of the nodes, $0.1 \leqslant \rho/n \leqslant 1$.

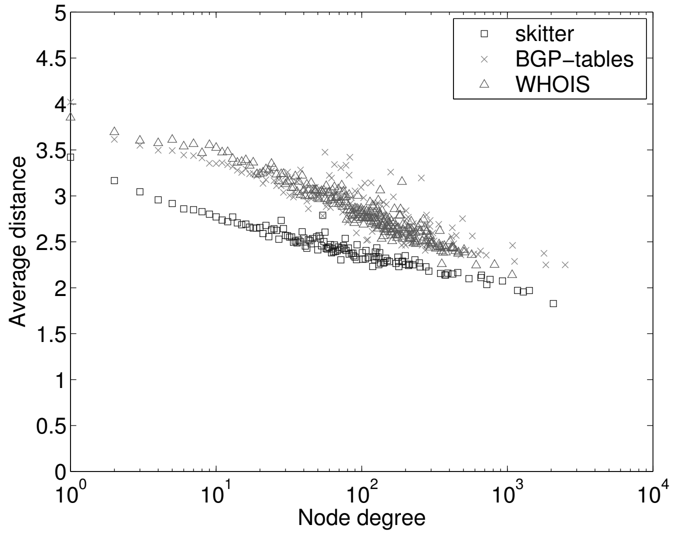
Again, the JDD plays a key role in determining the RCC values for all three graphs. The JDD captures connectivity information between pairs of nodes in the graph, while the RCC in effect measures the connectivity amongst the "rich" nodes in the graph. It follows that two graphs having the same JDD, will have the same RCC.

3.2.6 Distance

The distance distribution $d(x)$ is the probability that a random pair of nodes are at a distance x hops from each other. Although the distance distribution is a global topology characteristic, we can explain Figure 3.5(a) by the interplay between our local connectivity characteristics: the $\bar{k}$- and r-orders. First, we note that the skitter graph



(a) Distance $\mathbf{d}(\mathbf{x})$ distribution



(b) Average distance from k-degree nodes $\mathbf{d}(\mathbf{k})$.

Figure 3.5: Distance related metrics

stands out in Figure 3.5(a) as it has the smallest average distance and the smallest distribution width (Table 3.2). This result appears unexpected at first since the skitter graph has more nodes than the WHOIS graph and only about half the links. One would expect a denser graph (WHOIS) to have a lower average distance since adding links to a graph can only *decrease* the average distance in it. Surprisingly, the average distance of the most richly connected (highest $\bar{k}$) WHOIS graph is not the lowest. This result can be explained using the r-order. Indeed, a more disassortative graph has a greater proportion of radial links, shortening the distance from the fringe to the core. We use terms *fringe* and *core* to mean "zones" in the graph with low- and high-degree nodes respectively [96]. The skitter graph has the right balance between the relative number of links $\bar{k}$ and their radiality r, that minimizes the average distance. Compared to skitter, the BGP graph has larger distance because it is sparser (lower $\bar{k}$), and the WHOIS graph has larger distance because it is more assortative (higher r).

Another observation is that for all three graphs, including WHOIS, the average distance as a function of node degree exhibits relatively stable power laws in the full range of node degrees (Figure 3.5(b)), with exponents given in Table 3.2. The average distance for the WHOIS graph is higher than that of skitter and the BGP graph. As in the case of distance distribution, the r-order dictates these values. The existence of radian links in skitter reduces its average distance, while the WHOIS graph, being less disassortative does not have these links, thus increasing its average distance. In summary,the distance related metrics for the three graphs are determined by their JDD.

3.2.7 Betweenness

Betweenness measures the number of shortest paths passing through a node or link and, thus, estimates the potential traffic load on this node/link assuming uniformly distributed traffic following shortest paths. The simplest approach to calculating node betweenness requires long run times, but we used an efficient algorithm from [10]. We also modified it to compute link betweenness.

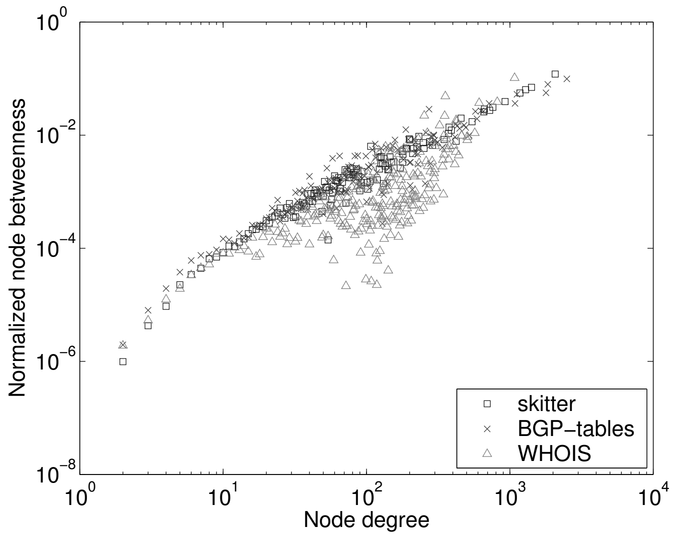
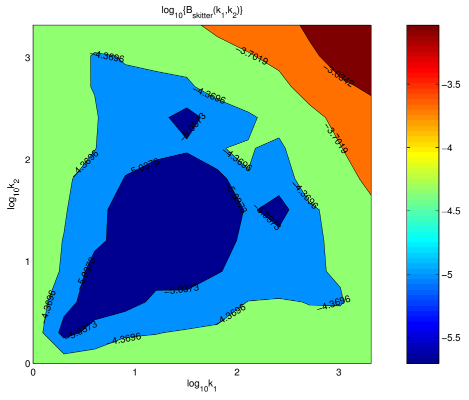


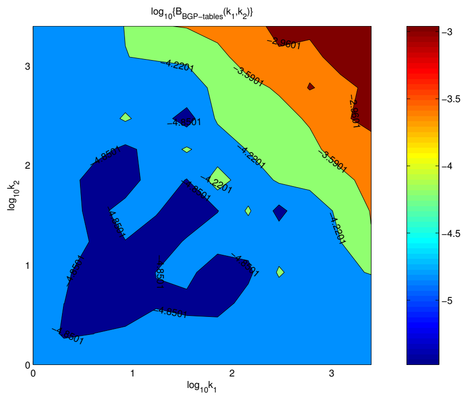
Figure 3.6: Node betweenness $\mathbf{B(k)/n/(n-1)}$.

For skitter and BGP graphs, node betweenness is a growing power-law function of node degree as seen in Figure 3.6 with exponents given in Table 3.2. An excess of medium degree nodes in the WHOIS graph (Figure 3.1) leads to greater path diversity and, hence, to lower betweenness values for these nodes.

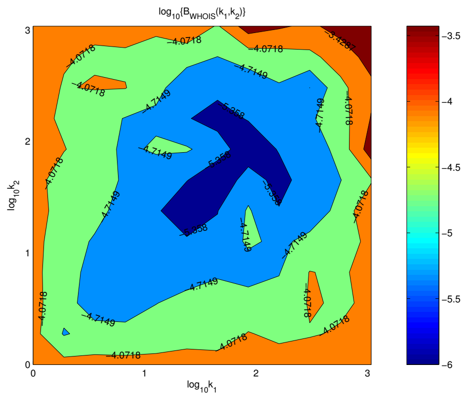
We also calculate average link betweenness as a function of degrees of nodes adjacent to a link $B(k_1, k_2)$ and show the plots in Figure 3.7. The contour plots provide information on the betweenness values of the links that connect similar or dissimilar degree nodes. One would expect links connecting high-degree nodes to exhibit highest link betweenness and thereby be used as a measure of link centrality. Contrary to popular belief, the contour plots show that link betweenness does not measure link centrality. First, betweenness of links adjacent to low-degree nodes (the left and bottom sides of the plots) is not the minimum. In fact, non-normalized betweenness of links adjacent to 1-degree nodes is constant and equal to $n-1$ (the number of destinations in the rest of the network). Similar values of betweenness characterize links elsewhere in the graph, including radial links between high and low-to-medium degree nodes and tangential links in the zone of medium-to-high degrees (diagonal zone from bottom-right to upper-left). While the maximum-betweenness links are between high-degree nodes as expected (the



(a) skitter



(b) BGP tables



(c) WHOIS

Figure 3.7: Logarithm of normalized link betweenness $\mathbf{B(k_1, k_2)/n/(n-1)}$ on a log-log scale.

upper right corner of the plots), the minimum-betweenness links are tangential in the medium-to-low degree zone (diagonal areas of low values from bottom-left to upper-right). We can explain the latter observation by the following argument. Let i and j be two nodes connected by a minimum-betweenness link l. The only shortest paths going through l are those between nodes that are *below* i and j, where "below" means further from the core and closer to the fringe. When the degrees of both i and j are small, the numbers of nodes below them (with lower degree) are small, too. Consequently, the number of shortest paths, proportional to the product of the number of nodes below i and j, attains its minimum at l. We conclude that link betweenness is not a measure of centrality but a measure of a certain combination of link centrality and radiality.

3.2.8 Spectrum

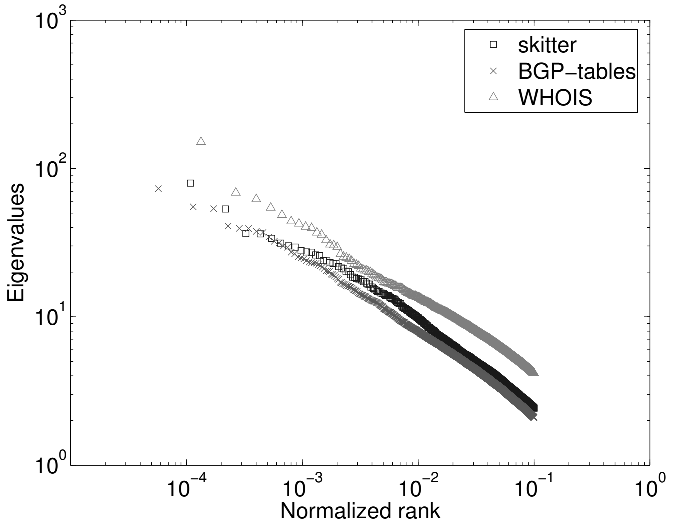


Figure 3.8: Spectrum - Absolute values of top 10% of eigenvalues ordered by their normalized rank: normalized rank is node rank divided by the total number of nodes in the graph.

The spectrum of a graph is the set of eigenvalues of its adjacency matrix. We computed the spectrum for the three graphs. The $\bar{k}$-order (BGP, skitter, WHOIS) plays a key role once again: the densest graph, WHOIS, is at the top in Figure 3.8 and its first eigenvalue is largest in Table 3.2. The eigenvalue distributions of all the three graphs follow power laws.

From our analysis, it follows that the relative positions of data curves for *all* topological metrics in our plots can be explained by either the average degree (increasing $\bar{k}$-order: BGP, skitter, WHOIS), or graph assortativity coefficient (decreasing r-order: WHOIS, BGP, skitter), or their interplay. How do these $\bar{k}$- and r-orders result from the specifics of the data sources? The explanation is relatively straightforward. In the WHOIS data, the abundance of tangential medium-degree links increases both the average degree $\bar{k}$ and assortativity coefficient r. Since the BGP graph has relatively more low-degree nodes than the skitter graph, its average degree is lower. At the same time, most links present only in BGP but not in skitter are backup links between low-degree ASes. Being tangential, these links increase the BGP graph's assortative coefficient.

3.3 Implications

To better understand the metrics and the interplay among them, we studied AS-level graphs obtained from various data sources—skitter, BGP, and WHOIS—with respect to all important metrics proposed in the literature. Of the set of metrics we considered, the *joint degree distribution* (JDD) $P(k_1, k_2)$ appears to play a central role in determining a wide range of other topological properties. Using only the average degree $\bar{k}$ and the assortativity coefficient r, the two coarse summary statistics of the JDD, we can explain the relative order of all other metrics for all our data sources. Indeed, the JDD plays a central role in even determining metrics such as distance distribution, betweenness and spectrum, all of which characterize global graph structure. Our finding regarding the JDD's definitiveness has important implications for developing an accurate topology generator since it would reduce the number of parameters one has to reproduce. By generating random graphs that reproduce the JDD of observed networks, it appears that these generated graphs will also automatically reproduce the other metrics. While this result might not hold for other kinds of Internet graphs, our study suggests that it would at least hold for Internet AS topologies.

Looking back at historical graph generation techniques, we observe that generators based on reproducing the degree distribution of observed graphs are more successful than generators based on merely reproducing the average degree. A graph's degree distribution contains more information about the graph structure than its average degree and as a result, degree distribution based generators do a better job at modeling topologies than generators based merely on average degree. Along the same lines, the JDD incorporates more information about the graph structure than its degree distribution; in fact the JDD captures connectivity information between any two nodes in the graph. Intuitively, the more node interconnectivity information we can extract from the original graph, the more accurate will be the random graphs reproducing this connectivity of the original graph. In the next chapter, we discuss how we extend this idea to define a graph property series, the dK-series, that captures connectivity amongst increasingly larger node neighborhoods and in the limit describes the original graph in its entirety. In addition to unifying all known graph metrics, the dK-series also forms a basis for all graph analysis and synthesis.

3.4 Acknowledgments

Chapter 3, in full, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Computer Communications Review (CCR), January 2006, Mahadevan, Priya; Krioukov, Dmitri; Fomenkov, Marina; Huffaker, Bradley; Dimitropoulos, Xenofontas; claffy, kc; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 4

dK-Series: Using Degree Correlations to Analyze and Generate Topologies

For the past fifteen years or so, researchers have been working towards developing algorithms to produce random graphs that match the topological characteristics of observed networks. One of the techniques of determining the accuracy of these random graphs generated is to compare them to the original graph with respect to all the important graph metrics such as distance distribution, betweenness, clustering, spectrum, *etc.* It is important that the generated graphs match the measured topology with respect to the metrics as values of these metrics dictate application behavior and performance.

Consider for example, a researcher studying the performance of a new routing protocol. The protocol performance is evaluated by typically emulating or simulating its behavior over an input network topology. Now, in order to correctly predict the routing protocol performance, it is critical that the input topology for the emulation has the same metric values as real observed networks, over which the protocol will ultimately be deployed. If this is not the case, then any conclusion that the researcher might draw about the protocol performance will be incorrect. While, the researcher might certainly use one of the real measured topologies such as the skitter, BGP or WHOIS graph (discussed in the previous chapter) as one of the input graphs to the evaluation infrastructure, ideally, the protocol should be evaluated on a wide range of random graphs. Studying the sensitiveness of the protocol to minor variations in the graph structure, might require the researchers to test their protocol over different graphs that have say the same joint degree distribution (JDD), but are random with respect to all other graph metrics. Studying the scalability of the protocol requires evaluating the protocol over topologies much larger but structurally similar to the real network. Thus it is imperative that we are able to generate random graphs of a range of sizes that reproduce important metric values of observed and measured topologies.

Metrics such as spectrum, distance distribution, and betweenness characterize global graph structure, while as discussed in Chapter 2, known approaches to generating graphs deal only with local, per-node statistics, such as the degree distribution. There are several problems with the graph metrics discussed in Section 2.1. First, they derive from a wide range of studies, and no one has established a systematic way to determine which metrics should be used in a given scenario. Second, there are no known algorithms capable of constructing graphs with desired values for most of the described metrics, save degree distribution and more recently, clustering [85]. Third, this list of metrics is incomplete. In particular, it cannot include any future metrics that may be of interest. Identifying such a metric might result in the finding that known synthetic graphs do not match this new metric's value and one will have to rewrite graph generation algorithms to account for this new metric. Such a process could potentially continue forever with the discovery of each new metric.

Given the limitations with reproducing metrics such as distance distribution, betweenness and spectrum, our approach, is to understand the *interdependencies* among these metrics. Is there possibly one graph property, or series of properties that define all others? If this theory indeed holds, then we only need to reproduce this one property instead of trying to reproduce the wide range of metrics.

Based on our analysis of a wide variety of graphs in the previous chapter, we put forth a series of graph characteristics, *node degree correlations*, and demonstrate how they progressively capture essential graph properties by incorporating more and more information about connectivity among increasingly larger node neighborhoods.

Using both the theoretical argument and empirical evidence, we show that by reproducing degree correlations at the right level, one can generate graphs that automatically match *all* other metrics, including those that have not been considered yet. Validity of the last strong statement follows from the fact that in the limiting case, our synthetic graphs converge to the original: if two graphs are the same, then *all* their metrics are the same, too. This unique convergence feature makes our approach stand separately from all the previous topology research and concludes the quest for a single set of definitive topology metrics. In the rest of this chapter, we establish our construction of the properties $\mathcal{P}_d$, which we will call the *dK-series*. We use this series to describe and constrain random graphs in successively finer detail.

4.1 Requirements for Defining a Unifying Property Series

Given the limitations with the current set of graph metrics, there are many requirements for defining our unifying property series. We want our graph generation algorithms to reproduce *any* graph metric – current or future. One way of achieving this goal is to ensure that our property series can describe the original graph in its entirety. In this case, any random graph that reproduces the description of the original graph in its entirety will necessarily reproduce all possible graph metrics. Also, given our inability to propose algorithms that can match given values of metrics such as spectrum and distance distribution, our goal is to ensure that we can generate random graphs that match specified values for the properties in our series.

To summarize, for any graph G, we wish to identify a *series* of graph properties $\mathcal{P}_d$, $d = 0, 1, \ldots$ that will satisfy the following requirements:

1. *constructability*: we can construct graphs having these properties;
2. *inclusion*: any property $\mathcal{P}_d$ subsumes all properties $\mathcal{P}_i$ with $i = 0, \ldots, d-1$: that is, a graph having property $\mathcal{P}_d$ is guaranteed to also have all properties $\mathcal{P}_i$ for $i < d$;

3. *convergence*: as d increases, the set of graphs having property $\mathcal{P}_d$ "converges" to G: that is, there exists a value of index $d = D$ such that all graphs having property $\mathcal{P}_D$ are isomorphic to G.

Our observations from the analysis of AS-level graphs in Chapter 3 provides us with a starting point for defining the property series. The graph's Joint Degree Distribution or JDD appeared to dictate the values for all metric including global ones such as distance distribution and betweenness. The JDD provides connectivity information on pairs of node degrees in the graph. We surmised that, at least for AS-level graphs, a random graph generator that reproduces the JDD of a given graph might also automatically reproduce values for other important graph metrics.

Looking at historical topology generators, we observe that the initial generators were based on reproducing a given graph's average degree. When it was shown that such graph models were inadequate, researchers developed generators based on reproducing the graph's node degree distribution. They found that the node degree distribution captured more information about the graph structure than the average degree and thus generators that reproduced the node degree distribution were able to better reproduce the graph characteristics. The average degree is a scalar value and does not provide any connectivity information on a per-node basis. The node degree distribution, on the other hand, is a vector and describes the connectivity for each node in the graph. Taking this point further, the JDD is a matrix and captures more connectivity information than both the average degree and node degree distribution. The JDD summarizes connectivity information between pairs of node degrees in the graph. Extending this idea further, it appears that connectivity information amongst increasingly larger node neighborhoods progressively described more information about the graph structure. Thus, if we can extract connectivity information amongst increasingly larger node neighborhoods, we'll come closer and closer to describing the original graph. We use this idea to define our unifying property series.

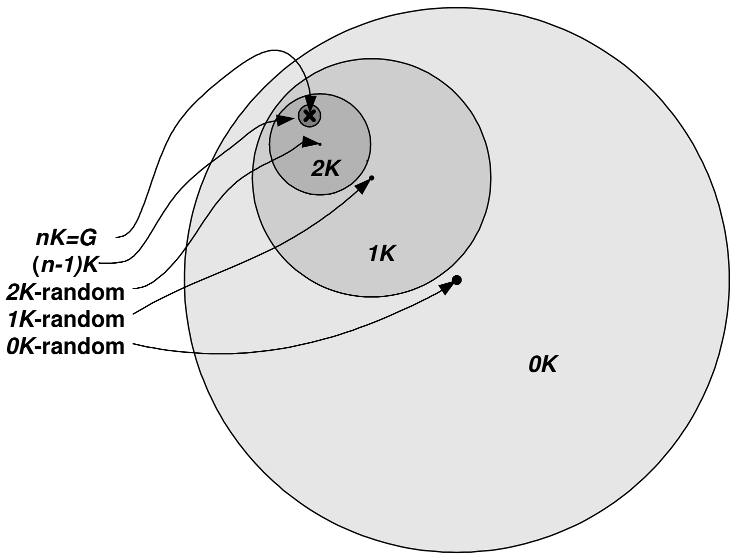
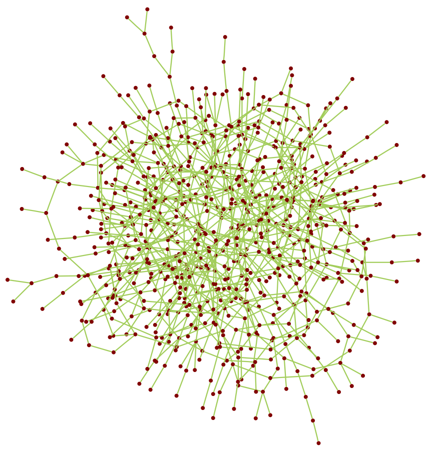


Figure 4.1: The dK- and dK-random graph hierarchy. The circles represent dK-graphs, whereas their centers represent dK-random graphs. The cross is the nK-graph isomorphic to a given graph G.

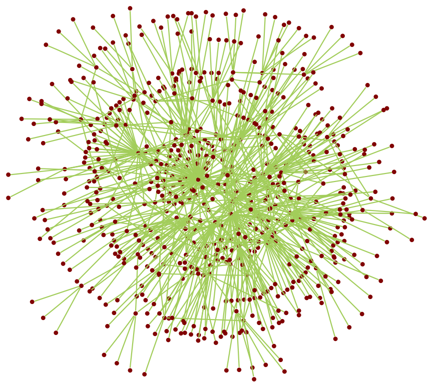
4.2 dK-series and dK-graphs

We begin with the observation that the most basic properties of a network topology characterize its connectivity. The coarsest connectivity property is the *average node degree* $\bar{k} = 2m/n$, where $n = |V|$ and $m = |E|$ are the numbers of nodes and links in a given graph $G(V, E)$. Therefore, the first property $\mathcal{P}_0$ in our dK-series $\mathcal{P}_d$ is that the graph's average degree $\bar{k}$ has the same value as in the given graph G. In Figure 4.1 we schematically depict the set of all graphs having property $\mathcal{P}_0$ as $0K$-graphs, defining the largest circle. Generalizing, we adopt the term *dK-graphs* to represent the set of all graphs having property $\mathcal{P}_d$.

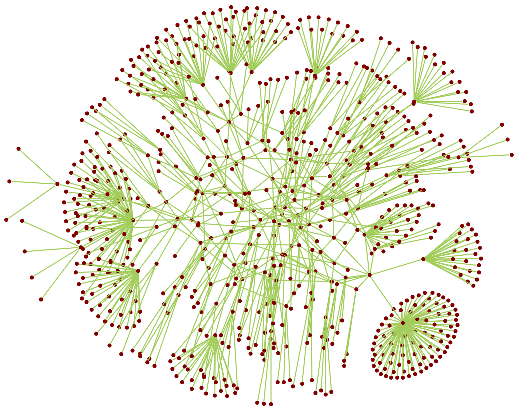
The $\mathcal{P}_0$ property tells us the average number of edges per node, but it does not tell us the distribution of degrees across nodes. In particular, we do not know the number of nodes $n(k)$ of each degree k in the graph. We define property $\mathcal{P}_1$ to capture this information: $\mathcal{P}_1$ is therefore the property that the graph's *node degree distribution* $P(k) = n(k)/n$ has the same form as in the given graph G. It is convenient to call $P(k)$ the *$1K$-distribution*. Sacrificing a certain amount of rigor, we interchangeably use the



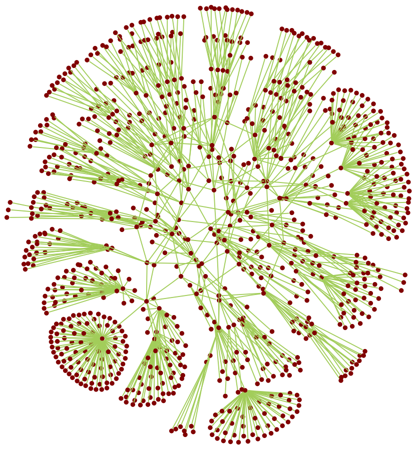
(a) $0K$-graph



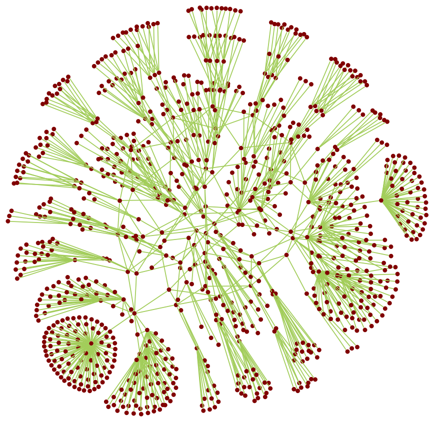
(b) $1K$-graph



(c) $2K$-graph



(d) $3K$-graph



(e) original HOT graph

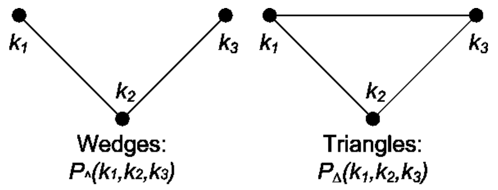
Figure 4.2: Picturizations of dK-graphs and the original HOT graph illustrating the convergence of dK-series.

enumeration of nodes having some property in a given graph, *e.g.*, $n(k)/n$, with the probability that a node has this property in a graph ensemble, *e.g.*, $P(k)$. The two become identical when $n \to \infty$. We refer the readers to [8] for additional information.

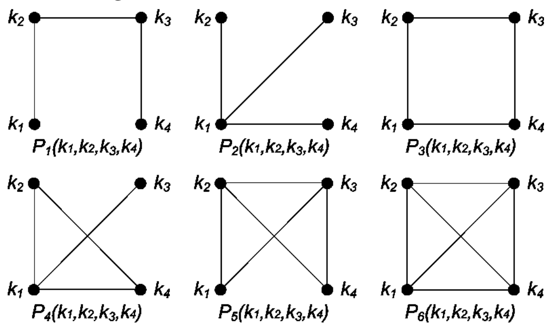
$\mathcal{P}_1$ implies at least as much information about the network as $\mathcal{P}_0$, but not vice versa: given $P(k)$, we find $\bar{k} = \sum kP(k)$. $\mathcal{P}_1$ provides more information than $\mathcal{P}_0$, and it is therefore a more restrictive metric: the set of $1K$-graphs is a subset of the set of $0K$-graphs. Figure 4.1 illustrates this inclusive relationship by drawing the set of $1K$-graphs inside the set of $0K$-graphs.

Continuing to $d = 2$, we note that the degree distribution constrains the number of nodes of each degree in the network, but it does not describe the interconnectivity of nodes with given degrees. That is, it does not provide any information on the total number $m(k, k')$ of links between nodes of degree k and k'. We define the third property $\mathcal{P}_2$ in our series as the property that the graph's *joint degree distribution* (JDD) has the same form as in the given graph G. The JDD, or the *$2K$-distribution*, is $P(k_1, k_2) = m(k_1, k_2)\mu(k_1, k_2)/(2m)$, where $\mu(k_1, k_2)$ is 2 if $k_1 = k_2$ and 1 otherwise. The JDD describes degree correlations for *pairs* of connected nodes. Given $P(k_1, k_2)$, we can calculate $P(k) = (\bar{k}/k)\sum_{k'} P(k, k')$, but not vice versa. Consequently, the set of $2K$-graphs is a subset of the $1K$-graphs. Therefore, Figure 4.1 depicts the smaller $2K$-graph circle inside $1K$.

We can continue to increase the amount of connectivity information by considering degree correlations among greater numbers of connected nodes. To move beyond $2K$, we must begin to distinguish the various geometries that are possible in interconnecting d nodes. To introduce $\mathcal{P}_3$, we require the following two components: 1) *wedges*: chains of 3 nodes connected by 2 edges, called the $P_\wedge(k_1, k_2, k_3)$ component; and 2) *triangles*: cliques of 3 nodes, called the $P_\triangle(k_1, k_2, k_3)$ component:

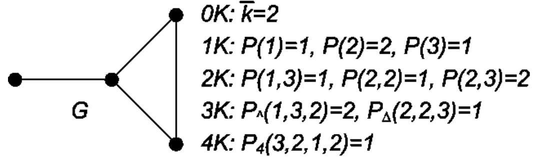


As the two geometries occur with different frequencies among nodes having different degrees, we require a separate probability distribution for each configuration. We call these two components taken together the $3K$*-distribution*.



For $\mathcal{P}_4$, we need the above six distributions: where instead of indices $\wedge, \triangle$ we use for $d = 3$, we have all non-isomorphic graphs of size 4 numbered by $1, \ldots, 6$. We note that the order of k-arguments generally matters, although we can permute any pair of arguments corresponding to pairs of nodes whose swapping leaves the graph isomorphic. For example: $P_\wedge(k_1, k_2, k_3) \neq P_\wedge(k_2, k_1, k_3) \neq P_\wedge(k_1, k_3, k_2)$, but $P_\wedge(k_1, k_2, k_3) = P_\wedge(k_3, k_2, k_1)$.

In the following figure, we illustrate properties $\mathcal{P}_d$, $d = 0, \ldots, 4$, calculated for a given graph G of size 4, where for simplicity, values of all distributions P are the total numbers of corresponding subgraphs, i.e., $P(2, 3) = 2$ means that G contains 2 edges between 2- and 3-degree nodes.


G
0K: k̄=2
1K: P(1)=1, P(2)=2, P(3)=1
2K: P(1,3)=1, P(2,2)=1, P(2,3)=2
3K: P∧(1,3,2)=2, P△(2,2,3)=1
4K: P4(3,2,1,2)=1

Generalizing, *we define the* dK-distributions *to be degree correlations within non-isomorphic simple connected subgraphs of size* d *and the* dK-series $\mathcal{P}_d$ *to be the* *series of properties constraining the graph's dK-distribution to the same form as in a given graph G.* In other words, $\mathcal{P}_d$ tells us how groups of d-nodes with degrees $k_1, ..., k_d$ interconnect. In the 'dK' acronym, 'K' represents the standard notation for node degrees, while '*d*' refers to the number of *d*egree arguments k of the dK-distributions $P(k_1, \ldots, k_d)$ and to the upper bound of the *d*istance between nodes with known degree correlations. Moving from $\mathcal{P}_d$ to $\mathcal{P}_{d+1}$ in describing a given graph G is somewhat similar to including the additional $d+1$'th term of the Fourier (time) or Taylor series representing a given function F. In both cases, we describe wider "neighborhoods" in G or F to achieve a more accurate representation of the original structure.

The dK-series definition satisfies the inclusion and convergence requirements described above. The inclusion requirement is satisfied because any graph of size d is a subgraph of some graph of size $d+1$. Convergence follows from the observation that in the limit of $d = n$, the set of nK-graphs contains only one element: G itself. As a consequence of the convergence property, any topology metric we can define on G will eventually be captured by dK-graphs with a sufficiently large d.

Hereafter, our main concerns with the dK-series become: 1) how well we can satisfy our first requirement of constructability and 2) how fast the dK-series converges toward the original graph. We address both these two concerns in the rest of this chapter.

The reason for the second concern is that the number of probability distributions required to fully specify the dK-distribution grows quickly with d: see [89] for the number of non-isomorphic simple connected graphs of size d. Relative to the existing work on topology generators typically limited to $d = 1$ [2, 61, 103], we present and implement algorithms for graph construction for $d = 2$ and $d = 3$. We present these algorithms in Section 4.3.1 and then show in Section 4.4 that the dK-series converges quickly: $2K$-graphs are sufficient for most practical purposes for the graphs we consider, while $3K$-graphs are essentially identical to observed and modeled Internet topologies.

To motivate our ability to capture increasingly complex graph properties by increasing d, we present visualizations of dK-graphs generated using the dK-randomizing approach we will discuss in Section 4.3.1. Figure 4.2 depicts random $0K$-, $1K$-, $2K$- and $3K$-graphs matching the corresponding distributions of the HOT graph, a representative router-level topology from [54]. This topology is particularly interesting, because, prior to our work, reproducing router-level topologies using only degree distributions had proven difficult [54]. However, a visual inspection of our generated topologies shows good convergence properties of the dK-series: while the $0K$-graph and $1K$-graph have little resemblance with the HOT topology, the $2K$-graph is much closer than the previous ones and the $3K$ graph is almost identical to the original. Although the visual inspection is encouraging, we defer more careful comparisons to Section 4.4.

4.3 Constructing dK-graphs

There are several approaches for constructing dK-graphs for $d = 0$ and $d = 1$. We extended a number of these algorithms to work for higher values of d. In Section 4.3.1, we describe these approaches, their practical utility, and our new algorithms for $d > 1$. In Section 4.3.2, we introduce the concept of *dK-random graphs*, in Section 4.3.3, a *dK-space exploration* methodology. We use this methodology to determine the lowest values of d such that dK-graphs approximate a given topology with the required degree of accuracy.

4.3.1 Algorithms to construct dK-graphs

We classify existing approaches to constructing $0K$- and $1K$-graphs into the following categories: *stochastic*, *pseudograph*, *matching*, and two types of *rewiring*: *randomizing* and *targeting*. We attempted to extend each of these techniques to general dK-graph construction. In this section, we qualitatively discuss the relative merits of each of these approaches before presenting a more quantitative comparison in Section 4.4.

Stochastic

The simplest and most convenient for theoretical analysis is the stochastic approach. For $0K$, reproducing an n-sized graph with a given expected average degree $\bar{k}$ involves connecting every pair of n nodes with probability $p_{0K} = \bar{k}/n$. This construction forms the classical (Erdős-Rényi) random graphs $\mathcal{G}_{n,p}$ [30]. Recent efforts have extended this stochastic approach to $1K$ and $2K$ [7, 18, 26]. In these cases, one first labels all nodes i with their expected degrees q_i drawn from the distribution $P(k)$ and then connects pairs of nodes (i, j) with probabilities $p_{1K}(q_i, q_j) = q_i q_j/(n\bar{q})$ or $p_{2K}(q_i, q_j) = (\bar{q}/n)P(q_i, q_j)/(P(q_i)P(q_j))$ reproducing the expected values of $1K$- or $2K$-distributions, respectively.

In theory, we could generalize this approach for any d in two stages: 1) *extraction*: given a graph G, calculate the frequencies of all (including disconnected) d-sized subgraphs in G, and 2) *construction*: prepare an n-sized set of q_i-labeled nodes and connect their d-sized subsets into different subgraphs with (conditional) probabilities based on the calculated frequencies. In practice, we find the stochastic approach performs poorly even for $1K$ because of high statistical variance. For example, many nodes with expected degree 1 wind up with degree 0 after the construction phase, resulting in many tiny connected components.

Pseudograph

The pseudograph (also known as *configuration*) approach is probably the most popular and widely used class of graph-generating algorithms. In its original form [2, 63], it applies only to the $1K$ case. Relative to the stochastic approach, it reproduces a given degree distribution exactly, but does not necessarily construct simple graphs. That is, it may construct graphs with both ends of an edge connected to the same node (self-loops) and with multiple edges between the same pair of nodes (loops).

It operates as follows: given the number of nodes, $n(k)$, of degree k, $n = \sum_{k=1}^{k_{\max}} n(k)$, first prepare $n(k)$ nodes with k stubs attached to each node, $k =$

$1, \ldots, k_{\max}$, and then randomly choose pairs of stubs and connect them to form edges. To obtain a simple connected graph, remove all loops and extract the largest connected component.

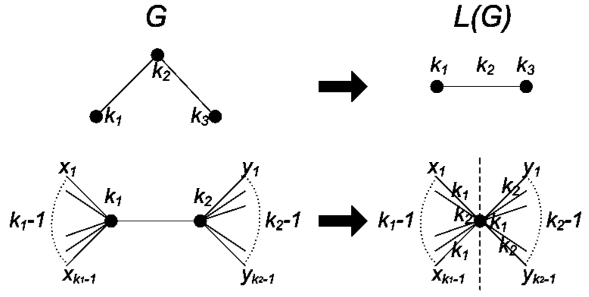
We extend this algorithm to $2K$ as follows: given the number $m(k_1, k_2)$ of edges between k_1- and k_2-degree nodes, $m = \sum_{k_1,k_2=1}^{k_{\max}} m(k_1, k_2)$, we first prepare a list of $m(k_1, k_2)$ disconnected edges and label the ends of each edge by their respective degree values k_1 and k_2, $k_1, k_2 = 1, \ldots, k_{\max}$. Next, corresponding to each degree k, $k = 1, \ldots, k_{\max}$, we create a list of all edge-ends that were labeled with k; from this list, we randomly select groups of k edge-ends to create the nodes in the graph with degree k, $k = 1, \ldots, k_{\max}$.

The pseudograph algorithm produces good results for $d = 2$. Unfortunately, we could not generalize it easily for $d > 2$ because starting at $d = 3$, d-sized subgraphs overlap over edges. Such overlapping introduces a series of topological constraints and non-local dependencies among different subgraphs, and we could not find a simple technique to preserve these combinatorial constraints during the construction phase.

To demonstrate the overlap problem, we consider the simplest $d = 3$ case. We first note that the $3K$ pseudograph algorithm applied to graph G is equivalent to the following modification of the $2K$ pseudograph algorithm applied to G's line graph $L(G)$. Recall that a line graph $L(G)$ [78] is a graph whose vertices are edges of G and two vertices of $L(G)$ are connected *iff* the two corresponding edges in G are adjacent. Note that an edge between k_1- and k_2-degree nodes in G (a (k_1, k_2)-edge) maps to a node of degree $k_1 + k_2 - 2$ in $L(G)$. We need more detailed degree information preserved in $L(G)$: we label every edge in $L(G)$ with degree triplets (k_1, k_2, k_3) indicating that a pair of (k_1, k_2)- and (k_2, k_3)-edges are connected over their k_2-ends (k_2-degree node) in G. Every (k_1, k_2, k_3)-labeled edge has the two ends which we refer to as (k_1, k_2)- and (k_3, k_2)-ends, i.e., k_2 labels the middle of the edge. We thus map (k_1, k_2, k_3)-wedges in G to (k_1, k_2, k_3)-edges in $L(G)$.

Simplifying our problem and assuming for a moment that G has zero clustering (no triangles), we see that every (k_1, k_2)-edge in G participates in

$k_1 - 1$ of (x_i, k_1, k_2)-wedges and in $k_2 - 1$ of (k_1, k_2, y_j)-wedges, where x_i and y_j, $i = 1, \ldots, k_1 - 1$, $j = 1, \ldots, k_2 - 1$, are some random degrees. In $L(G)$, this observation maps to the constraint that we need $k_1 - 1$ of (x_i, k_1, k_2)-edges and $k_2 - 1$ of (k_1, k_2, y_j)-edges to connect over their, respectively, (k_2, k_1)- and (k_1, k_2)-ends to form a $(k_1 + k_2 - 2)$-degree node:



Our $3K$ algorithm for G becomes the $2K$ algorithm for $L(G)$ with the following modifications. Given the number $m_\wedge(k_1, k_2, k_3)$ of (k_1, k_2, k_3)-wedges in G, first prepare $m_\wedge(k_1, k_2, k_3)$ disconnected $L(G)$ edges labeled by (k_1, k_2, k_3), and then randomly select groups of $p - 1$ edge-ends labeled by (q, p) and $q - 1$ edge-ends labeled by (p, q) to form $(p + q - 2)$-degree nodes.

Unfortunately, the above construction does not preserve one of the basic topological constraints that k-degree nodes in G map to k-cliques in $L(G)$. Indeed, this constraint means that for all k, every $(\cdot, k, \cdot)$-labeled edge in $L(G)$ belongs to a k-clique. We see that our modification of the $2K$ pseudograph algorithm for $L(G)$ does not preserve this property, and we could not find an easy way to resolve this problem.

Matching

The matching approach differs from the pseudograph approach in avoiding loops during the construction phase. In the $1K$ case, the algorithm works exactly as its pseudograph counterpart but skips pairs of stubs that form loops if connected.

Unfortunately, loop avoidance suffers from various forms of deadlock for both $1K$ and $2K$. In both cases, the algorithms can end up in incomplete configurations when not all edges are formed, and the graph cannot be completed because there are no suitable stub pairs remaining that can be connected without forming loops. We devise several techniques to deal with these problems. Specifically, while randomly choosing pairs of stubs to connect in order to create an edge, we choose stubs in such a manner that adding an edge between them does not create a self-edge or a loop in the graph. To solve the problem of not enough suitable stub pairs remaining to connect in order to complete the graph, we resorted to the technique described below. Consider the case where we need to connect degrees k_1 and k_2 to form an edge. Without loss of generality, let us assume that we have a stub attached to node n_p having degree k_1 and we need to find a stub attached to a node with degree k_2. If there are no more suitable stubs attached to k_2-degree nodes, we randomly pick a stub attached to node n_q having degree k_2. Next, we walk through the list of edges already created by connecting pairs of k_1- and k_2-degree stubs. We pick an edge at random and let node n_a with degree k_1 and node n_b with degree k_2 be the nodes on which the chosen edge was incident. We now remove the edge between n_a and n_b and check if by creating edges n_p-n_b and edge n_a-n_q, the graph stays loop-free and does not have any self-edges. If not, we go back and find another random edge and repeat the process again. This technique, while attempting to find new nodes to connect between degrees k_1 and k_2, also ensures that the JDD of the graph remains unchanged.

With these additional techniques, we obtained good results for $2K$ graphs. Once again, we could not generalize matching for $d > 2$ for essentially the same reasons related to subgraphs' overlapping and non-locality as in the pseudograph case.

Rewiring

The rewiring approaches are generalizable to any d and work well in practice. They involve dK-preserving rewiring as illustrated in Figure 4.3.

The main idea is to rewire *random* (pairs of) edges preserving an existing form of the dK-distribution. For $d = 0$, we rewire a random edge to a random pair of nodes, thus preserving $\bar{k}$. For $d = 1$, we rewire two random edges that do not alter $P(k)$,

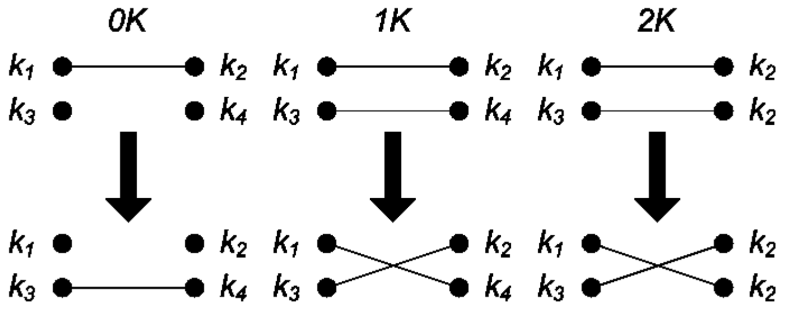


Figure 4.3: dK-preserving rewiring for $d = 0, 1, 2$.

as shown in Figure 4.3. If, in addition, there are at least two nodes of equal degrees adjacent to the different edges in the edge pair, then the same rewiring leaves $P(k, k')$ intact. Due to the inclusion property of the dK-series, $(d + 1)K$-rewirings form a subset of dK-rewirings for $d > 0$. For example, to preserve $3K$, we permit a $2K$-rewiring only if it also preserves the wedge and triangle distributions.

The *dK-randomizing rewiring algorithm* amounts to performing dK-preserving rewirings a sufficient number of times for some dK-graph. A "sufficient number" means enough rewirings for this process to lead to graphs that do not change their properties even if we subject them to additional rewirings. In other words, this rewiring process *converges* after some number of steps, producing random graphs having property $\mathcal{P}_d$. Even for $d = 1$, there are no known rigorous results regarding how quickly this process converges, but [38] shows that this process is an irreducible, symmetric and aperiodic Markov chain and demonstrates experimentally that it takes $O(m)$ steps to converge.

In our experiments in Section 4.4, we employ the following strategy applicable for any d. We first calculate the number of possible initial dK-preserving rewirings. By "initial rewirings" we mean rewirings we can perform on a given graph G, to differentiate them from rewirings we can apply to graphs obtained from G after its first (and subsequent) rewirings. We then subtract the number of rewirings that leave the graph isomorphic. For example, rewiring of any two $(1, k)$- and $(1, k')$-edges is a dK-preserving rewiring, for any d, and more strongly, the graph before rewiring is isomorphic to the graph after rewiring. We multiply this difference by 10, and perform the resulting number of random rewirings. At the end of our rewiring procedure, we explicitly verify that randomization is indeed complete and the process has converged by further increasing the number of rewirings and checking that all graph characteristics remain unchanged.

One obvious problem with dK-randomization is that it requires an original graph G as input to construct its dK-random versions. It cannot start with a description of the dK-distribution to generate random dK-graphs as is possible with the other construction approaches discussed above.

To address this limitation, we consider the inverse process of *dK-targeting $d'K$-preserving rewiring*, also known as *Metropolis dynamics* [62]. It incorporates the following modification to $d'K$-preserving rewiring: every rewiring step is accepted only if it moves the graph "closer" to $\mathcal{P}_d$. In practice, we can then employ targeting rewiring to construct dK-graphs with high values of d by beginning with any $d'K$-graph where $d' < d$. Recall that we can always compute $\mathcal{P}_{d'}$ from $\mathcal{P}_d$ due to the inclusion property of the dK-series. For instance, we can start with a graph having a given degree distribution ($d' = 1$) [99], and then move it toward a dK-graph via dK-targeting $1K$-preserving rewiring.

The definition of "closer" above requires further explanation. We require a set of distance metrics (functions) that quantitatively differentiate two graphs based on the values of their dK-distributions. In our experiments, we use the sum of squares of differences between the existing and target numbers of subgraphs of a given type. For example, in the $d = 2$ case, we measure the distance between the target graph's JDD and the JDD of the current graph being rewired by $\mathcal{D}_2 = \sum_{k_1,k_2} \left[m_{\text{current}}(k_1,k_2) - m_{\text{target}}(k_1,k_2)\right]^2$, and at each rewiring step, we accept the rewiring only if it decreases this distance. Note that $\mathcal{D}_2$ is non-negative and equals zero only when reaching the target JDD. For $d = 3$, this distance $\mathcal{D}_3$ is a sum of squares of differences between the current and target numbers of wedges and triangles, and we can generalize it to $\mathcal{D}_d$ for any d. A potential problem with dK-targeting $d'K$-

preserving rewiring is that it can be non-ergodic, meaning that there might be no chain of $d'K$-preserving $\mathcal{D}_d$-decreasing rewirings leading from the initial $d'K$-graph to the target dK-graph. In other words, we cannot be sure beforehand that any two $d'K$-graphs are connected by a sequence of $d'K$-preserving and $\mathcal{D}_d$-decreasing rewirings.

To address this problem we note that the $d'K$-randomizing and dK-targeting $d'K$-preserving rewirings are actually two extremes of an entire family of rewiring processes. Indeed, let $\Delta\mathcal{D}_d = \mathcal{D}_{d,\text{after}} - \mathcal{D}_{d,\text{before}}$ be the difference of distance to the target dK-distribution computed before and after a $d'K$-preserving rewiring step. As with the usual dK-targeting rewiring, we accept a rewiring step if $\Delta\mathcal{D}_d < 0$, but even if $\Delta\mathcal{D}_d \geqslant 0$, we also accept this step with probability $e^{-\Delta\mathcal{D}_d/T}$, where $T > 0$ is some parameter that we call *temperature* because of the similarity of the process to simulated annealing.

In the $T \to 0$ limit, this probability goes to 0, and we have the standard dK-targeting $d'K$-preserving rewiring process. When $T \to \infty$, the probability approaches 1, yielding the standard $d'K$-randomizing rewiring process. To verify ergodicity, we can start with a high temperature and then gradually cool the system while monitoring any metric known to have different values in dK- and $d'K$-graphs. If this metric's value forms a continuous function of the temperature, then our rewiring process is ergodic. Maslov *et al.* performed these experiments in [60] and demonstrated ergodicity in the case with $d' = 1$ and $d = 2$. In our experiments in Section 4.4, we always obtain a good match for all target graph metrics in considering $(d', d) < 4$. Thus, we perform rewiring at zero temperature without further considering ergodicity. If however in some future experiments one detects the lack of a smooth convergence of rewiring procedures, then one should first verify ergodicity using the methodology above.

For all the algorithms discussed in this section, we do not check for graph connectedness at each step of the algorithm since: 1) it is an expensive operation and 2) all resulting graphs always have giant connected components (GCCs) with characteristics similar to the whole disconnected graphs.

Table 4.1: The summary of dK-series.

Tag dK	Property symbol	dK-distribution	$\mathcal{P}_d$ defines $\mathcal{P}_{d-1}$	Edge existence probability in stochastic constructions
$0K$	$\mathcal{P}_0$	k		$p_{0K} = k/n$
$1K$	$\mathcal{P}_1$	$P(k)$	$\bar{k} = \sum kP(k)$	$p_{1K}(q_1, q_2) = q_1 q_2/(n\bar{q})$
$2K$	$\mathcal{P}_2$	$P(k_1, k_2)$	$P(k) = (\bar{k}/k) \sum_{k'} P(k, k')$	$p_{2K}(q_1, q_2) = (\bar{q}/n)P(q_1, q_2)/(P(q_1)P(q_2))$
$3K$	$\mathcal{P}_3$	$P_\wedge(k_1, k_2, k_3)$ $P_\triangle(k_1, k_2, k_3)$	By counting edges, we get $P(k_1, k_2) \sim \sum_k \{P_\wedge(k, k_1, k_2) + P_\triangle(k, k_1, k_2)\} /(k_1 - 1) \sim \sum_k \{P_\wedge(k_1, k_2, k) + P_\triangle(k_1, k_2, k)\} /(k_2 - 1)$, where we omit normalization coefficients.	
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
nK	$\mathcal{P}_n$	G		

4.3.2 dK-random graphs

No dK-graph-generating algorithm can quickly construct the set of *all* dK-graphs because: 1) such sets are too large, especially for small d; and, less obviously, 2) all algorithms try to produce graphs having property $\mathcal{P}_d$ while remaining *unbiased* (random) with respect to all other properties. One can check directly that the last characteristic applies to all the algorithms we have discussed above.

As a consequence, the dK-graph construction algorithms result in non-uniform sampling of graphs with different values of properties that are not fully defined by $\mathcal{P}_d$. More specifically, two generated dK-graphs having different forms of a $d'K$-distribution with $d' > d$ can appear as the output of these algorithms with drastically different probabilities. Some dK-graphs have such a small probability of being constructed that we can safely assume they never arise.

For example, consider the simplest $0K$ stochastic construction, i.e., the classical random graphs $\mathcal{G}_{n,p}$. Using a probabilistic argument, one can show that the naturally-occurring $1K$-distribution (degree distribution) in these graphs has a specific form: binomial, which is closely approximated by the Poisson distribution: $P_{0K}(k) = e^{-\bar{k}}\bar{k}^k/k!$ [27]. The $0K$ stochastic algorithm can produce a graph with a different $1K$-distribution, *e.g.*, the power-law $P(k) \sim k^{-\gamma}$ with extremely low probability. Indeed, suppose $n \sim 10^4$, $\bar{k} \sim 5$, and $\gamma \sim 2.1$, so that the characteristic maximum degree is $k_{\max} \sim 2000$ (we chose these values to reflect measured values for Internet AS topologies). In this case, the probability that a $\mathcal{G}_{n,p}$-graph contains at least one node with degree equal to $k_{\max}$ is dominated by $1/2000! \sim 10^{-6600}$, and the probability that the remaining degrees simultaneously match those required for a power law is much lower.

It is thus natural to introduce a set of graphs that correspond to the graphs most likely to be generated by dK-graph constructing algorithms. We call such graphs the *dK-random graphs*. These graphs have property $\mathcal{P}_d$ but are unbiased with respect to any other more constraining property. In this sense, the dK-random graphs are the *maximally random* or *maximum-entropy* dK-graphs. We note that the entropy of a discrete distribution $P(x)$ is $H[P(x)] = -\sum_x P(x) \log P(x)$. If the sample space is also finite, then among all the distributions with a fixed average, the binomial distribution maximizes entropy [40]. Our term *maximum entropy* here has the following justification. As we have just seen, $0K$-random graphs have the maximum-entropy value of the $1K$-distribution since their node degree distribution is the distribution with the maximum entropy among all the distributions with a fixed average.

The $1K$-random graphs have the maximum-entropy value of the $2K$-distribution since their joint degree distribution, $P_{1K}(k_1, k_2) = \tilde{P}(k_1)\tilde{P}(k_2)$, where $\tilde{P}(k) = kP(k)/\bar{k}$ [27], is the distribution with the maximum joint entropy (minimum mutual information) among all the joint distributions with fixed marginal distributions. The mutual information of a joint distribution $P(x, y)$ is $I[P(x, y)] = H[P(x)] + H[P(y)] - H[P(x, y)]$, where $P(x)$ and $P(y)$ are the marginal distributions.

The main point we extract from these observations is that in trying to construct dK-graphs, we generally obtain graphs from subsets of the dK-space. We call these subsets dK-random graphs and schematically depict them as centers of the dK-circles in Figure 4.1. These graphs do have property $\mathcal{P}_d$ and, consequently, properties $\mathcal{P}_i$ with $i < d$, but they might not ever display property $\mathcal{P}_j$ with $j > d$ since the jK-distributions has specific, maximum-entropy values in the jK-space that may not overlap with dk-

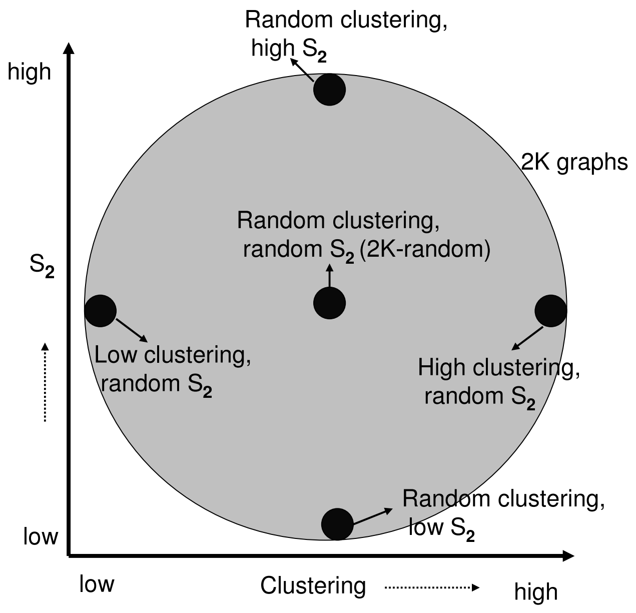

Random clustering, high S_2
high
2K graphs
Random clustering, random S_2 (2K-random)
S_2
Low clustering, random S_2
High clustering, random S_2
Random clustering, low S_2
low
low
Clustering
high

Figure 4.4: $2K$-space exploration

random graphs.

4.3.3 dK-space explorations

In general, to understand whether a good approximation of the original topology can be achieved with dK-random graphs, we need to *explore the dK-graph space*. In schematics of Figure 4.1, such exploration means that we want to find out how large the dK circle is. If it is large, then the spread of metrics that are not fixed by the dK-distribution is wide within this circle and we have little chance to have a good approximation of the original topology with dK-random graphs. If the circle is small, then the dK-random graphs must be necessarily close to the original. We can assess the size of the dK circle by checking properties of the graphs located at the periphery of the circle – non-random dK-graphs with the extremal values of the metrics that are not unambiguously determined by the dK-distribution.

Often, we are also interested in exploring the structural diversity among dK-graphs. To motivate the need for this exploration, let us go back to our example of an experimenter evaluating a new inter-domain routing protocol and wishing to test its sensitivity to a range of possible deployments. At this point, it may have been established that Internet AS topologies are $2K$-random. Thus, while the experimenter may test the behavior of their protocol relative to a known Internet AS topology, he or she may also be interested in the sensitivity of their protocol to other topologies that follow the same $2K$-distribution. So, it might not be sufficient to generate only $2K$-random graphs. For a thorough evaluation of the protocol sensitivity, we need to generate a wide range of $2K$-graphs.

To explore structural diversity among all dK-graphs, we must generate dK-graphs that are not random. These non-random dk-graphs are still constrained by $\mathcal{P}_d$ but have extremely low probabilities of being generated unperturbed by dK-graph constructing algorithms.

Now, we cannot construct all possible dK-graphs, so we need to use heuristics to generate some dK-graphs and adjust them according to a distance metric that draws us closer to the types of dK-graphs we seek. One such heuristic is based on the inclusion feature of the dK-series. Because all dK-graphs have the same values of dK- but not of $(d+1)K$-distributions, we look for simple metrics fully defined by $\mathcal{P}_{d+1}$ but not by $\mathcal{P}_d$. While identifying such metrics can be challenging for high d's, we can always retreat to the following two simple extreme metrics:

- the correlation of degrees of nodes located at distance d;
- the concentration of d-simplices (cliques of size $d+1$).

These metrics are "extreme" in the sense that they correspond to the $(d+1)$-sized subgraphs with, respectively, the maximum (d) and minimum (1) possible diameter. We can then construct dK-graphs with extreme values, *e.g.*, the smallest or largest possible, for these (extreme) metrics. The dK-random graphs have the values of these metrics lying somewhere in between the extremes.

If the goal is to find the smallest d that results in sufficiently constraining graphs, we can compute the difference between the extreme values of these metrics, as well as of other metrics we might consider. If this difference is too large, then the selected value of d is not constraining enough and we will need to increase it. dK-space exploration may further be used to move beyond the relatively small circle of dK-random graphs and generate graphs that lie on the edges of the dK-circle.

To illustrate this approach in practice, we consider $1K$- and $2K$-space explorations. For $1K$, the simplest metric defined by $\mathcal{P}_2$ is any scalar summary statistics of the $2K$-distribution, such as likelihood S (cf. Section 2.1). To construct graphs with the maximum value of S, we can run a form of targeting $1K$-preserving rewiring that accepts each rewiring step only if it increases S. We can perform the opposite to minimize S. This type of experiment was at the core of recent work that led the authors of [54] to conclude that $d = 1$ was not constraining enough for the topology they considered.

To perform $2K$-space explorations, we need to find simple scalar metrics defined by $\mathcal{P}_3$. Since the $3K$-distribution is actually two distributions, $P_\wedge(k_1, k_2, k_3)$ and $P_\triangle(k_1, k_2, k_3)$, we should have two independent scalar metrics. The *second-order likelihood* S_2 is one such metric for $P_\wedge(k_1, k_2, k_3)$.
$S_2 \sim \sum_{k_1,k_2,k_3} k_1 k_3 P_\wedge(k_1, k_2, k_3)$; we define S_2 as the sum of the products of degrees of nodes located at the ends of wedges, so that any two graphs with the same $P_\wedge(k_1, k_2, k_3)$ have the same S_2. For the $P_\triangle(k_1, k_2, k_3)$ component, average clustering $\bar{C} \sim \sum_{k_1,k_2,k_3} k_1 P_\triangle(k_1, k_2, k_3)$ is an appropriate candidate. We note that these two metrics are also the two extreme metrics in the sense defined above: S_2 measures the properly normalized correlation of degrees of nodes located at distance 2, while $\bar{C}$ describes the concentration of 2-simplices (triangles). The $2K$-explorations amount then to performing the following two types of targeting $2K$-preserving rewiring: accept a $2K$-rewiring step only if it maximizes or minimizes: 1) S_2, or 2) $\bar{C}$. Figure 4.4 illustrates the process of $2K$-graph space exploration. The $2K$-random graphs, with unbiased, random values of clustering and second-order likelihood, are in the center, while maximally non-random graphs lie at the periphery of the $2K$ circle. Specifically, the bottom point in the $2K$ circle represents the set of $2K$ graphs whose S_2 has been minimized. The top point in the $2K$ circle represents the set of graphs whose S_2 has

Table 4.2: Scalar graph metrics notations.

Metric	Notation
Average degree	$\bar{k}$
Assortativity coefficient	r
Average clustering	$\bar{C}$
Average distance	$\bar{d}$
Standard deviation of distance distribution	σ_d
Second-order likelihood	S_2
Smallest eigenvalue of the Laplacian	λ_1
Largest eigenvalue of the Laplacian	λ_{n-1}

been maximized. Note that for both these points, the $\bar{C}$ value is still the same as that of $2K$-random graphs. The left and right points in the circle correspond to the set of $2K$ graphs whose clustering $\bar{C}$ has been maximized or minimized respectively.

4.4 Evaluation

We conducted a number of experiments to demonstrate the ability of the dK-series to capture important graph properties. We implemented all the dK-graph-constructing algorithms from Section 4.3.1, applied them to both measured and modeled Internet topologies, and calculated all the topology metrics from Section 2.1 on the resulting graphs.

We experimented with the three measured AS-level topologies described in the previous chapter, namely - *skitter* traceroute [13] from CAIDA, RouteViews' *BGP* [83], and RIPE's *WHOIS* [44] data for the month of March 2004, as well as with a synthetic router-level topology—the HOT graph from [54]. The qualitative results of our experiments are similar for the skitter and BGP topologies, while the WHOIS topology lies somewhere in-between the skitter/BGP and HOT topologies. In the case of skitter comprising of 9204 nodes and 28959 edges, we will see that its degree distribution itself places significant constraints upon the graph generation process. Thus, even $1K$-random graphs approximate the skitter topology reasonably well. The HOT topology with 939

Table 4.3: Scalar metrics for $2K$-random HOT graphs generated using different techniques.

Metric	Stochastic	Pseudograph	Matching	$2K$-random	$2K$-targeting	Original HOT
$\bar{k}$	2.87	2.19	2.22	2.18	2.18	2.10
r	-0.22	-0.24	-0.21	-0.23	-0.24	-0.22
$\bar{d}$	4.99	6.25	6.22	6.32	6.35	6.81
σ_d	0.85	0.75	0.74	0.70	0.70	0.57

Table 4.4: GCC sizes of $2K$-random graphs generated using different techniques.

Metric	Based on	Matching	$2K$-randomizing rewiring	Pseudo-graph	$2K$-targeting rewiring	Stoch-astic	Orig. graph
nodes	skitter	9200	9198	9200	9200	7777	9200
edges	skitter	28,848	28,957	28,955	28,957	26,609	28959
nodes	HOT	740	819	775	807	665	939
edges	HOT	821	893	850	880	954	989

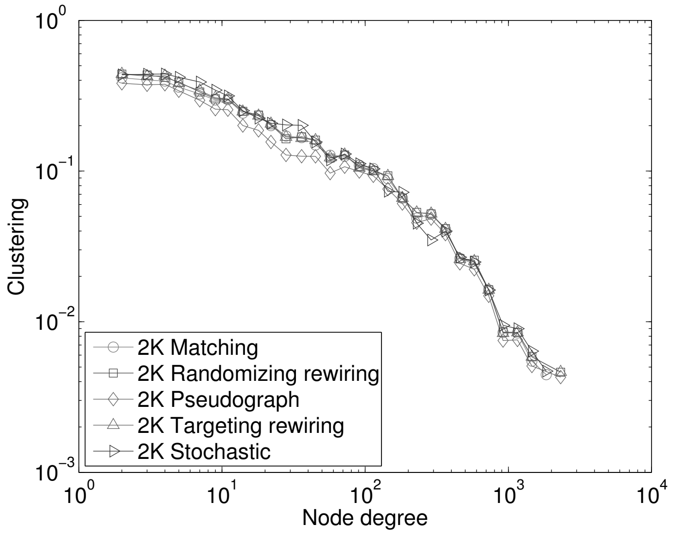
nodes and 988 edges is at the opposite extreme. It is the least constrained; $1K$-random graphs approximate it poorly, and dK-series' convergence is slowest. We report results for these two extreme cases, skitter and HOT.

Our results represent averages over 100 graphs generated with a different random seed in each case, using the notation in Table 4.2.

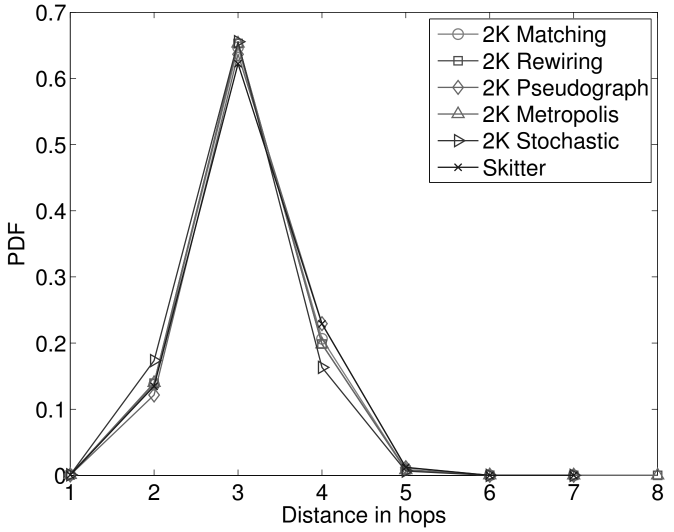
4.4.1 Algorithmic comparison

We first compare the different graph generation algorithms discussed in Section 4.3.1. All the algorithms give consistent results, except the stochastic approach, which suffers from the problems related to high statistical variance discussed in Section 4.3.1. This conclusion immediately follows from Figure 4.6, Figure 4.5, Table 4.3 and Table 4.6 showing graph metric values for the different $2K$ and $3K$ algorithms described in Section 4.3.1.

In our experience, we find that dK-randomizing rewiring is easiest to use.

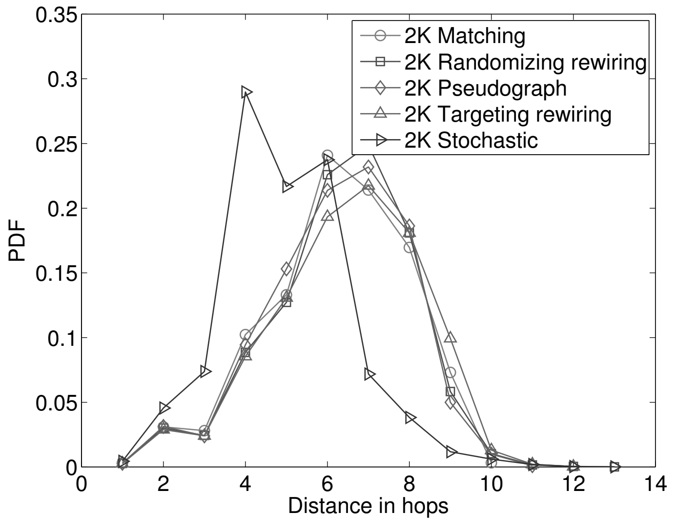


(a) Clustering in skitter for different $2K$ algorithms

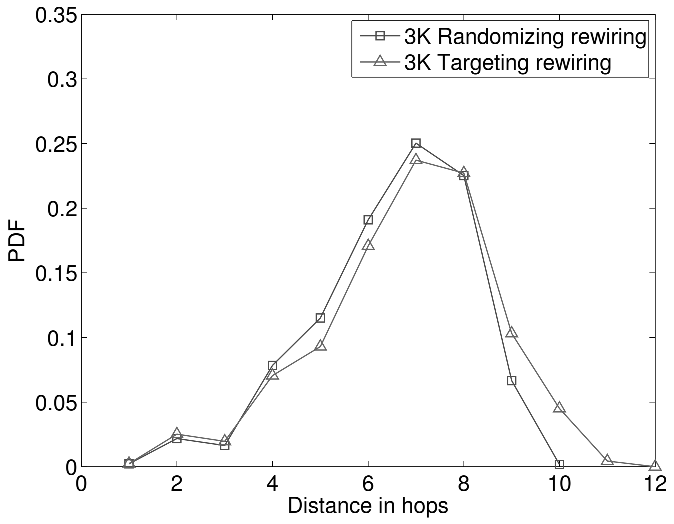


(b) Distance distribution in skitter for different $2K$ algorithms

Figure 4.5: Comparison of $2K$-graph-constructing algorithms in skitter.



(a) Distance distribution in HOT for different $2K$ algorithms



(b) Distance distribution in HOT for different $3K$ algorithms

Figure 4.6: Comparison of $2K$- and $3K$-graph-constructing algorithms.

Table 4.5: GCC sizes of $3K$-random graphs generated using different techniques.

Based on	$3K$-randomizing rewiring	$3K$-targeting rewiring	Original graph
skitter	9200	9200	9200
HOT	939	886	939

Table 4.6: Scalar metrics for $3K$-random HOT graphs generated using different techniques.

Metric	$3K$-randomizing rewiring	$3K$-targeting rewiring	Original HOT
k	2.10	2.13	2.10
r	-0.22	-0.23	-0.22
$\bar{d}$	6.55	6.79	6.81
σ_d	0.84	0.72	0.57

However, it requires the original graph as input. If only the target dK-distribution is available and if $d \leqslant 2$, we find the pseudograph algorithm most appropriate in practice. We note that our $2K$ version results in fewer pseudograph "badnesses", i.e., (self-)loops and small connected components (CCs), than PLRG [2], its commonly-known $1K$ counterpart. This improvement is due to the additional constraints introduced by the $2K$ case. For example, if there is only one node of high degree x and one node of another high degree y in the original graph, then there can be only one link of type (x, y). Our $2K$ modification of the pseudograph algorithm must consequently produce exactly one link between these two x- and y-degree nodes, whereas in the $1K$ case, the algorithm tends to create many such links. Similarly, a $1K$ generator would tend to produce many isolated degree-1 nodes connected to one another. Since the original graph does not have such pairs of 1-degree nodes, our $2K$ generator, as opposed to $1K$, does not form these small 2-node CCs either.

While the pseudograph algorithm is a good $2K$-random graph generator, we could not generalize it for $d \geqslant 3$ (as explained in Section 4.3.1). Therefore, to generate dK-random graphs with $d \geqslant 3$ when an original graph is unavailable, we use dK-targeting rewiring. We first bootstrap the process by constructing $1K$-random graphs

Table 4.7: Numbers of possible initial dK-randomizing rewirings for the HOT graph.

d	Possible initial rewirings	Possible initial rewirings, ignoring obvious isomorphisms
0	435,546,699	-
1	477,905	440,355
2	326,409	268,871
3	146	44

Table 4.8: Comparing scalar metrics for dK-random and skitter graphs.

Metric	$0K$	$1K$	$2K$	$3K$	skitter
k	6.31	6.34	6.29	6.29	6.29
r	0	-0.24	-0.24	-0.24	-0.24
$\bar{C}$	0.001	0.25	0.29	0.46	0.46
$\bar{d}$	5.17	3.11	3.08	3.09	3.12
σ_d	0.27	0.4	0.35	0.35	0.37
λ_1	0.2	0.03	0.15	0.1	0.1
λ_{n-1}	1.8	1.97	1.85	1.9	1.9

using the pseudograph algorithm and then apply $2K$-targeting $1K$-preserving rewiring to obtain $2K$-random graphs. To produce $3K$-random graphs, we apply $3K$-targeting $2K$-preserving rewiring to the $2K$-random graphs obtained in the previous step.

4.4.2 Topology comparison

We next test the convergence of our dK-series for the skitter and HOT graphs. Since all dK-graph constructing algorithms yield consistent results, we selected the simplest one, the dK-randomizing rewiring from Section 4.3.1, to obtain dK-random graphs in this section.

The number of possible initial dK-randomizing rewirings is a good preliminary indicator of the size of the dK-graph space. We show these numbers for the HOT graph in Table 4.7. If we discard rewirings leading to obvious isomorphic graphs, cf. Section 4.3.1, then the number of possible initial rewirings is even smaller.

We compare the skitter topology with its dK-random counterparts,

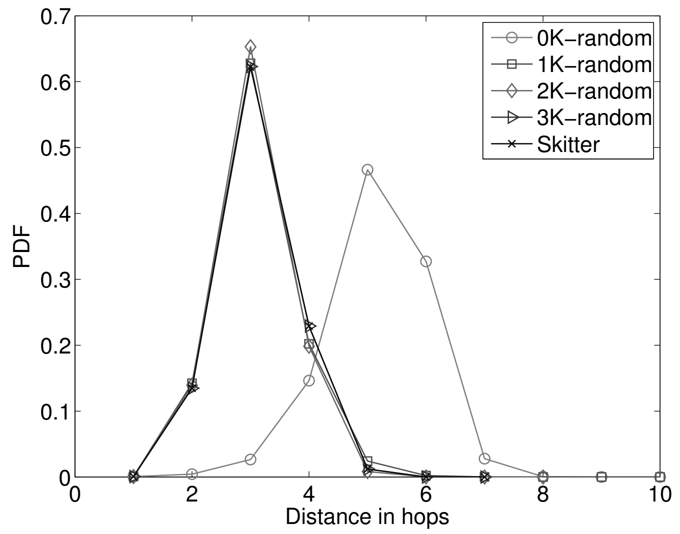


Figure 4.7: Comparison of distance distribution in dK-random and skitter graphs.

$d = 0, \ldots, 3$, in Table 4.8 and 4.4.2. We report all the metrics calculated for the giant connected component (GCCs). Minor discrepancies between values of average degree $\bar{k}$ and r result from GCC extractions. If we do not extract the GCC, then $\bar{k}$ is the same as that of the original graph for all $d = 0, \ldots, 3$, and r is exactly the same for $d > 1$. The degree distributions match when considering the entire graph and are very similar for the GCCs for all $d > 0$. When $d = 0$, the degree distribution is binomial, as expected.

We see that all other metrics gradually converge to those in the original graph as d increases. A value of $d = 1$ provides a reasonably good description of the skitter topology, while $d = 2$ matches all properties except clustering. The $3K$-random graphs are identical to the original for all metrics we consider, including clustering.

We perform the $2K$-space explorations described in Section 4.3.3 to check if $d = 2$ is indeed sufficiently constraining for the skitter topology. We observe small variations of clustering $\bar{C}$, second-order likelihood S_2, and spectrum, shown in Table 4.9 and Figure 4.10. All other metrics do not change thereby confirming that $d = 2$, i.e., the joint degree distribution, indeed provides a reasonably accurate description of observed AS-level topologies.

The HOT topology is more complex than AS-level topologies. Earlier work

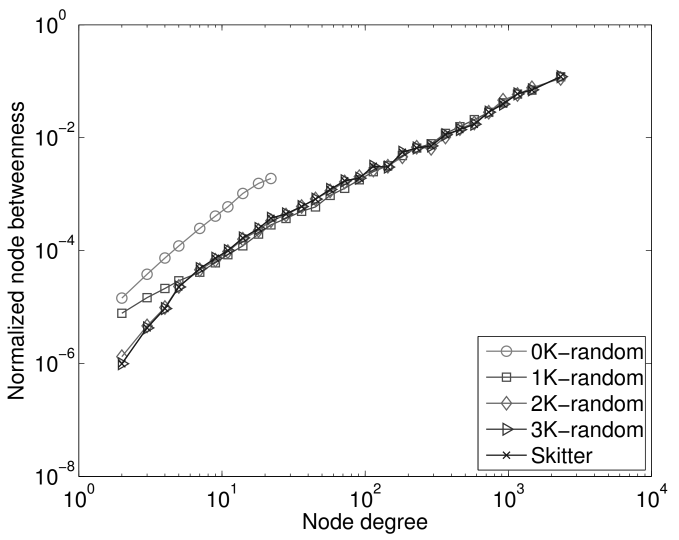


Figure 4.8: Betweenness in dK-random and skitter graphs.

Table 4.9: Scalar metrics for $2K$-space explorations for skitter.

Metric	Min $\bar{C}$	Max $\bar{C}$	Min S_2	Max S_2	$2K$-random	Skitter
k	6.29	6.29	6.29	6.29	6.29	6.29
r	-0.24	-0.24	-0.24	-0.24	-0.24	-0.24
$\bar{C}$	0.21	0.47	0.4	0.4	0.29	0.46
$\bar{d}$	3.06	3.12	3.12	3.10	3.08	3.12
σ_d	0.33	0.38	0.37	0.36	0.35	0.37
λ_1	0.25	0.11	0.11	0.1	0.15	0.1
λ_{n-1}	1.75	1.89	1.89	1.89	1.85	1.9
$S_2/S_2^{\max}$	0.988	0.961	0.955	1.000	0.986	0.958

argues that this topology cannot be accurately modeled using degree distributions alone [54]. We therefore selected the HOT topology graph as a difficult case for our approach.

A preliminary inspection of visualizations in Figure 4.2 indicates that the dK-series converges at a reasonable rate even for the HOT graph. The $0K$-random graph is a classical random graph and lacks high-degree nodes, as expected. The $1K$-random graph has all the high-degree nodes we desire, but they are crowded toward the core, a property absent in the HOT graph. The $2K$ constraints start pushing the high-degree

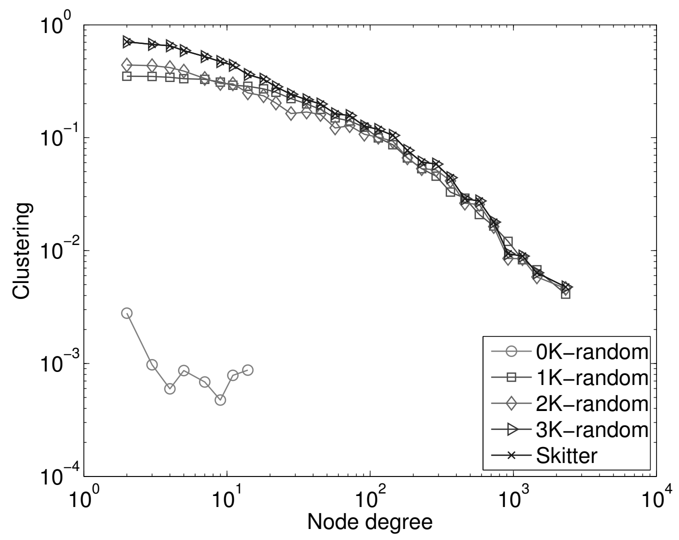


Figure 4.9: Clustering in dK-random and skitter graphs.

nodes away to the periphery, while the lower-degree nodes migrate to the core, and the $2K$-random graph begins to resemble the HOT graph. The $3K$-random topology looks remarkably similar to the HOT topology.

Of course, visual inspection of a small number of randomly generated graphs is insufficient to demonstrate our ability to capture important metrics of the HOT graph. Thus, we compute the different metric values for each of the dK-random graph and compare them with the corresponding value for the original HOT graph. In Table 4.10, Figure 4.11(a) and Figure 4.11(b) we see that the dK-series converges more slowly for HOT than for skitter. Note that we do not show clustering plots because clustering is almost zero everywhere: the HOT topology has very few cycles; it is almost a tree. The $1K$-random graphs yield a poor approximation of the original topology, in agreement with the main argument in [54]. Both Figure 4.2 and Figure 4.11(b) indicate that starting with $d = 2$, low- but not high-degree nodes form the core: betweenness is approximately as high for nodes of degree ~ 10 as for high-degree nodes. Consequently, the $2K$-random graphs provide a better approximation, but not nearly as good as it was for skitter. However, the $3K$-random graphs match the original HOT topology *almost exactly*. We thus conclude that the dK-series captures the essential characteristics of even particularly

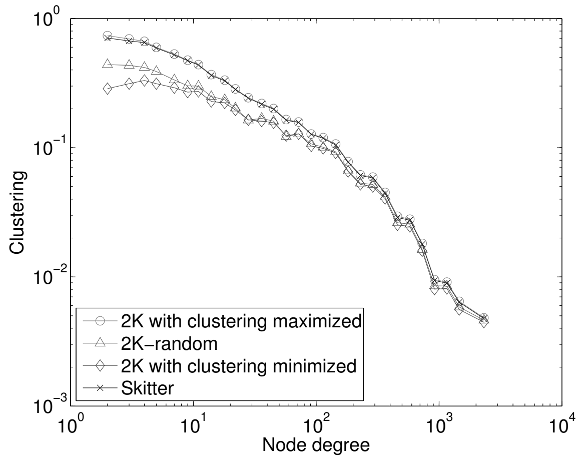


Figure 4.10: Varying clustering in $2K$-graphs for skitter.

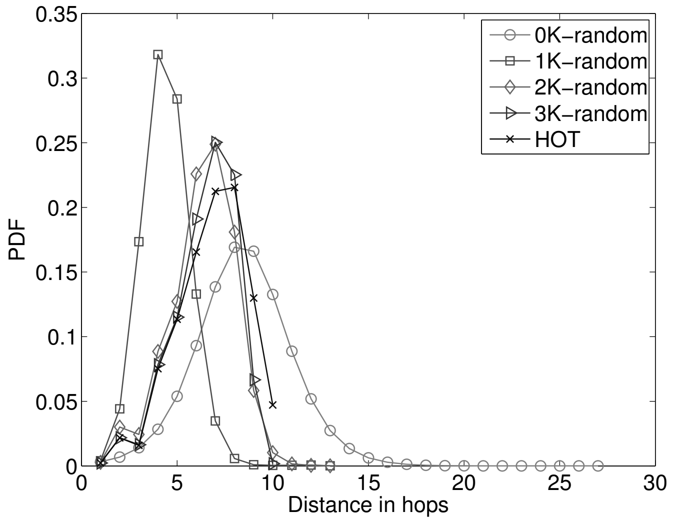
Table 4.10: Comparing scalar metrics for dK-random and HOT graphs.

Metric	$0K$	$1K$	$2K$	$3K$	HOT
$\bar{k}$	2.47	2.59	2.18	2.10	2.10
r	-0.05	-0.14	-0.23	-0.22	-0.22
$\bar{C}$	0.002	0.009	0.001	0	0
$\bar{d}$	8.48	4.41	6.32	6.55	6.81
σ_d	1.23	0.72	0.71	0.84	0.57
λ_1	0.01	0.034	0.005	0.004	0.004
λ_{n-1}	1.989	1.967	1.996	1.997	1.997

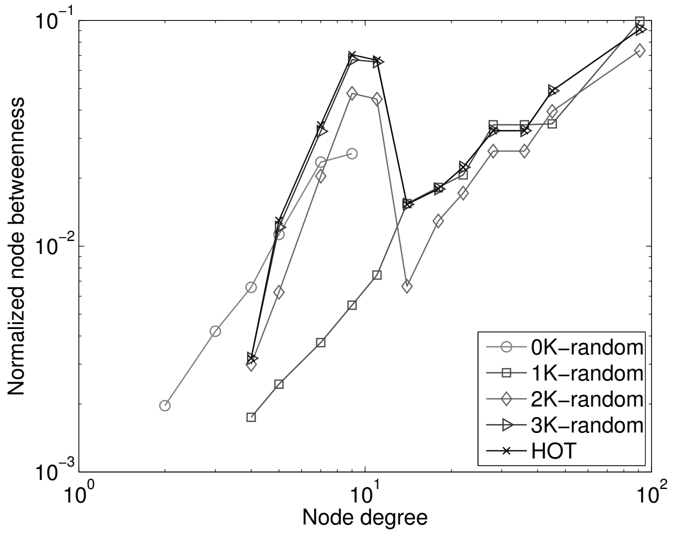
difficult topologies, such as HOT, by sufficiently increasing d, in this case to 3. We note that the speed of dK-series convergence depends both on the structure and size of an original graph G. It must converge faster for smaller graphs of similar structure. However, here we see that the graph structure plays a more crucial role than its size. The dK-series converges slower for HOT than for skitter, even though the former graph is an order of magnitude smaller than the latter.

4.5 Discussion and Summary

While our approach to topology analysis is promising, we point out a few relevant issues in using the dK-series.



(a) Distance distribution



(b) Betweenness

Figure 4.11: Comparison in dK-random and HOT graphs

First, one must determine appropriate values of d to carry out studies of interest. Our experience to date suggests that $d = 2$ is sufficient to reproduce most metrics of interest and that $d = 3$ faithfully reproduces all metrics we are aware of for Internet-like graphs. It also appears likely that $d = 3$ will be sufficient for self-organized small-worlds in general. This issue is particularly important because the computational complexity of producing dK-graphs grows rapidly with d. Studies requiring large values of d may limit the practicality of our approach. However, we are encouraged by our initial experiments with non-Internet graphs such as protein networks, social networks and other biological networks. All of these graphs are small-world and $d = 3$ was sufficient to reproduce all the metrics for these graphs as well.

In general, more complex topologies may necessitate developing algorithms for generating dK-random graphs with high d's. We needed higher d to describe the HOT topology as accurately as the skitter topology. The intuition behind this observation is that the HOT router-level topology is "less random" because it results from targeted design and engineering. The skitter AS-level topology, on the other hand, is "more random" since there is no single point of external human control over its shape and evolution. It is a cumulative result of local decisions made by individual ASes.

A second important question concerns the discrete nature of our model. For instance, we are able to reproduce $1K$- and $2K$-distributions but it is not meaningful to consider reproducing $1.4K$-distributions. Consider a graph property X not captured by $1K$ but successfully captured by $2K$. It could turn out that the space of $2K$-random graphs over-constrains the set of graphs reproducing X. That is, while $2K$-graphs do successfully reproduce X, there may be other graphs that also match X but are not $2K$-graphs.

Fundamental to our approach is that we seek to reproduce important characteristics of a *given* network topology. We cannot use our methodology to discover laws governing the evolutionary growth of a particular network. Rather, we are restricted to observing changes in degree correlations in graphs over time, and then generating graphs that match such degree correlations. However, the goals of reproducing important characteristics of a given set of graphs and discovering laws governing their evolution are complementary and even symmetric.

They are complementary because the dK-series can simplify the task of validating particular evolutionary models. Consider the case where a researcher wishes to validate a model for Internet evolution using historical connectivity information. The process would likely involve starting with an initial graph, *e.g.*, reflecting connectivity from 5 years ago, and generating a variety of larger graphs, *e.g.*, reflecting modern-day connectivity. Of course, the resulting graphs will not match known modern connectivity exactly. Currently, validation would require showing that the graph matches "well enough" for all known ad hoc graph properties. Using the dK-series however, it is sufficient to demonstrate that the resulting graphs are dK-random for an appropriate value of d, i.e., constrained by the dK-distribution of modern Internet graphs (with $d = 3$ known to be sufficient in this case). As long as the resulting graphs fall in the dK-random space, the theory of dK-randomness explains any variation from ground truth. This methodology also addresses the issue of defining "well enough" above: dK-space exploration can quantify the expected variation in ad hoc properties not fully specified by a particular dK-distribution.

The approaches are symmetric in that they both attempt to generate graph models that accurately capture values of topology metrics observed in real networks. Both approaches have inherent tradeoffs between accuracy and complexity. Achieving higher accuracy with the dK-series requires greater numbers of statistical *constraints* with increasing d. The number of these constraints is upper-bounded by n^d (the size of dK-distribution matrices) times the number of possible simple connected d-sized graphs [89]. Although the upper bound of possible constraints increases rapidly, sparsity of dK-distribution matrices increases even faster. The result of this interplay is that the number of non-zero elements of dK-distributions for any given G increases with d first but then quickly decreases, and it is surely 1 in the limit of $d = n$ (the example in Section 4.2). Achieving higher accuracy with network evolution modeling requires richer sets of system-specific external parameters [17]. Every such parameter represents a *degree of freedom* in a model. By tuning larger sets of external parameters, one can more closely match observed data. It could be the case that the number of parameters required to characterize the evolution of the Internet is smaller than the number of constraints required by the dK-series (this remains to be seen). However, with the dK-series, the same set of constraints applies to any network, including social, biological, and physical. With evolution modeling, one must develop a separate model for each network.

In summary, we define a series of graph structural properties to both systematically characterize arbitrary graphs and to generate random graphs that match specified characteristics of the original. The dK-distribution is a collection of distributions describing the correlations of degrees of d connected nodes in nay given graph. The properties $\mathcal{P}_d$, $d = 0, \ldots, n$, comprise the dK-series. A random graph is said to have property $\mathcal{P}_d$ if its dK-distribution has the same form as in a given graph G. By increasing the value of d in the series, it is possible to capture more complex properties of G and, in the limit, a sufficiently large value of d yields complete information about G's structure. We show that $0K$- and $1K$-graphs are two known graph classes corresponding to special instances of our properties: average node degree and degree distribution, respectively. $2K$-graphs capture relationships among pairs of nodes of given degrees; $3K$-graphs capture such relationships among node triples of given degrees, and so on. For all Internet graphs (both AS- and router-level) that we consider, we find that the $d = 2$ case is sufficient for most practical purposes, while $d = 3$ essentially reconstructs the Internet AS- and router-level topologies almost exactly. All the generated dK-graphs in this chapter are the same size as the original graph. We describe our techniques to rescale graph sizes using dK-distributions in the next chapter.

4.6 Acknowledgments

Chapter 4, in full, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Conference, Pisa, Italy, September 2006, Mahadevan, Priya;

Krioukov, Dmitri; Fall, Kevin; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Rescaling Topologies

Based on our analysis of a wide variety of AS-graphs in Chapter 3, we put forth a series of unifying graph properties, *the dK-series*, and demonstrate how they progressively capture increasingly finer graph metrics by incorporating more and more information about connectivity among increasingly larger node neighborhoods. Using the distributions associated with the dK-series, we construct dK-random graphs and find that for a wide range of measured and modeled Internet topologies, including comparatively complex AS-level and router-level graphs, we have been able to successfully reproduce virtually all known graphs metrics in the literature. The dk-series, by its definition, also guarantees to reproduce any future graph metric that may be proposed.

However, the algorithms to generate dK-random graphs described in the previous chapter are only capable of generating graphs of the same size as the given graph. Scaling graphs to a range of sizes is important as an enabler for a range of simulation and emulation studies and is an important requirement for any topology generator. We now look at several strategies for generating graphs of different sizes using the dK-series.

The goal of our rescaling algorithms is to generate random graphs that maintain important local characteristics about graph interconnectivity while appropriately scaling up or down global graph characteristics (*e.g.*, maximum degree). Ideally, our rescaling algorithms should be generic to arbitrary topology classes and applicable to, for instance, biological, social, or physical networks.

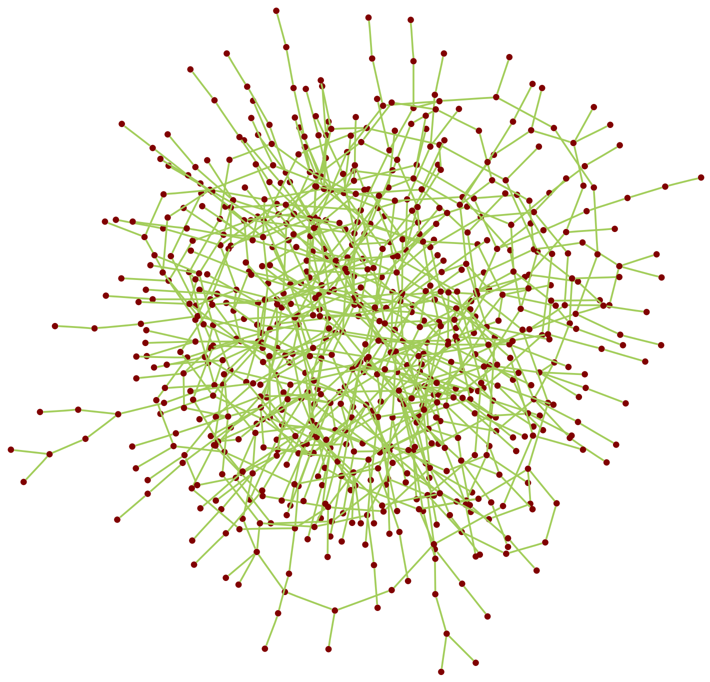
To gain more insight about the connectivity properties, we first study historic data for AS-level graphs collected over five years. Additionally, we also examine router-level topologies of various sizes for thousands of ASes. The historic data provide insight into how the structure of the graph changes as the network grows. We are interested to see whether measures of local node interconnectivity changes as the graph grows globally. Based on the results we obtain from historical data analysis, we identify several rescaling algorithms. To evaluate our success in rescaling, we compare the original to the generated graph for a range of commonly used topology metrics described in Chapter 2. We discuss these techniques and results in the rest of this chapter.

5.1 Challenges in Rescaling

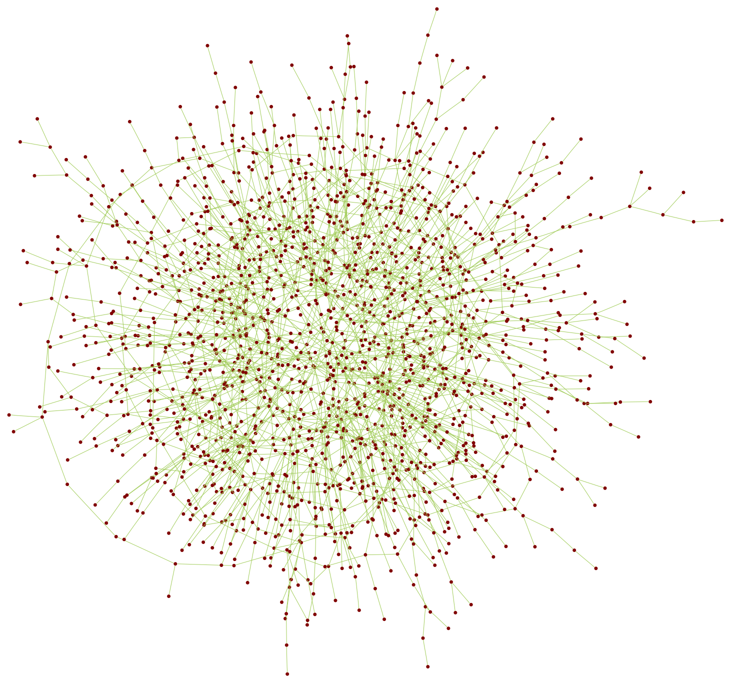
From the experiments in the previous chapter, $2K$-distribution for an input graph can reproduce virtually all important graph metrics proposed in the literature for Internet graphs. In this chapter, we focus on algorithms to generate $1K$-random and $2K$-random graphs with variable numbers of nodes given a target distribution for some fixed-sized graph. Rescaling $0K$-random graphs is straightforward because the $0K$-distribution consists of a single scalar describing the average degree of nodes in the graph; it is independent of topology size.

Consider the $1K$ case however. The $1K$-distribution corresponds to a given graph's degree distribution. It is a function of one integer variable, i.e., node degree k. The support[1] of $P(k)$ lies within $[0, n-1]$, $0 \leqslant \mathrm{supp}(P) \leqslant n-1$, while the values of $P(k)$ are between 0 and 1. If we wish to generate a random graph of a different size n', then the main question is how the support and values of $P(k)$ should change to result in the appropriately rescaled degree distribution $P'(k')$ of the new graph. Consider the case of scaling a graph from 1000 to 2000 nodes.Should the maximum degree k'_{max} in the new degree distribution scale with the ratio n'/n? How should the degrees in the given graph as well as their corresponding distribution be rescaled in the new graph? The question

[1]The support $\mathrm{supp}(f)$ of function $f(x)$ is the set of values of its argument x such that $f(x) \neq 0$.

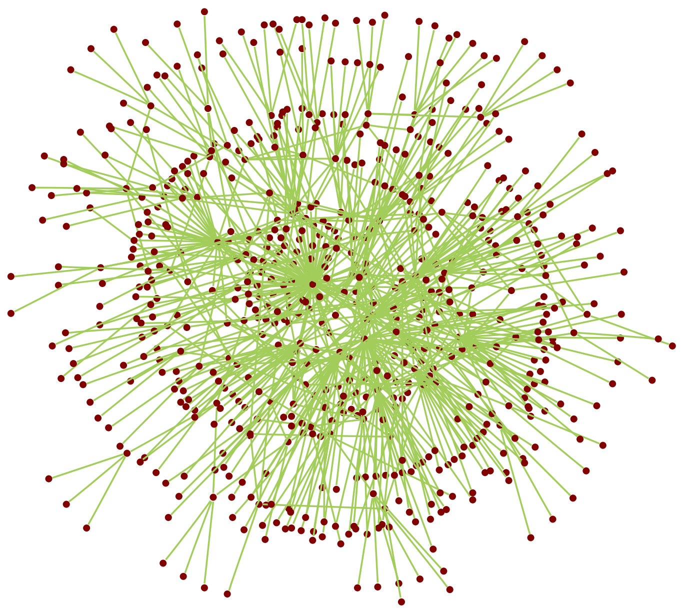


(a) Non-rescaled $0K$-graph

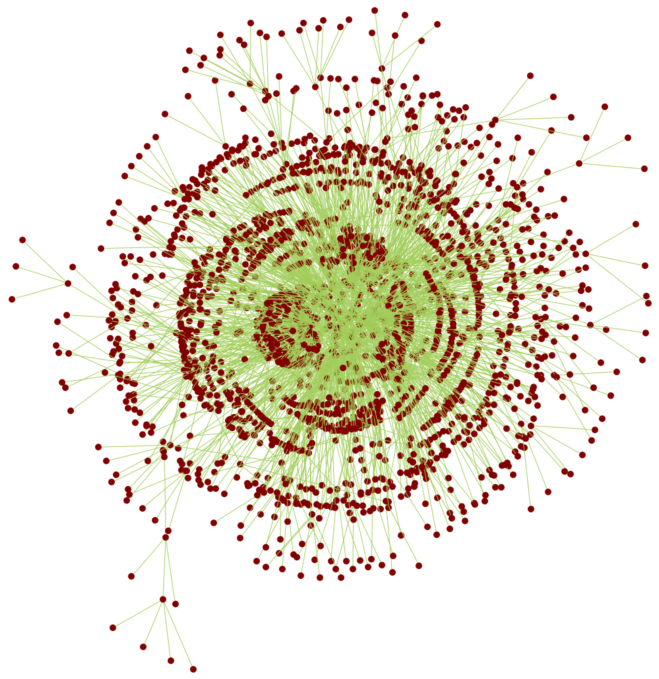


(b) Rescaled $0K$-graph, 2000 nodes

Figure 5.1: Rescaled and non-rescaled $0K$-graph

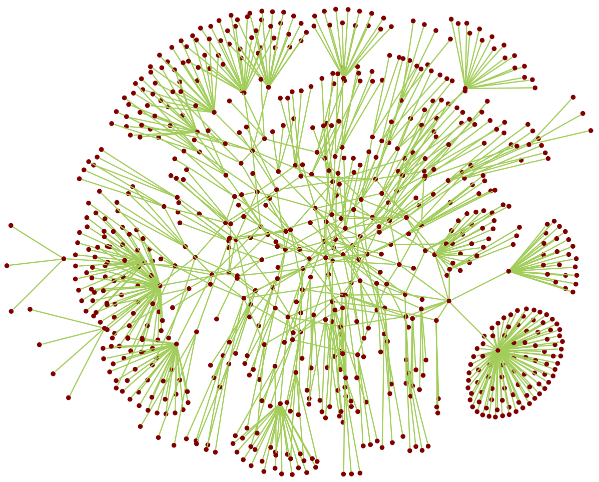


(a) Non-rescaled $1K$-graph

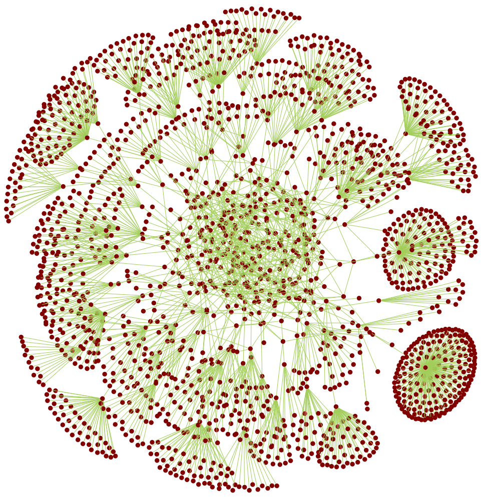


(b) Rescaled $1K$-graph, 2000 nodes

Figure 5.2: Rescaled and non-rescaled $1K$-graph



(a) Non-rescaled $2K$-graph



(b) Rescaled $2K$-graph, 2000 nodes

Figure 5.3: dK-graphs of size 2000 and the original HOT graph (939 nodes) illustrating rescaling

becomes more complex when considering, for instance, $2K$-distributions $P(k_1, k_2)$ that are functions of two arguments.

To motivate how our rescaling works in practice, we present visualizations of original sized and rescaled graphs versions of the HOT [54] graph starting with the initial $0K$, $1K$ and $2K$ in Figure 5.1, Figure 5.2 and Figure 5.3 respectively. While we will quantify our ability to reproduce important graph metrics in subsequent sections, visually we see that the rescaled graphs maintain much of the same interconnectivity and structure of their original-sized counterparts.

There are a number of approaches for scaling an initial degree distribution for a new graph of a given size. Most of them unfortunately reduce to ad hoc mathematical transformations that involve either compressing the existing distribution or spreading it out over larger number of possible degrees. In all cases, these transforms require an assumption about how a graph's maximum degree should scale with the number of nodes in the graph. While such mathematical transformations are feasible, we believe that the appropriate rescaling approach depends on the characteristics of the class of graphs being considered. That is, individual types of graphs will scale up and down in an application-specific manner. The way that Internet graphs grow may very well be different from how social or biological networks grow. Similarly, growth characteristics for a given organization's router topology may well differ from the growth characteristics of the Internet's global AS peering graph.

5.1.1 Internet topology input data

AS topology historic data

Since we are interested in building an Internet router topology generator that also combines AS information for each router, we decided to study the historical growth characteristics of Internet graphs. In particular, we considered the dK-distributions for historical AS-level topologies extracted from skitter [48] and RouteViews [83] data in March of each year between 2000 and 2005. In particular, the AS-level topology ob-

Table 5.1: Scalar metric values for historic skitter AS-level topologies.

Year	Nodes	$\bar{k}$	k_{max}	r	$\bar{C}$	$\bar{d}$	σ_d
2000	3308	5.77	836	-0.25	0.38	3.14	0.41
2001	6021	5.48	1461	-0.22	0.40	3.21	0.42
2002	8359	6.49	2355	-0.24	0.44	2.99	0.35
2003	8512	5.17	1443	-0.23	0.38	3.30	0.43
2004	9204	6.23	2070	-0.24	0.46	3.12	0.37
2005	8500	5.97	1783	-0.23	0.45	3.17	0.39

tained from skitter grew by almost a factor of 3 during this time period.

Our hypothesis was that the graph would demonstrate some steady (rather than ad hoc) growth characteristics and that we could then apply our understanding of the Internet's AS growth to generating graphs of a range of sizes given some initial dK distribution. While we only present results for the skitter data, the conclusion from the RouteViews data is statistically similar. Figure 5.4 plots degree distribution and Table 5.1 presents some of the commonly used graph metrics for skitter data during this time period.

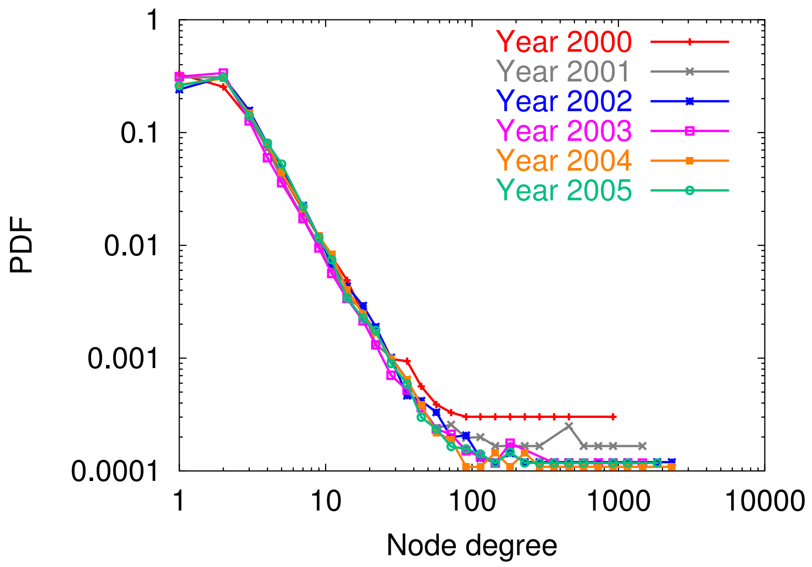


Figure 5.4: PDF of degree distribution observed in historical skitter data

From the historic skitter data we make the following observations:

1. As is well-known, the degree distribution follows a power-law, $P(k) \sim k^{-\gamma}$, with

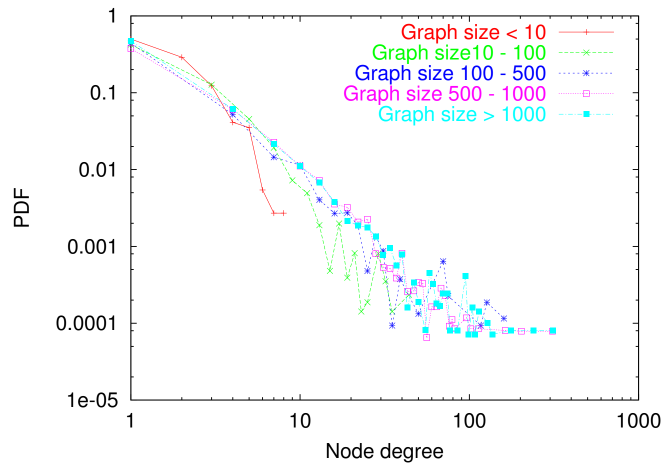


Figure 5.5: PDF of degree distribution observed in skitter router-level topology of different sizes

$\gamma \approx 2.1$, and the maximum degree k_{max} scales almost linearly with the graph size.[2]

2. Up to a certain threshold degree, the values of the degree distribution stay the same and the power-law exponent γ remains unchanged with the evolution of the topology. The average degree stays the same due to linear scaling of values of high degrees in the power-law tail.

3. The assortativity coefficient, a scalar summary of the $2K$-distribution, and mean clustering, a partial summary of the $3K$-distribution, remain almost constant over the five year period.

4. Some other global metrics such as the distance distribution, do not drastically change either.

Router topology data

Next, we consider the growth characteristics of router topologies within individual ASes. Unlike AS-level topologies, we do not have access to detailed historic

[2]In fact, the expected maximum among n samples of a random variable distributed according to a power law with exponent $-\gamma$ is $k_{\max} \sim n^{1/(\gamma-1)}$ [8]. For the observed values of γ, this scaling is almost linear since $1/(2.1-1) \approx 0.9$.

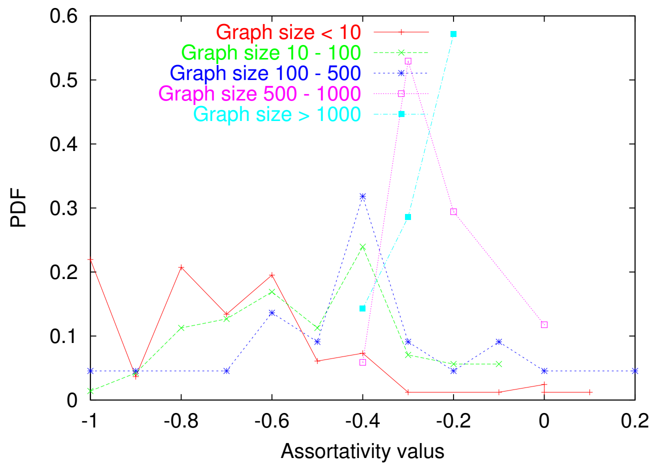


Figure 5.6: PDF of assortativity observed in skitter router-level topology of different sizes

router-level topologies to understand their growth patterns. Instead we gathered router-level topologies for all the ASes observed in the skitter traceroute data from September 1- 15, 2006.

We first present details of skitter's router-level traces. Skitter is a traceroute based tool; 25 monitors strategically located around the globe send traceroute probes to more than a million destination addresses. Skitter records each hop from the source monitors to all the destinations by incrementing the "time to live" (TTL) of each IP packet header and recording replies from each router leading to the destination host. From the collected traceroute packets, the corresponding router-level topology is obtained by the following steps:

1. From the traceroute traces, we first extract IP links. Using the RouteViews data, we assign the source AS of the longest matching prefix to each IP link.

2. Interface aliases are merged using `iffinder` [42] into routers. `iffinder` sends UDP probe packets to all IP addresses seen in the traces with destination UDP ports set to presumably unused values. If router R receives such a packet from prober P destined to R's IP interface X, while R's route to P goes via some other R's IP interface Y, then R is supposed to reply to P with ICMP `port`

`unreachable` message with the source address set to Y. Prober P can thus conclude that X and Y belong to the same router [39].

3. We translate IP links from step 1 to router links using the results from step 2. Next, we assign AS numbers corresponding to the IP links to the router links. Since all traceroute traces are directed from source to destination, the graph obtained at this step is also directed, *i.e.*, each node is characterized by its in- and out-degree.

4. We discard all nodes with either in-degree 0 or out-degree 0 as majority of nodes removed this way are end hosts.

After applying the above steps, we obtain a router-level topology of the Internet with more than 200,000 routers. This router-level topology has information on AS membership, *i.e.*, the AS number that each link in the graph belongs to. By putting together links that belong to the same AS, we are able to create a router-level topology for each of the ASes observed in the data.

The resulting topology suffers from several limitations. In particular, since skitter sends traceroute probes from the source monitors to the hosts in its destination lists, typically not all routers inside an AS will be discovered. As a result, the router-level topology for a particular AS need not be connected (disjoint traceroutes may discover different portions of the same AS). We extract the giant connected component (GCC) from the router-level topology for each AS and discard ASes where the GCC is less than 75% of the total number of routers in that AS. We consider the connectivity information for these ASes to be insufficient to inform subsequent topology generation. We are left with router-level topologies of varying sizes for approximately 5700 ASes. There are a number of other concerns with the quality of obtained topologies. We list just a few:

- Traceroute explorations are widely known to introduce sampling biases since they find only those links that, roughly, lie in a collection of shortest-path trees rooted at the monitors; other links are missing [22].

- Traceroute probes are also susceptible to degree inflation, since they do not account for layer-2 connectivity. For example, when several routers are connected through a switch, traceroute techniques can mistakenly assume that these routers are directly connected to each other. As a result, the degrees of the routers need not be accurate.

- Several concerns stem from using `iffinder` to map IP interfaces to routers. In particular, while there are no false positives in the mapping, it is difficult to quantify the false negatives.

Despite these limitations, the skitter data remains one of the few sources of router-level data that can be used for topological modeling and inference.

We next smooth the obtained router topology statistics. Specifically, we perform logarithmic binning to group all the router graphs obtained for each AS into the following four categories based on the AS size equal to the number of routers within an AS:

AS category	AS size S
T_1	$S < 10$
T_2	$10 \leqslant S < 100$
T_3	$100 \leqslant S < 1000$
T_4	$S \geqslant 1000$

We plot the degree distributions and assortativity coefficients averaged for each category in Figure 5.5 and Figure 5.6, and make the following observations:

- Router graphs with more than 100 nodes have degree distributions that can be loosely approximated by power laws, although the power law is not as good a fit as was observed for AS-level topologies. Graphs smaller than 100 nodes have such scarce degree statistics that any discussion whether their degree distributions follow any specific laws or not is impossible.

- The observed maximum degree does not scale in a similar fashion as observed in historic AS-level graphs. In fact, the maximum degree does not increase significantly with increase in graph size and the observed maximum node degree is about 330 in the largest router graphs.

This observation agrees with the intuition behind router topologies: the number of interfaces per router cannot be arbitrarily large; it is bounded by simple technical network design constraints [54].

- We observe no specific and statistically significant values of assortativity coefficients for the obtained router graphs of different sizes.

These observations suggest that our rescaling techniques should be appropriately adjusted to account for the specifics of router topologies. The maximum degree, for example, cannot scale linearly as in the AS topology case. Instead, it should be bounded above by a specific value representing current technological and practical limitations.

Based on the findings from historic AS-level topology and router-level topologies of various sizes, we implemented a rescaling algorithm to generate both $1K - random$ and $2K - random$ graphs. This algorithm, however, would only be effective for scale-free graphs such as the Internet's topologies. In other words, the scaling of biological or social networks might exhibit different patterns.

5.2 $1K$-rescaling

We base our rescaling techniques on the observations made in Section 5.1.1. In our $1K$-rescaling, we attempt to preserve the shape of the PDF of the graph's degree distribution. We do so by: i) keeping the proportion of low-degree nodes in the rescaled graph the same as in the original graph, ii) scaling linearly, with the size of a rescaled graph, the values of high degrees, and iii) keeping the number of nodes of rescaled high degrees the same as the corresponding number of nodes in the original graph.

Specifically, our algorithm to rescale $1K$-distributions of AS graphs works as follows.

Input:

- The original graph size n.

- The original degree distribution $P(k)$.
- The new graph size n'.

Output:

- The degree distribution $P'(k')$ in the new graph.

Procedure:

1. Find the low-degree threshold $k_l > 1$ defined as the lowest degree value such that the smoothed degree distribution behaves as $P(k_l + 1) > P(k_l)$, *i.e.*, value k_l is such that statistical noise in $P(k)$ becomes significant for $k > k_l$. We require that k_l be sufficiently large such that the nodes of the remaining degrees account for less than 10% of the total nodes (the degree distribution PDF does not typically follow a power law for the first few degrees and most of the nodes in the graph fall in this range).
2. Find the high-degree threshold k_h defined as the lowest degree value such that the smoothed degree distribution $P(k)$ is a constant function, *i.e.*, value k_h is such that the number of nodes $n(k) = nP(k)$ is approximately the same for all $k > k_h$.
3. Let $k \in \mathrm{supp}(P)$ be the set of degree values k such that $P(k) \neq 0$, and $k'(k) \in \mathrm{supp}(P')$ be the set of corresponding degree values in the new graph.
4. For $k \leqslant k_l$, $k'(k) = k$ and $P'(k) = P(k)$.
5. For $k \geqslant k_h$, $k'(k) = kn'/n$ and $n'(k'(k)) = n(k) \Leftrightarrow P'(kn'/n) = n/n'P(k)$.[3]
6. Let $M = (k_l, k_h)$ be an open interval between k_l and k_h, and $M' = (k_l, k_h n'/n)$ be an open interval between $k'(k_l)$ and $k'(k_h)$. For $k \in M$, we use linear rescaling to glue the two regimes of $P'(k')$ defined at steps 4 and 5 as follows:
   (a) Let u be the number of nodes in the original graph with degrees $k \in M$, $u = n\sum_{k \in M} P(k)$, let $i = 1, \ldots, u$ be the rank of node i in the list of nodes

[3]Rounding to closest integers is assumed whenever needed.

with degrees in M sorted in the order of non-increasing degrees, and let $k(i)$ be the degree of node i. This way $k(1) = k_h$ and $k(u) = k_l$.

(b) Let u' be the number of nodes in the new graph that should have degrees $k' \in M'$,
$u' = n'(1 - \sum_{k' \notin M'} P'(k'))$.

(c) For nodes $j = 1, \ldots, u'$ in the new graph, compute their linearly rescaled degree values by
$k'(j) = \left(\frac{1-n'/n}{u'-1}(j-1) + \frac{n'}{n}\right) k\left[\frac{u-1}{u'-1}(j-1)+1\right]$.

(d) Let $l(x)$ be a linear function such that $l(k'_l) = P'(k'_l)$ and $l(k'_h) = P'(k'_h)$, *i.e.*, $l(x) = ax + b$, where $a = (n/n'P(k_h n'/n) - P(k_l))/(k_h n'/n - k_l)$ and $b = P(k_l) - ak_l$. Let ρ be a small random variable uniformly and symmetrically distributed around 0. For values k' in the new graph produced by step 6c, compute the corresponding values of the degree distribution by $P'(k') = c(l(k') + \rho)$, where the constant c is determined from the normalization condition for the whole $P'(k')$, *i.e.*, $\sum_{k' \in \text{supp}(P')} P'(k') = 1$.

We then supply the output degree distribution $P'(k')$ as input to the $1K$-random topology generation algorithms described in the previous chapter to obtain the final graph. As before, we extract the GCC from the generated graph.

To rescale router topologies of sizes smaller than 100 nodes, we also maintain the degree distribution of the given graph. Rescaling router topologies larger than 100 nodes, we impose an additional constraint on the maximum degree to not exceed 330. We stress that this value can be configured by the user based on better data for Internet connectivity or updated router technology information.

5.3 $2K$-rescaling

Rescaling an original $2K$-distribution is more difficult because the $2K$-distribution contains strictly more information than the $1K$-distribution. An element

$P(k_1, k_2)$ in the $2K$-distribution matrix is the probability of having an edge between two nodes of degree k_1 and k_2. The main idea behind our $2K$-rescaling is the same as in the $1K$ case—we try to preserve the shape of the $2K$-distribution $P(k_1, k_2)$. In other words, we want the degree correlation "profile" of rescaled graphs be similar to the original. We preserve it by means of the following algorithm:

Input:

- The original graph size n.
- The original joint degree distribution (JDD) $P(k_1, k_2)$.
- The new graph size n'.

Output:

- The JDD in the new graph $P'(k'_1, k'_2)$.

Procedure:

1. Compute the $1K$-distribution from the given $2K$-distribution by $P(k) = \bar{k}/k \sum_{k_1} P(k, k_1)$.
2. Rescale $P(k)$ to $P'(k')$ of the new graph as in Section 5.2.
3. Let $\hat{k}'(k)$ be the mapping between the old and new degree values induced by $1K$-rescaling. Specifically, let $M =$
(k_l, k_h) be as in Section 5.2. If $k \notin M$, then $\hat{k}'(k) = k'(k)$ from steps 4 and 5 in Section 5.2. Otherwise, if $k \in M$, $\hat{k}'(k)$ is given by the degree mapping from step 6c in Section 5.2.
4. Let $X = |\text{supp}(P)|_M$ and $X' = |\text{supp}(P')|_{M'}$ be the sizes of supports of $P(k)$ and $P'(k')$ within the M and M' intervals respectively. When scaling up, $n' > n$,

$X' \geqslant X$, and the $2K$-distribution is computed as follows:

$$P'(\hat{k}'_1(k_1), \hat{k}'_2(k_2)) = \begin{cases} P(k_1, k_2), & \text{if } k_1 \notin M \\ & \text{and } k_2 \notin M, \\ \left(\frac{X}{X'}\right)^2 P(k_1, k_2), & \text{if } k_1 \in M \\ & \text{and } k_2 \in M, \\ \left(\frac{X}{X'}\right) P(k_1, k_2), & \text{otherwise.} \end{cases}$$

When scaling down, $n' < n$, $X' \leqslant X$, and

$$P'(k'_1, k'_2) = \sum_{k_{1,2} | \hat{k}'_{1,2}(k_{1,2}) = k'_{1,2}} P(k_1, k_2)$$

That is, the JDD shapes before and after rescaling are the same, while the above expressions guarantee that the JDDs of both original and new graphs are properly normalized.

As in the $1K$ case, we then supply the produced $2K$-distribution $P'(k'_1, k'_2)$ to $2K$-random graph construction algorithms in the previous chapter to obtain the final graph.

5.3.1 $1K + r$-rescaling

We have also experimented with a rescaling technique lying somewhat in-between $1K$- and $2K$-rescaling. The motivation for it is that for the AS-level graphs, the assortativity coefficient r, a summary statistic of the $2K$-distribution, has remained roughly constant over time (see Table 5.1). We can thus perform $1K$-rescaling and then move the resulting graph to a target value of r by a sequence of $1K$-preserving r-targeting rewirings. At each rewiring step, we select a random pair of edges (v_1, v_2) and (v_3, v_4) and rewire them to the pair (v_1, v_4) and (v_2, v_3) only if the $1K$-distribu-tion does not change and the value of r after rewiring is closer to its target value than before (see [56] for further details).

To generate 2K random graphs of any size, we first compute the assortativity coefficient r of the given graph. We then generate a $1K$-random graph of the desired size using the $1K$-rescaling algorithm described in section 5.2. Next, we perform $1K$-preserving rewiring on this graph in such a manner that the rewirings take the graph towards the target r value. Specifically, we choose a random pair of edges in the generated $1K$-graph and rewire those two edges only if the r value of the resulting graph is closer to the target r-value. Note that it is not possible to perform rewiring targeting the exact $2K$-distribution because it is not mathematically meaningful to evaluate whether a rewiring moves us closer to a target matrix. We keep iterating on this rewiring procedure until we reach the target r-value or when a threshold number of candidate random rewirings do not move us closer to the target.

One potential problem with this rewiring technique is that it may be non-ergodic, meaning that there may not be a continuous sequence of rewirings that will take the graph from its initial state to a final state that achieves the target r value. While we did not encounter this limitation in the graphs we considered, techniques such as simulated annealing could mitigate potential problems with this technique if necessary. A second issue is that assortativity is only a summary of our target $2K$-distribution; seen another way, no scalar metric can capture all of the information in arbitrary $2K$-distributions represented by matrices. Hence this second technique may give up some accuracy.

We present a comparison between these two techniques for $2K$-rescaling in Section 4.4. At a high level, the first techniques of shrinking or expanding the original $2K$-distribution matrix has the potential to be more accurate. However, our technique for, essentially, filling in missing information may introduce inaccuracies of its own. The rewiring technique, on the other hand is simpler, and can be extended to higher-order distributions as long as we can find a scalar summary of a given dK-distribution (for instance, the mean clustering and correlation of degrees of nodes located at distance 2 from each other are such summaries for the $3K$-distribution). However, increasing amounts of information are lost as we complex more and more data onto a single scalar. Further, the rewiring is not guaranteed to converge to a target value as discussed earlier.

5.4 Results

In this section, we conduct a number of experiments to demonstrate the ability of our rescaling algorithms to reproduce important graph metric values. We compare our generated topologies to the original observed topology with respect to some of the most popular metrics used in the networking literature.

To report our metric values, we use notations described in Table 3.2 in the previous chapter. The results below represent averages over 10 generated graphs in each case. We note that the standard deviation and variance for all the metrics is negligible.

Table 5.2: Scalar metrics for $1K$-rescaled skitter AS graphs (Section 5.2

	Number of nodes					
Metric	Original (9200)	9200	6000	12000	15000	30000
k	6.29	6.34	6.17	6.38	6.35	6.44
r	-0.24	-0.24	-0.23	-0.23	-0.22	-0.21
$\bar{C}$	0.46	0.25	0.23	0.25	0.27	0.27
$\bar{d}$	3.12	3.11	3.18	3.15	3.13	3.1
σ_d	0.37	0.4	0.44	0.42	0.41	0.39
λ_1	0.1	0.03	0.1	0.09	0.09	0.08
λ_{n-1}	1.9	1.97	1.89	1.9	1.9	1.93

5.4.1 $1K$-rescaling for AS-graphs

The skitter AS-level topology for March 2004 has 9200 ASes, with an average degree of 6.29 and an assortativity coefficient of -0.24. As seen in the previous chapter, a $1K$-random graph with 9200 nodes reproduces most metric values of the original skitter topology except for clustering. We generate random graphs of varying sizes using the $1K$-rescaling algorithm and summarize our results for some of the metrics listed above in Table 5.2.

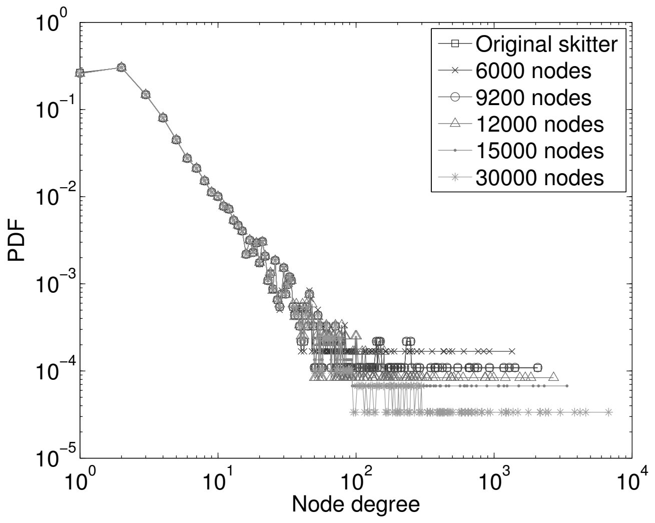


Figure 5.7: Degree distribution for $1K$-rescaled skitter AS graphs

In Figure 5.7 we plot the degree distribution of these different sized graphs. We find that the metric values are invariant for most graph sizes, and they closely match the corresponding values of the input skitter topology. The average degree, average distance and the assortativity coefficient remain constant even as the graph grows or shrinks in size. The distance distribution across different sized skitter graphs is the same as that of the original skitter graph, hence we do not show the plot for brevity. We generated larger size graphs containing up to 80,000 nodes and note that the average degree and assortativity coefficient values are maintained for these graphs as well.

5.4.2 $1K$-rescaling for router-graphs

Next, we experimented with a synthetic router-level topology –the HOT graph from [54]. As shown in the previous chapter, $2K$-random graphs reproduce metric values of the original graph much better than their $1K$-random counterparts. The original HOT graph has 939 nodes with an assortativity coefficient of -0.22 (making it disassortative). We generate different sized graphs for the HOT topology using our $1K$-rescaling technique for router-graphs described in Section 5.2. We report metric values for different sized HOT graphs in Table 5.3 and Figure 5.8 and Figure 5.9. Unlike AS-level

Table 5.3: Scalar metrics for $1K$-rescaled HOT graphs (Section 5.2

	Number of nodes					
Metric	Original (939)	939	250	2000	5000	8000
$\bar{k}$	2.1	2.59	2.09	2.25	2.28	2.22
r	-0.22	-0.14	-0.32	-0.12	-0.1	-0.1
$\bar{C}$	0	0.009	0.002	0.002	0.003	0.001
$\bar{d}$	6.81	4.41	5.4	5.02	4.94	5.4
σ_d	0.57	0.72	1.09	0.91	0.86	1.01
λ_1	0.004	0.034	0.006	0.013	0.015	0.007
λ_{n-1}	1.997	1.967	1.994	1.987	1.985	1.994

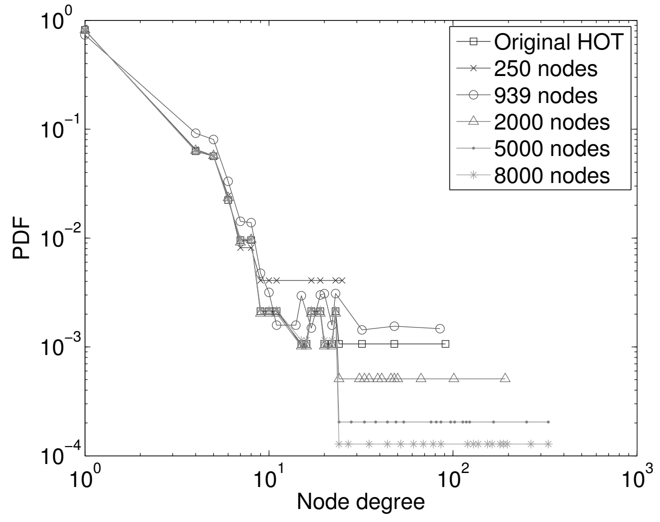


Figure 5.8: Degree distribution for $1K$-rescaled HOT graphs

topologies, we notice that the $1K$-random graphs do not accurately reproduce most metric values such as assortativity coefficient and average distance. The only exception is the assortativity coefficient for the $1K$-random HOT graph consisting of 250 nodes. The 250 node graph is almost 4 times smaller than the original graph and the proportionate scaling of the maximum node degree causes the change in the assortativity coefficient value. These results once again verify that reproducing HOT's $1K$-distribution is insufficient for reproducing important graph properties.

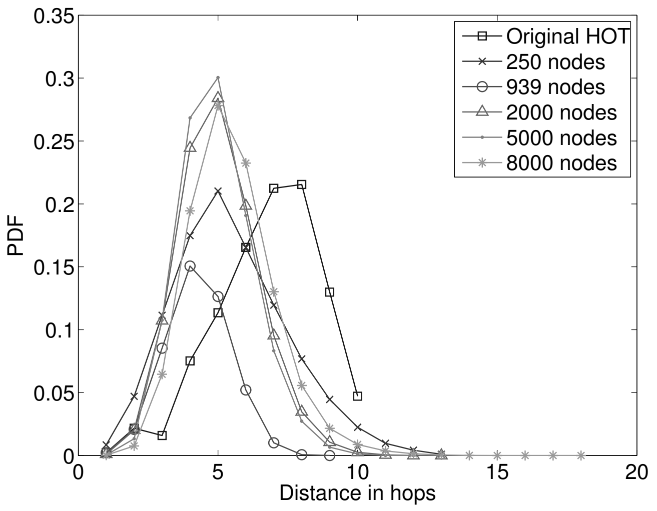


Figure 5.9: Distance distribution for HOT using $1K$-rescaling

5.4.3 $2K$ rescaling for AS-graphs

We generate different-sized skitter graphs using our $2K$-rescaling algorithm (Section 5.3) and present metric values for these graphs in Table 5.4. As in the $1K$ case, we notice that for variable-sized $2K$-random graphs, all metric values accurately mimic that of the original skitter graph, except for clustering. Earlier work [56] shows that clustering can be reproduced by a $3K$-generator. We have not yet implemented $3K$-rescaling, though we note that employing rewiring toward a target clustering value should result in appropriate clustering values for the rescaled graphs. For brevity, we do not plot the degree distribution of these graphs as they match their $1K$-counterparts. Finally, Figure 5.10 plots the distance distribution of these graphs. We see that for all the graphs, the distance distribution remains almost unchanged, matching the trends in the input graph data for a variety of graph sizes.

5.4.4 $1K + r$-rescaling for AS-graphs

Next, we generate the graphs using our $1K$-rescaling algorithm and then subject them to the $1K$-preserving r-targeting rewiring process. The process terminates upon reaching required value of r. We present the metric values for the skitter graphs

Table 5.4: Scalar metrics for $2K$-rescaled skitter AS topologies

	Number of nodes					
Metric	Original (9200)	9200	6000	12000	15000	30000
$\bar{k}$	6.29	6.29	6.09	6.31	6.35	6.44
r	-0.24	-0.24	-0.22	-0.23	-0.23	-0.22
$\bar{C}$	0.46	0.29	0.26	0.28	0.28	0.27
$\bar{d}$	3.12	3.08	3.1	3.1	3.1	3.12
σ_d	0.37	0.35	0.38	0.36	0.35	0.35
λ_1	0.1	0.15	0.13	0.13	0.15	0.12
λ_{n-1}	1.9	1.85	1.87	1.88	1.88	1.91

Table 5.5: Scalar metrics for skitter using target r rewiring

	Number of nodes				
Metric	Original (9200)	6000	12000	15000	30000
$\bar{k}$	6.29	6.17	6.38	6.35	6.44
r	-0.24	-0.24	-0.24	-0.24	-0.24
$\bar{C}$	0.46	0.24	0.26	0.25	0.25
$\bar{d}$	3.12	3.1	3.09	3.09	3.2
σ_d	0.37	0.24	0.36	0.33	0.34
λ_1	0.1	0.13	0.13	0.15	0.12
λ_{n-1}	1.9	1.88	1.88	1.89	1.91

in Table 5.5. Since all the generated $1K$-random skitter graphs have values for r close to the required r value of the original graph, a few rewirings are sufficient for all the graphs to reach their target state. In the case of skitter, the loss of accuracy from using the r value instead of the entire JDD matrix appears minimal. In fact, for all metrics we considered, the r-targeting rewiring performs as well as the $2K$-rescaling technique. The reason behind this effect is that the skitter AS topology is almost $1K$-random [56], *i.e.*, it can be accurately captured using only its $1K$-distribution.

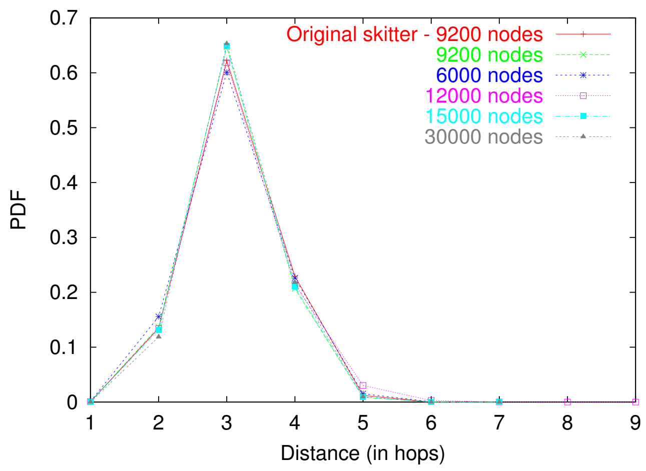


Figure 5.10: Distance distribution for skitter topologies using $2K$-rescaling

5.4.5 $2K$-rescaling for router-graphs

We now generate different sized $2K$-random HOT graphs using our $2K$-rescaling technique. We present the metric values for these graphs in Table 5.6 and plot the distance distribution in Figure 5.11 and the normalized betweenness distribution in Figure 5.12. Betweenness is a hard metric to reproduce for the HOT graph, but all the 2K random-graphs reproduce the shape of the betweenness curve exactly. The difference in the betweenness values for these graphs is due to the difference in their sizes. The $2K$-random hot graphs better reproduce the metric values of the original HOT graph than the $1K$-random HOT graphs, even when scaling up by a factor of 10 or scaling down by a factor of 4. We observe that while the average degree remains constant for all sized HOT graphs, the assortativity coefficient does change. This behavior results from the fact that we have not allowed the higher degrees in the graph to scale in an exact linear manner and that we have imposed a limit of 330 on the maximum degree in the graph (to reflect technological constraints, see Section 5.1.1).

5.4.6 $1K + r$-rescaling for router-graphs

Next, we present results for our $1K + r$ rescaling of router graphs. Unlike $1K$-random AS graphs, for the $1K$-random HOT graphs, reaching the target r values

Table 5.6: Scalar metrics for HOT graphs using $2K$-rescaling

	Number of nodes					
Metric	Original (939)	939	250	2000	5000	8000
$\bar{k}$	2.1	2.18	2.2	2.19	2.2	2.21
r	-0.22	-0.23	-0.36	-0.19	-0.18	-0.18
$\bar{C}$	0	0.0001	0.0005	0.0004	0.0005	0.0001
$\bar{d}$	6.81	6.32	5.4	6.4	6.6	6.92
σ_d	0.57	0.71	0.84	0.83	0.97	1.02
λ_1	0.004	0.005	0.01	0.004	0.004	0.005
λ_{n-1}	1.997	1.996	1.986	1.997	1.996	1.996

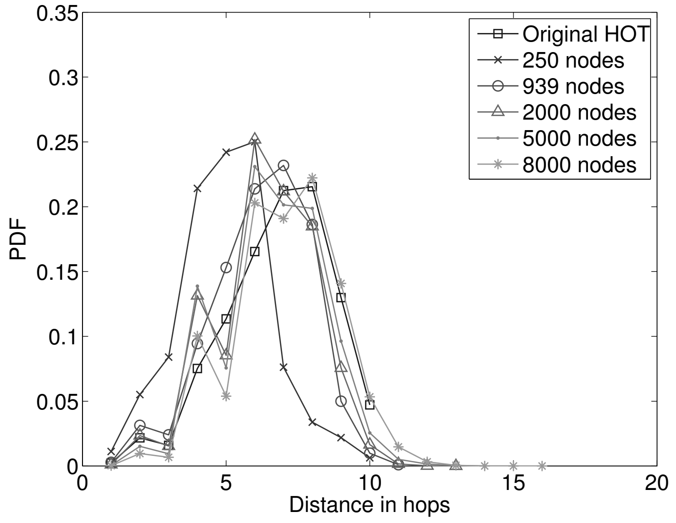


Figure 5.11: Distance distribution for $2K$-rescaled HOT topologies

takes longer as the r values of $1K$-random HOT graphs are not close to the target value. We report some of the scalar metric values for the HOT graph in Table 5.7. While most metric values are similar to the HOT graphs from the $2K$-rescaling technique, the average distance value is higher for all the graphs obtained using the r-targeting technique. This increase in value for the average distance and, correspondingly, in the distance distribution results from employing the scalar summary of the $2K$-distribution instead of the entire function. Since the HOT graph is not $1K$-random but almost $2K$-random [56], the full information contained in its $2K$-distribution is required to accurately capture all

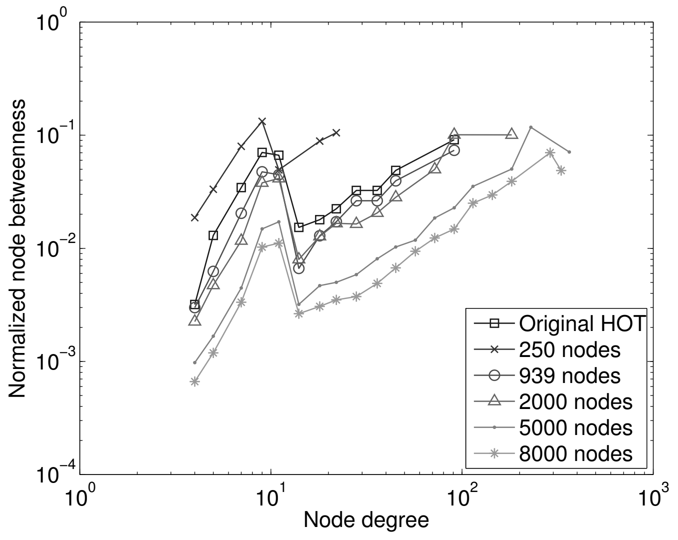


Figure 5.12: Betweenness distribution $2K$-rescaled HOT topologies

Table 5.7: Scalar metrics for $1K + r$-rescaled HOT graphs (Section 5.3.1)

	Number of nodes				
Metric	Original (939)	250	2000	5000	8000
k	2.1	2.09	2.26	2.28	2.22
r	-0.22	-0.22	-0.2	-0.19	-0.18
$\bar{C}$	0	0.005	0.0006	0.0003	0.0005
$\bar{d}$	6.81	5.4	7.3	7.41	7.8
σ_d	0.57	0.89	0.82	.73	0.74
λ_1	0.004	0.01	0.001	0.002	0.003
λ_{n-1}	1.997	1.989	1.998	1.998	1.998

its global properties. Of course, its $1K + r$-random version must closer to the original than the $1K$-random version, but farther than the $2K$-random one.

5.5 Summary

Overall, we are satisfied with our ability to generate graphs of a range of sizes while maintaining important graph properties. Importantly, our analysis of historical AS topologies and router topologies of a range of sizes shows that a range of local graph properties often remain invariant even as the graph as a whole changes size. Our algorithms are motivated by this observation. We have shown how to scale a graph up or down (*e.g.*, number of nodes, maximum degree) while still maintaining other important graph properties.

As we have described, available data sources have a number of limitations and inaccuracies, which means that our generated topologies will necessarily reflect any inaccuracies in the underlying data. The primary contribution of this work is not the summary data we collected about Internet characteristics, but rather a methodology for topology generation given an available data source.

Given our heuristics for rescaling, one interesting question is the maximum rescaling degree that we can safely apply to a given graph. We believe that given the data sources available to us and our analysis of this data, scaling by a factor of approximately 10 and perhaps 100 is meaningful using our methodology. However, it would not, for instance, be meaningful to scale a ten node topology to one with one million nodes. Such scaling would likely require a fundamental understanding of the laws governing the evolution of a given graph [17] rather than the observation-based techniques we employ to see how graphs evolve historically based on mathematical topology metrics. However, given that we do not yet have a firm grasp of such evolutionary laws, our rescaling techniques and our generator that combines both AS and router-level information presents an intermediate solution for researchers interested in evaluating the performance of services and protocols for a range of graph sizes.

5.6 Acknowledgments

Chapter 5, in full, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Conference, Kyoto, Japan, August 2007, Mahadevan, Priya; Hubble, Calvin; Krioukov, Dmitri; Huffaker, Bradley; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Annotating Topologies

In the previous chapters, we have seen how our dK-series forms the basis for all graph synthesis and analysis. The dK-series unifies all graph metrics, both current and as well as ones that maybe proposed in the future. Using the associated dK-distributions as input, we have been able to successfully generate dK-random graphs that capture increasingly fine-grained metrics of Internet topologies. We have also put forth algorithms to effectively generate dK-random graphs of a wide variety of sizes, while still maintaining the structure and important metric values of the original measured topology. We now move to the final requirement of a topology generator as stated in Chapter 1 – the ability to annotate the generated graphs.

A graph that merely reproduces the structure of the Internet's AS-level or router-level topology, though still useful, might not adequately help researchers who would like to evaluate the performance of their network applications and services. One reason is that influence of node and link annotations such as latency, loss-rate, AS membership, *etc.*, can significantly impact application performance and protocol behavior. Several studies have emphasized the effect of link latencies on end-to-end application and protocol behavior and the need to accurately model observed latencies [104, 20, 71, 107]. Similarly, nodes in the generated topology should be annotated with AS membership information to enable studies that account for routing behavior. Business relationships among peering ASes (peering, customer, *etc.*) should also be included.

Unfortunately, router-level topologies produced by generators today either do not contain any AS membership information or are not reflective of real measurements. In the absence of AS membership information in the generated router graphs, most studies end up computing shortest paths between pairs of end nodes. Researchers have shown that actual packet paths between Internet hosts are substantially longer than the corresponding shortest paths [34, 91, 95]. Thus, studies that use shortest path routing between end hosts will likely yield incorrect results for packet round-trip times and thus negatively impact application performance. Annotating each router with AS membership will enable us to implement correct routing paths in the topology. As an example, one can employ modified shortest-path algorithms such as [57] that respect AS membership and routing policies between the ASes.

Using AS-level topologies instead for these studies has its own set of limitations. AS-topology generators such as Inet [103] do not provide details of the topology within an AS. They assume that each AS can be effectively modeled by one router. However, an AS comprises of fairly complex interconnectivity amongst varying number of routers. Muhlbauer *et al.* [65] show that the internal structure of an AS is critical as it influences inter-domain routing such as hot-potato routing [79, 80]. Representing an AS as a single node results in an AS always choosing a single best-path to its neighbors and thus cannot capture diverse behavior of BGP such as hot-potato routing observed in reality. Thus, to capture the route diversity, it is critical to model the router-level topology within an AS.

In this chapter, we focus on techniques for annotating our generated router graphs. Annotations in a router-level topology can include latencies and bandwidths for edges in the graph and AS-memberships for routers in the graph. We specifically describe our method to generate router-level graphs annotated with AS-membership as well as whether a router is internal or peering. We note that our methodology for annotating the routers are generic and can be applied to other types of topologies such as AS-topologies. Our algorithms can be adapted to annotate the nodes (*i.e.* ASes) with tiers (tier1, tier2, *etc.*) and the edges can be annotated with the type of relationship between the two ASes such as peer-to-peer or customer-provider.

6.1 AS-membership Annotations

Since our goal is to annotate router-level topologies with AS-membership, we choose the skitter data, which is one of the best available measurements today. The skitter traceroute data described in the previous chapter (Section 5.1.1) includes information on both ASes and router connectivity. Given AS annotation information in an original router topology, there are two possible techniques for maintaining AS annotations in the randomly generated rescaled graph. The first, bottom up technique, would simply rescale the input router topology and then devise techniques to “grow” contiguous ASes to match some target number of ASes, making some assumptions about how the number of ASes scales with the total number of routers and using observations of the number of routers per AS in the original topology.

Unfortunately, we could not devise any straightforward techniques for filling in the details of this bottom up technique. Thus, we propose a top down technique for generating a rescaled, annotated topology. This technique consists of the following high-level phases illustrated in Figure 6.1:

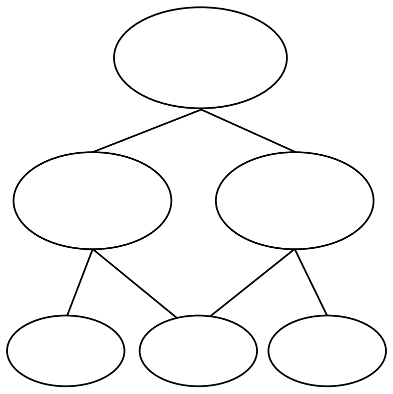
1. Generate AS-level topology of desired size.

2. Populate each AS with a router-level topology using information on correlations between AS degrees and sizes measured by the number of routers within an AS.

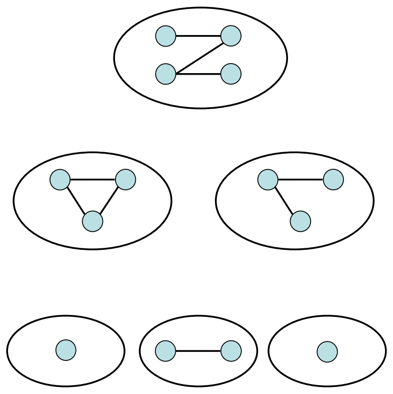
3. Select peering (*i.e.*, inter-AS or border) routers for each AS based on the peering router statistics extracted from the traceroute data.

4. Glue per-AS router topologies into a global router topology by connecting peering routers.

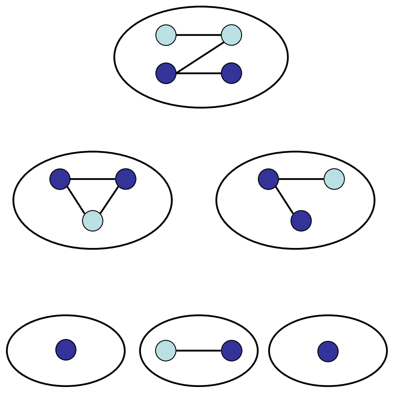
While we describe details of our methodology in the context of skitter below,



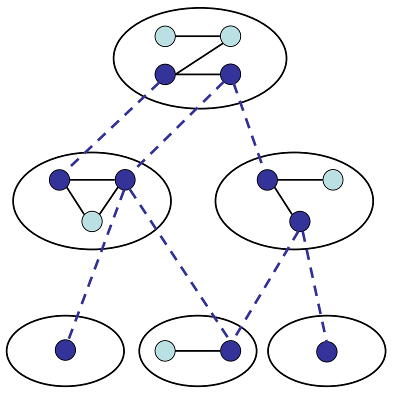
(a) Generating AS-graph



(b) Generating router graphs for each AS



(c) Assigning peering routers for each AS



(d) Connecting peering routers

Figure 6.1: Generating a router-level topology, where each router is annotated with AS-membership

we note that our approach is general to a variety of data sources. For example, we could use Rocketfuel [92] or iPlane [55] data to generate router-level topologies for each AS.

To provide more details about our annotation techniques, we first describe some additional post-processing of the skitter data from Section 5.1.1. In addition to size-based AS categories $T_1, \ldots, T_4$ from Section 5.1.1, we also use logarithmic binning to coarsely smooth the AS degree distribution and split all ASes into the following degree-based classes C_1, C_2, C_3:

AS class	AS degree K
C_1	$K < 10$
C_2	$10 \leqslant K < 100$
C_3	$K \geqslant 100$

Note that after the data processing in Section 5.1.1, we do not have ASes with $K > 1000$.

We next statistically relate AS classes $c = C_1, C_2, C_3$ and categories $t = T_1, \ldots, T_4$. Let $A(c,t)$ be the set of ASes of class c and category t. For each combination (c,t), we keep the following statistics:

- The number of ASes $N(c,t)$ in $A(c,t)$.
- The collection $J_a(c,t)$, $a = 1, \ldots, N(c,t)$, of the intra-AS router topology JDDs of all ASes in $A(c,t)$. Each $J_a(c,t)$ denotes the whole $2K$-distribution $P(k_1,k_2)$ of the router topology of AS $a \in A(c,t)$.
- The corresponding collection $D_a(c,t)$ of the peering router degree distributions. For each AS a, $D_a(c,t) = P_B(k)$, where $P_B(k)$ is proportional to the number of routers within AS a that have degree k and that peer with routers in other ASes.

Having prepared the statistics above, we generate rescaled AS-annotated router graphs as follows:

1. Rescale the AS-level graph using $2K$-rescaling as described in Section 5.3 (Figure 6.1(a)).
2. For each AS A in the rescaled AS graph:

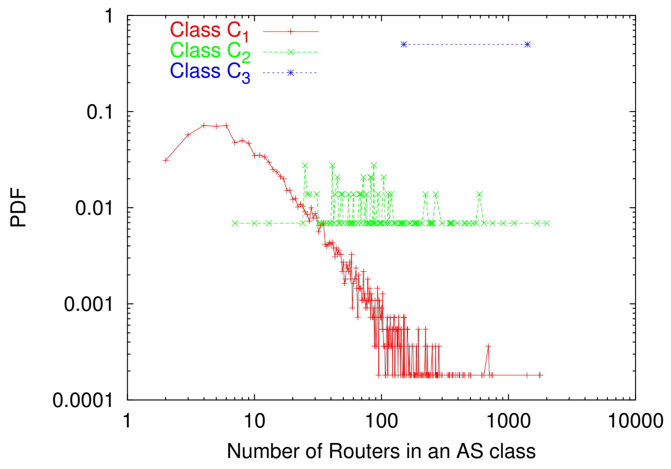


Figure 6.2: PDF of number of routers in an AS class

(a) Determine A's class c.

(b) Randomly select A's category t with a conditional probability proportional to $N(c, t)$.

(c) Select an AS a from $A(c, t)$ uniformly at random.

(d) Populate A with a $2K$-random router topology based on $P(k_1, k_2) = J_a(c, t)$ (Figure 6.1(b)).

(e) Choose peering routers within AS a having degree k based on a probability proportional to $P_B(k) = D_a(c, t)$ (Figure 6.1(c)).

3. Walk over the pairs of adjacent ASes in the rescaled AS graph and connect random pairs of designated peering routers in each AS.

6.2 Results

To report our metric values, we use notations described in Table 4.2. We now evaluate our techniques for performing AS-membership annotations in our generated router topologies. Since this topology combines both router and AS information, in addition to evaluating metrics for the overall graph, we present results on the fraction of peering-routers within an AS, and degree distribution of routers within an AS.

Table 6.1: Scalar metric values for router-level topologies annotated with AS membership

	$\bar{k}$	r	k_{max}	$\bar{d}$
Original	4.25	0.006	1141	9.14
Generated	3.9	-0.0009	1333	8.53

We classify the ASes in the AS-level topology based on its degrees (Section 6.1). Our AS-level topology consists of a total of 5662 ASes. Of all the ASes, 97% belong to class 0, 2.6% belong to class 1 and the remaining belong to class 2. Figure 6.2 plots the PDF for the number of routers in an AS for every AS class. As expected, we observe that a majority of the ASes belong to class 0 have fewer than 10 routers in their router-level topology. The number of routers belonging to the class 0 ASes appear more widespread than class 1 or class 2 routers. In fact, the maximum number of routers observed for a class 0 AS is 1774. The number of routers for a class 1 AS ranges from 10–1996. For class 2 ASes, the minimum number of routers is 150. This agrees well with the intuition that ASes that have a higher peering degree tend to have more routers.

Next, we present some scalar metric values to compare generated combined AS-router-level topology with a similar graph extracted from the skitter traces in Table 6.2. These metrics are for the overall router-level topology across all of the ASes for both the original as well as the generated graph.

Since both graphs have approximately 200,000 nodes, computing clustering, distance and betweenness distributions will require extremely long running times. Hence, we randomly choose 50,000 unique paths from our original and generated graphs and compute the average distance for these sampled paths using shortest-path algorithm. We note that average distance for the generated graph compares well with that of the original graph. In Figure 6.3, we show the PDF of distance distribution for the original and generated skitter-router topology.

Next, we plot the degree distribution of all routers within an AS for each AS class for both the original and generated graphs in Figure 6.4, Figure 6.5, and Figure

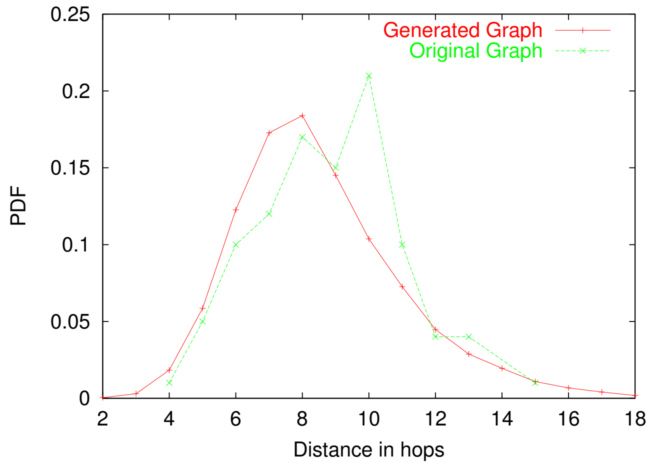


Figure 6.3: Distance distribution in skitter-router topology for original and generated graphs

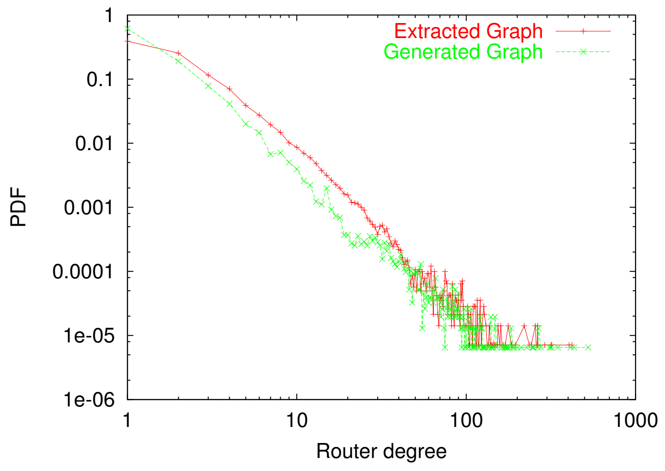


Figure 6.4: Degree distribution for routers belonging to AS class 0

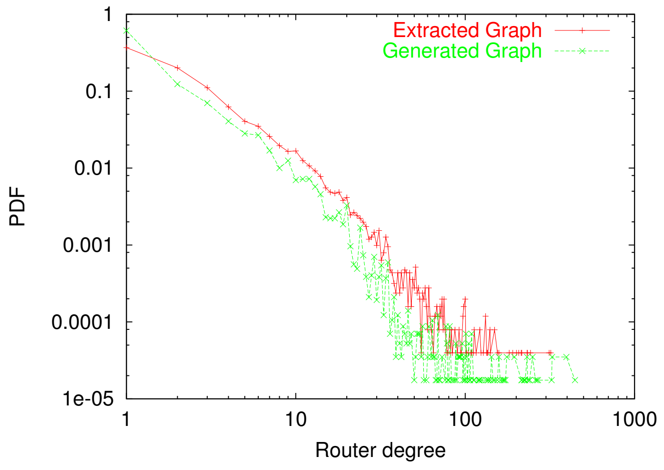


Figure 6.5: Degree distribution for routers belonging to AS class 1

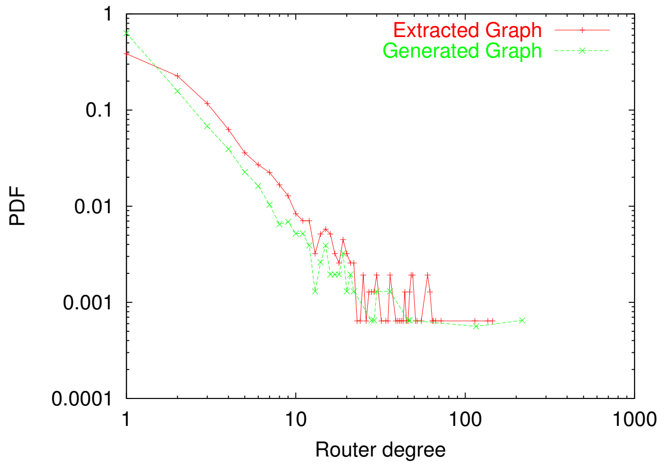


Figure 6.6: Degree distribution for routers belonging to AS class 2

6.6. This plot allows us to compare how close the extracted and generated topologies are with respect to the degree distribution of all the routers belonging to each AS class. We observe that the variations in the degree distributions stem from the limitations in our data set.

6.3 Summary

In this chapter we describe techniques to annotate nodes in the generated router-level graphs with AS-membership as well as router role (peering or internal). Other useful annotations include link latencies and bandwidths for the edges in the graph. While this dissertation restricts its attention to AS annotations for generated router topologies, we believe our techniques can be generalized and used support annotations for link latency and capacity based on available measurement sources.

6.4 Acknowledgments

Chapter 6, in full, is a reprint of the material as it appears in the Proceedings of the ACM SIGCOMM Conference, Kyoto, Japan, August 2007, Mahadevan, Priya; Hubble, Calvin; Krioukov, Dmitri; Huffaker, Bradley; Vahdat, Amin. The dissertation author was the primary investigator and author of this paper.

Chapter 7

Conclusions and Future Work

Topology generators that can accurately reproduce characteristics of observed networks play a very important role in a range of networking studies such as evaluation of network protocols and applications, network management, traffic engineering, *etc.* To date, researchers have not been able to successfully generate random graphs that reproduce all metrics of real measured topologies.

Our work is the first to propose and implement algorithms capable of generating random graphs that can reproduce all graph metrics including any that maybe defined in the future. We also present methods to generate graphs of sizes different from the original topology and outline techniques to annotate these generated graphs. Specifically, we show how to annotate router-level topologies with router role (peering or internal) as well as AS-membership. In this final chapter, we summarize our contributions. Further, we discuss scenarios that could benefit from our research. We close this dissertation with a discussion on avenues for future research.

7.1 Contributions

Researchers have proposed a wide variety of graph metrics for comparing and analyzing network topologies. Values for a particular graph metric may capture a graph's resilience to failure or its routing efficiency. Knowledge of appropriate metric values may influence the engineering of future topologies, repair strategies in the face of failure and understanding of fundamental properties of existing networks. Unfortunately, there are typically no algorithms to generate graphs matching one or more proposed metrics and there is little understanding of the relationships among individual metrics or their applicability to different settings. Further, existing techniques typically either cannot perform graph rescaling or do not do so in a manner that reflects known patterns of network evolution.

To this end, our dissertation makes the following contributions.

1. We present a finite set of unifying graph properties, the dK-series, to describe and constrain random graphs in successively finer detail. In our model, we make use of probability distributions on the subgraphs of size d in some given input graph. More formally, the dK-distribution describe the correlations of degrees of d connected nodes. We call *dK-graphs* the sets of graphs constrained by such distributions. In the limit, these generated dK-graphs are isomorphic to a given graph and thus will necessarily reproduce *any* graph metric of the original graph, by virtue of this isomorphism.

2. We develop and implement new algorithms for constructing $2K$- and $3K$-graphs (algorithms to generate $0K$- and $1K$-graphs are already known). Producing a family of $0K$-graphs for a given input graph requires reproducing only the *average* node degree of the original graph, while producing a family of $1K$-graphs requires reproducing the original graph's node degree *distribution*. $2K$-graphs reproduce the *joint* degree distribution of the original graph as well—the probability that a node of degree k is connected to a node of degree k'. $3K$-graphs consider triples of nodes, and so forth.

3. We explore interesting tradeoffs in choosing the appropriate value of d to compare two graphs. As we increase d, the set of randomly generated graphs having the

corresponding dK-distribution becomes increasingly constrained and the resulting graphs are increasingly likely to reproduce a variety of metrics of interest. At the same time, the algorithmic complexity associated with generating the graphs increases sharply. Thus, we present a methodology where practitioners choose the smallest d that captures essential graph characteristics for their study. For the comparatively complex Internet AS- and router-level topologies that we consider, $d = 2$ is sufficient for most cases and $d = 3$ captures all graph properties proposed in the literature known to us.

4. To explore the structural diversity amongst all dK-graphs, we present *space exploration* techniques that allow us to generate graphs with arbitrary values for metrics that are not defined by the corresponding dK-distribution.

5. We present algorithms to rescale the appropriate dK-distributions in order to generate graphs of sizes vastly different from the original graph. We examine historical Internet connectivity data to determine our rescaling strategies. We experimentally show that we are able to produce $1K$- and $2K$-graphs of a variety of sizes while still maintaining other important graph characteristics for AS- and router-level topologies.

6. We present a top-down technique for generating router-level topologies annotated with AS membership. We further classify routers in this topology based on whether they are peering or internal routers. We compare our randomly generated, annotated router topologies to observed Internet router topologies and find close matches for a range of graph metrics proposed in the literature.

7. Finally, all of our graph generation techniques are not dependent on the topology type or its evolution. Thus, all of our algorithms are applicable to other topologies such as enterprise and wireless networks, biological networks, social networks, *etc.*

Our topology generator will serve as valuable input to a range of studies in the networking community. We outline several specific cases here.

Evaluation of routing protocols: Studying routing protocol scalability and convergence requires knowledge of both topology and AS relationships. Convergence of BGP has been shown to depend on the topological characteristics of the network [52, 1]. Our topology generator can be used in the evaluation of scalability of such routing protocols.

Evaluation of network protocols and applications: Many studies on network protocols especially congestion control employ simple "dumbbell"-style topologies. Such studies will benefit from considering more realistic graphs that reflect the structure of actual measured topologies. We hope that our topology generator suite will enable protocol and application researchers to test system behavior under a suite of randomly generated yet appropriately constrained and realistic network topologies. For example, they can pose and answer questions such as how does the estimation of round-trip times vary with variance in topological features, how does the sending rate adapt with the detection of bottlenecks in the network, *etc.*

Network coordinate systems: Developing network coordinates [20, 71, 87] and geo-localization [105, 14] has recently gained prominence in the networking community. Network coordinate systems involve assigning synthetic coordinates to Internet hosts such that the Euclidean distance between two hosts coordinates predicts the network latency between them. The goal of geo-localization systems is to determine the physical location of Internet hosts. These systems play a key role in several scenarios such as network management, fault diagnosis, *etc.* Our topology generator can supply a range of input Internet graphs and potential deployment scenarios in support of these systems.

GENI: One important research initiative is GENI (Global Environment for Network Innovation) [35], whose goal is to lower the barriers for validation and deployment of software, and to focus on innovative designs for the Future Internet. Our generator will serve as a valuable input for infrastructures that will evaluate future networking applications and protocols prior to their deployment. One expected result of GENI is the availability of an annotated map of the entire Internet including detailed router and link attributes. We envision our topology generator suite as a component of the GENI project. While network infrastructures may change over time, leading to different topological features of the Internet than the ones observed today, our generator will be able to generate these future Internet graphs as well.

Traffic Engineering: Knowledge of network topology is critical for ISPs for traffic engineering. Traffic engineering provides the ability to move traffic flows away from the shortest path and onto a potentially less congested physical path across the service provider's network. While this ability requires intimate knowledge of the topology, engineers can evaluate their traffic engineering solutions on a variety of graphs produced by our generator. If it is found that certain connectivity characteristics in the topology yield better alternate routes for forwarding the traffic, then the network can be engineered to follow the optimum connectivity pattern. In general, our generated graphs allow network engineers and administrators to play around with different 'what if' scenarios and effectively validate their solutions.

Network evolution: The dK-series is a descriptive model in the sense of matching all graph-theoretic properties of observed networks as compared to network evolutionary models, that aim to seek the fundamental forces driving the growth of the network. While the dK-series cannot be used to understand the evolutionary laws driving the network growth, it can simplify the task of validating evolutionary models. Consider the case where a researcher wishes to validate a model for Internet evolution using historical connectivity information. The process would likely involve starting with an initial graph, *e.g.*, reflecting connectivity from 5 years ago, and generating a variety of larger graphs, *e.g.*, reflecting modern-day connectivity. Of course, the resulting graphs will not match known modern connectivity exactly. Currently, validation would require showing that the graph matches "well enough" for all known ad hoc graph properties. Using the dK-series however, it is sufficient to demonstrate that the resulting graphs are dK-random for an appropriate value of d, i.e., constrained by the dK-distribution of modern Internet graphs (with $d = 3$ known to be sufficient in this case). As long as the resulting graphs fall in the dK-random space, the theory of dK-randomness explains any variation from ground truth. This methodology also addresses the issue of defining "well enough" above: dK-space exploration can quantify the expected variation in ad hoc properties not fully specified by a particular dK-distribution.

Other complex networks: Complex networks are studied across many fields such as biology, physics, mathematics, sociology, *etc.* A network topology graph is the simplest abstraction of any complex system consisting of many interacting elements. Typically, elements of the system are abstracted as nodes and their interactions as links. Examples of complex networks include protein-protein interaction networks, social networks, *etc.*

Protein-protein interaction networks, which are commonly used by biologists, depict interactions among all the proteins of an organism. Biologists have shown that proteins that interact are more likely to co-evolve [23, 29, 59, 75], therefore it is possible to make inferences about interactions between pairs of proteins based on their phylogenetic distances. Thus, using a network to represent protein interactions and analyzing such a network has now become an accepted practice among biologists.

A social network is a set of people or groups of people with some pattern of contacts or interactions between them [67]. The patterns of friendships between individuals [64, 66] and business relationships between companies [66] are examples of social networks that have been studied in the community. Another well studied example is the actor network. In this network, actors are represented as nodes in a graph and two actors are considered connected if they have appeared in a film together [68]. Graphs are typically used to represent the above networks. Network modeling, analysis and random graph generation have become popular among sociologists studying such networks.

Since our graph generation algorithms do not depend on the type of the topology (such as AS- or router-level) and the evolution of the underlying network, all our graph analysis and synthesis techniques are applicable to other complex networks such as biological, economic and social networks. Our initial results with biological and social networks show that $d = 3$ is sufficient to reproduce metrics of these graphs as well. Thus our graph generation algorithms will benefit researchers in biology, sociology, physics, mathematics and economics where graph synthesis and analysis plays an important role.

7.2 Future Work

Our dissertation demonstrates that we can generate random graphs of a wide range of sizes that match observed topologies with respect to all metrics proposed or that may be proposed in the literature. Indeed, by reproducing local connectivity properties ($2K$- and $3K$-distributions), we have also been able to successfully reproduce global metrics such as distance distribution, spectrum and betweenness. Along the way, we have discovered a number of interesting avenues for future research.

- *Effective techniques for measuring Internet topologies and characteristics:* While there are several sources of data available for both AS-level and router-level topologies, researchers recognize that these are essentially incomplete. Incompleteness of the data as well as biases during measurement may distort our view of the Internet [50]. Indeed, our ability to reproduce observed input graph properties is limited by the quality of available Internet measurements. However, the methodology and algorithms for generating annotated, rescaled topologies constitute the primary contributions of this work, and not necessarily the particular generated graphs. While we can verify that our topologies match observed router topology characteristics, we cannot claim that either the input or generated topologies accurately reflect reality. The quality of our generated topologies will improve with the quality of available measurements.

  Thus, an important future direction is developing effective and efficient methodologies to measure the Internet's AS- and router-level topologies such that these measurements reflect the real network connectivity. Additionally, accurate measurements of Internet characteristics such as bandwidth, latency and loss-rate will be critical to any further developments in network modeling.

- *Enterprise networks:* One promising avenue is studying the characteristics of enterprise networks. Networks of large organizations, in general, are poorly understood. It is likely that the topological features of enterprise networks are different from that of the Internet's AS- or router-level topology. One interesting study is determining the smallest value of d that can accurately model enterprise topologies. Can these complex networks indeed be modeled using low values of d such as 2 and 3? Such a study would enable network administrators of large enterprises to better manage both the network and the traffic flowing through it.

  Similarly, one can also study the topological features of wireless networks, especially *ad hoc* wireless networks. Verifying that $d = 3$ is sufficient to reproduce important characteristics of wireless networks will allow us to generate realistic wireless network models. Such generated graphs will benefit researchers involved in designing wireless applications and protocols.

- *Why is $d = 3$ sufficient?* Another important question pertains to the speed of convergence of the dK-series for Internet topologies. Why is it possible for $3K$-random graphs to accurately reproduce all known metrics for even comparatively complex router-level topologies? An interesting area of further research is to theoretically connect $d = 3$ with decisions that guide the engineering of router-level topologies. Analysis along these lines will enable us to further understand the principles guiding the evolution of Internet topologies and thus bridge the gap between descriptive models such as ours and evolutionary models.

- *Annotations for bandwidth and loss-rate:* In this dissertation we described techniques to annotate nodes in router-level topologies with AS-membership as well as router roles (internal or peering). Our success in maintaining topology characteristics while supporting the above node annotations suggests that our ideas and methodologies can also be used to augment the edges in the generated graphs. Currently, we are working on techniques to annotate the links in router-level topologies with latencies and bandwidth values taken from various measurement studies. It remains to be seen whether such annotations can match the distributions found in real networks but we are encouraged by available data sets both for end-to-end and per-hop latency distributions.

We are releasing the source code for our topology generator suite to produce dK-random graphs of any specified size for $d = 1, 2$. Additionally, we are also releasing code to compute all important metrics for any given topology. We believe that our techniques will provide a more rigorous and consistent method of comparing graphs and enable protocol and application researchers to test system behavior with more realistic network topologies.

Bibliography

[1] A. Ahuja, C. Labovitz, S. Venkatachary, and R. Wattenhofer. The Impact of Internet Policy and Topology on Routing Convergence. In *Microsoft Technical Report - MSR-TR-2000-74*, July 2000.

[2] W. Aiello, F. Chung, and L. Lu. A Random Graph Model for Massive Graphs. In *STOC*, 2000.

[3] Akamai, Inc. http://www.akamai.com.

[4] D. Applegate and E. Cohe. Making Intra-Domain Routing Robust to Changing and Uncertain Traffic Demands. In *Proceedings of the ACM SIGCOMM Conference*, August 2003.

[5] A.-L. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286:509–512, 1999.

[6] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford. In VINI Veritas: Realistic and Controlled Network Experimentation. In *Proceedings of the ACM SIGCOMM Conference*, 2006.

[7] M. Boguñá and R. Pastor-Satorras. Class of Correlated Random Networks with Hidden Variables. *Physical Review E*, 68:036112, 2003.

[8] M. Boguñá, R. Pastor-Satorras, and A. Vespignani. Cut-offs and Finite Size Effects in Scale-Free Networks. *European Physical Journal B*, 38:205–209, 2004.

[9] A. Brady and L. Cowen. Compact Routing on Power-law Graphs with Additive Stretch. In *ALENEX*, 2006.

[10] U. Brandes. A Faster Algorithm for Betweenness Centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[11] Y. Breitbart, C.-Y. Chan, M. Garofalakis, R. Rastogi, and A. Silberschatz. Efficiently Monitoring Bandwidth and Latency in IP Networks. In *INFOCOM*, 2001.

[12] T. Bu and D. Towsley. On Distinguishing Between Internet Power Law Topology Generators. In *INFOCOM*, 2002.

[13] CAIDA. Macroscopic Topology AS Adjacencies. `http://www.caida.org/tools/measurement/skitter/as_adjacencies.xml`.

[14] M. Casado and M. J. Freedman. Peering Through the Shroud: The Effect of Edge Opacity on IP-based Client Identification. In *Proceedings of the USENIX Annual technical Conference (USENIX)*, 2007.

[15] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: High-bandwidth Multicast in a Cooperative Environment. In *19th ACM Symposium on Operating Systems Principles (SOSP'03)*, Oct. 2003.

[16] H. Chang, R. Govindan, S. Jamin, S. J. Shenker, and W. Willinger. Towards Capturing Representative AS-Level Internet Topologies. *Computer Networks Journal*, 44:737–755, April 2004.

[17] H. Chang, S. Jamin, and W. Willinger. To Peer or not to Peer: Modeling the Evolution of the Internet's AS-Level Topology. In *INFOCOM*, 2006.

[18] F. Chung and L. Lu. Connected Components in Random Graphs with Given Degree Sequences. *Annals of Combinatorics*, 6:125–145, 2002.

[19] F. K. R. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. American Mathematical Society, Providence, RI, 1997.

[20] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A Decentralized Network Coordinate System. In *Proceedings of the ACM SIGCOMM Conference*, 2004.

[21] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area Cooperative Storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP'01)*, October 2001.

[22] L. Dall'Asta, I. Alvarez-Hamelin, A. Barrat, A. Vázquez, and A. Vespignani. Exploring Networks with Traceroute-Like Probes: Theory and Simulations. *Theoretical Computer Science, Special Issue on Complex Networks*, 2005. `http://arxiv.org/abs/cs.NI/0412007`.

[23] T. Dandekar, B. Snel, M. Huynen, and P. Bork. Conservation of Gene Order: A Fingerprint of Proteins that Physically Interact. In *Trends Biochem Science, volume 23, pages 324-328*, 1998.

[24] DETER. http://www.isi.edu/deter.

[25] M. Doar. A Better Model for Generating Test Networks. In *GLOBECOM*, 1996.

[26] S. N. Dorogovtsev. Networks with Given Correlations. `http://arxiv.org/abs/cond-mat/0308336v1`.

[27] S. N. Dorogovtsev and J. F. F. Mendes. *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford University Press, Oxford, 2003.

[28] A. Elwalid, C. Jin, S. Low, and I. Widjaja. MATE: MPLS Adaptive Traffic Engineering. In *INFOCOM*, pages 1300–1309, 2001.

[29] A. Enright, I. Iliopoulos, N. Kyripides, and C. Ouzounis. Protein Interaction Maps for Complete Genomes Based on Gene Fusion Events. In *Nature, volume 402, pages 86-90*, 1999.

[30] P. Erdős and A. Rényi. On Random Graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[31] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On Power-law Relationships of the Internet Topology. In *Proceedings of the ACM SIGCOMM Conference*, 1999.

[32] S. Floyd and E. Kohler. Internet Research Needs Better Models. In *Proceedings of HotNets–I*, October 2002.

[33] M. J. Freedman, E. Freudenthal, and D. Mazires. Democratizing Content Publication with Coral. In *Proceedings of the 1st USENIX Symposium on Networked Systems Design and Implementation (NSDI '04)*, San Francisco, California, March 2004.

[34] L. Gao and F. Wang. The Extent of AS Path Inflation by Routing Policies. In *IEEE Global Internet Symposium*, 2002.

[35] GENI: Global Environment For Network Innovations. http://www.geni.net.

[36] C. Gkantsidis, M. Mihail, and A. Saberi. Conductance and Congestion in Power Law Graphs. In *SIGMETRICS*, 2003.

[37] C. Gkantsidis, M. Mihail, and E. Zegura. Spectral Analysis of Internet Topologies. In *INFOCOM*, 2003.

[38] C. Gkantsidis, M. Mihail, and E. Zegura. The Markov Simulation Method for Generating Connected Power Law Random Graphs. In *ALENEX*, 2003.

[39] R. Govindam and H. Tangmunarunkit. Heuristics for Internet Map Discovery. In *Proceedings 2000 IEEE INFOCON Conference*, March 2000.

[40] P. Harremoës. Binomial and Poisson Distributions as Maximum Entropy Distributions. *Transactions on Information Theory*, 47(5):2039–2041, 2001.

[41] J. Hawkinson and T. Bates. *Guidelines for Creation, Selection, and Registration of an Autonomous System (AS)*. IETF, RFC 1930, 1996.

[42] Iffinder tool. http://www.caida.org/tools/measurement/iffinder.

[43] Internet World Statistics. http://www.internetworldstats.com/stats.htm.

[44] Internet Routing Registries. `http://www.irr.net/`.

[45] Internet Systems Consortium. http://www.isc.org/.

[46] S. Jaiswal, A. L. Rosenberg, and D. Towsley. Comparing the Structure of Power-Law Graphs and the Internet AS Graph. In *ICNP*, 2004.

[47] S. Kandula, D. Katabi, B. Davie, and A. Charny. Walking the Tightrope: Responsive yet Stable Traffic Engineering. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, volume 35, August 2005.

[48] kc claffy, T. E. Monk, and D. McRobb. Internet Tomography. *Nature*, January 1999. `http://www.caida.org/tools/measurement/skitter/`.

[49] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh. 2003.

[50] D. Krioukov, F. Chung, kc claffy, M. Fomenkov, A. Vespignani, and W. Willinger. The Workshop on Internet Topology (WIT) Report. *Computer Communication Review*, 37(1), 2007.

[51] D. Krioukov, K. Fall, and X. Yang. Compact Routing on Internet-Like Graphs. In *INFOCOM*, 2004.

[52] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet Routing Convergence. In *Proceedings of the ACM SIGCOMM Conference*, September 2000.

[53] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling Biases in IP Topology Measurements. In *INFOCOM*, 2003.

[54] L. Li, D. Alderson, W. Willinger, and J. Doyle. A First-Principles Approach to Understanding the Internets Router-Level Topology. In *Proceedings of the ACM SIGCOMM Conference*, 2004.

[55] H. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An Information Plane for Distributed Services. In *OSDI*, November 2006.

[56] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat. Systematic Topology Analysis and Generation using Degree Correlations. In *Proceedings of the ACM SIGCOMM Conference*, September 2006.

[57] Z. M. Mao, L. Qiu, J. Wang, and Y. Zhang. On AS-Level Path Inference. In *SIGMETRICS*, 2005.

[58] Z. M. Mao, J. Rexford, J. Wang, and R. H. Katz. Towards an Accurate AS-Level Traceroute Tool. In *Proceedings of the ACM SIGCOMM Conference*, 2003.

[59] E. Marcotte, M. P. H. Ng, D. Yeates, and D. Eisenberg. Detecting Protein Function and Protein-protein Interactions from Genome Sequences. In *Science, volume 285, pages 751-753*, 1999.

[60] S. Maslov, K. Sneppen, and A. Zaliznyak. Detection of Topological Patterns in Complex Networks: Correlation Profile of the Internet. *Physica A*, 333:529–540, 2004.

[61] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An Approach to Universal Topology Generation. In *MASCOTS*, 2001.

[62] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation-of-State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087, 1953.

[63] M. Molloy and B. Reed. A Critical Point for Random Graphs With a Given Degree Sequence. *Random Structures and Algorithms*, 6:161–179, 1995.

[64] J. Moreno. Who Shall Survive? Beacon House, Beacon, NY 1934.

[65] W. Muhlbauer, A. Feldmann, O. Maennel, M. Roughan, and S. Uhlig. Building an AS-Topology Model. In *Proceedings of the ACM SIGCOMM Conference*, 2006.

[66] M. Mzruchi. The American Corporate Network,1904-1974. Sage, Beverly Hills, 1982.

[67] M. Newman. The Structure and Function of Complex Networks. `http://arxiv.org/cond-mat/0303516v1`.

[68] M. Newman, S. Strogatz, and D. Watts. Random graphs with arbitrary degree distributions and their applications. Physics Review E 64, 026118 (2001).

[69] M. E. J. Newman. Assortative Mixing in Networks. *Physical Review Letters*, 89:208701, 2002.

[70] M. E. J. Newman. Properties of Highly Clustered Networks. *Physical Review E*, 68:026121, 2003.

[71] T. S. E. Ng and H. Zhang. A Network Positioning System for the Internet. In *Proceedings of the USENIX Annual technical Conference (USENIX)*, 2004.

[72] The DIMES Project. http://www.netdimes.org.

[73] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[74] V. Paxson and S. Floyd. Why We Don't Know How to Simulate the Internet. In *WSC '97: Proceedings of the 29th conference on Winter simulation*, pages 1037–1044, New York, NY, USA, 1997. ACM Press.

[75] F. Pazos and A. Valencia. Similarity of Phylogenetic Trees as Indicator of Protein-protein Interaction. In *Protein Engineering 9(14), pages 609-614*, 2001.

[76] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach.* SIAM, Philadelphia, PA, 2000.

[77] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A Blueprint for Introducing Disruptive Technology into the Internet. In *ACM HotNets*, October 2002.

[78] E. Prisner. *Graph Dynamics*. Longman, Harlow, 1995.

[79] T. G. R. Teixeira, A. Shaikh and J. Rexford. Dynamics of Hot-Potato Routing in IP Networks. June 2004.

[80] T. G. R. Teixeira, A. Shaikh and G. Voelker. Network Sensitivity to Hot-Potato Disruptions. August 2004.

[81] P. Radoslavov, H. Tangmunarunkit, H. Yu, R. Govindan, S. Shenker, and D. Estrin. On Characterizing Network Topologies and Analyzing their Impact on Protocol Design. Technical Report USC-CS-TR-00-731, March 2000.

[82] Y. Rekhter and T. Li. *A Border Gateway Protocol 4 (BGP-4).* IETF, RFC 1771, 1995.

[83] University of Oregon RouteViews Project. `http://www.routeviews.org/`.

[84] A. Rowstron and P. Druschel. Storage Management and Caching in PAST, a Large-Scale, Persistent Peer-to-Peer Storage Utility. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP'01)*, October 2001.

[85] M. A. Serrano and M. Boguñá. Tuning Clustering in Random Networks with Arbitrary Degree Distributions. *Physical Review E*, 72:036133, 2005.

[86] C. Shannon and D. Moore. The Spread of the Witty Worm. In *Proceedings of IEEE Security and Privacy*, July 2004.

[87] Y. Shavitt and T. Tankel. Big-Bang Simulation for Embedding Network Distances in Euclidean Space. In *Proceedings of IEEE INFOCOM*, April 2003.

[88] G. Siganos and M. Faloutsos. Analyzing BGP Policies: Methodology and Tool. In *INFOCOM*, 2004.

[89] N. J. A. Sloane. Sequence A001349. The On-Line Encyclopedia of Integer Sequences. `http://www.research.att.com/projects/OEIS?Anum=A001349`.

[90] S. N. Soffer and A. Vázquez. Clustering Coefficient Without Degree Correlations Biases. `http://arxiv.org/abs/cond-mat/0409686`.

[91] N. Spring, R. Mahajan, and T. Anderson. Quantifying the Causes of Path Inflation. In *Proceedings of the ACM SIGCOMM Conference*, 2003.

[92] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *Proceedings of the ACM SIGCOMM Conference*, August 2002.

[93] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer to Peer Lookup Service for Internet Applications. In *Proceedings of the 2001 SIGCOMM*, August 2001.

[94] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. Network Topology Generators: Degree-Based vs. Structural. In *Proceedings of the ACM SIGCOMM Conference*, 2002.

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

[96] S. Tauro, C. Palmer, G. Siganos, and M. Faloutsos. A Simple Conceptual Model for the Internet Topology. In *Global Internet*, 2001.

[97] `traceroute`. `http://www.traceroute.org/#source%20code`.

[98] A. Vahdat, K. Yocum, K. Walsh, P. Mahadevan, D. Kostic, J. Chase, and D. Becker. Scalability and Accuracy in a Large-Scale Network Emulator. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, December 2002.

[99] F. Viger and M. Latapy. Efficient and Simple Generation of Random Simple Connected Graphs with Prescribed Degree Sequence. In *COCOON*, 2005.

[100] D. Vukadinović, P. Huang, and T. Erlebach. A Spectral Analysis of the Internet Topology. Technical Report TIK-NR. 118, ETH, 2001.

[101] B. M. Waxman. Routing of Multipoint Connections. *IEEE JSAC*, 1988.

[102] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An Integrated Experimental Environment for Distributed Systems and Networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, December 2002.

[103] J. Winick and S. Jamin. Inet-3.0: Internet Topology Generator. Technical Report UM-CSE-TR-456-02, University of Michigan, 2002.

[104] B. Wong, A. Slivkins, and E. Sirer. Meridian: A Lightweight Network Location Service without Virtual Coordinates. In *Proceedings of the ACM SIGCOMM Conference*, August 2005.

[105] B. Wong, I. Stoyanov, and E. Sirer. Octant: A Comprehensive Framework for the Geolocalization of Internet Hosts. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2007.

[106] E. Zegura, K. Calvert, and S. Bhattacharjee. How to Model an Internetwork. In *INFOCOM*, 1996.

[107] B. Zhang, T. S. E. Ng, A. Nandi, R. H. Riedi, P. Druschel, and G. Wang. Measurement Based Analysis, Modeling, and Synthesis of the Internet Delay Space. In *Proceedings of the ACM Internet Measurement Conference*, 2006.

[108] S. Zhou and R. J. Mondragón. Accurately Modeling the Internet Topology. *Physical Review E*, 70:066108, 2004. `http://arxiv.org/abs/cs.NI/0402011`.

[109] S. Zhou and R. J. Mondragón. The Rich-Club Phenomenon in the Internet. *IEEE Communications Letters*, 8 No.3, March 2004.