CHARACTERIZING TRAFFIC-AWARE OVERLAY TOPOLOGIES: A MACHINE LEARNING APPROACH

by

BENJAMIN DAVID MCBRIDE

B.S., Kansas State University, 2003

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Electrical and Computer Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2007

Approved by:

Major Professor
Caterina Scoglio

Copyright

Abstract

Overlay networks are application-layer networks that are constructed using the existing Internet infrastructure. Nodes in an overlay network construct logical links toward other nodes to form an overlay topology. Common routing algorithms, such as the link state and distance vector algorithms, are then used to determine how to route data in the overlay network. Previous work has demonstrated that overlay networks can be used to improve routing performance in the Internet. These quality of service improvements make overlay networks attractive for a variety of network applications.

Recently, game-theoretic approaches to constructing overlay network topologies have been proposed. In these approaches, nodes establish logical links toward other nodes in a decentralized and selfish manner. Despite the selfish behavior, it has been shown that desirable global network properties emerge. These approaches, however, neglect the traffic-demand between nodes. In this thesis, a game-theoretical approach is presented to constructing overlay network topologies that considers the traffic-demand between nodes. This thesis shows that the traffic-demand between nodes has a significant effect on the topologies formed. Nodes with statistically higher traffic-demand from others become members of the graph center, while nodes that have statistically higher traffic-demand toward others establish logical links toward members of the graph center. This thesis also shows that a traffic-demand aware overlay network topology is better suited to transport the required traffic in the overlay network.

Unfortunately, the game-theoretic approach is intractable. In order to construct larger overlay networks, approximate or heuristic approaches are required. In this thesis, a machine learning approach is proposed that characterizes the attributes of neighbor nodes during the construction of the overlay network topology. The approach proposed uses this knowledge

and experience to learn a set of human-readable rules. This rule set is then used to decide whether to construct a logical link toward a node. This thesis shows that the machine learning approach results in similar overlay network topologies as the game-theoretic approach. Additionally, it is shown that the machine learning approach is tractable and scales to larger networks.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost I thank the Lord Jesus Christ, without whom I could do nothing.

I would like to thank Amber for her support and patience during the whole process. I would like to thank my advisor, Dr. Scoglio, for her direction and guidance during the research process. I would like to thank Drs. Gruenbacher and Hsu for their wisdom and insights. Lastly, I thank my officemates for putting up with me.

Chapter 1

Introduction

This thesis proposes a cost model for overlay network creation that includes the traffic-demand between nodes and introduces a machine learning approach for low-cost overlay network formation. I believe that the traffic-demand between nodes is a critical parameter in constructing overlay network topologies that efficiently route and carry data in the network. I believe that the consideration of traffic-demand in overlay network formation will dramatically affect the resulting network topologies.

This chapter introduces the Internet routing mechanisms that motivate this work. I also introduce the concept of overlay networks and briefly review how they work and the tradeoffs involved in their use. Finally, I touch on the contributions of the work. Namely, the introduction of a traffic-demand aware cost model, empirical results showing the necessity of traffic-demand consideration when forming overlay network topologies, and a machine learning approach for low-cost network formation.

1.1 Motivation

Routing in the Internet is not optimal. The Internet is not just one network. It is a network of networks. Individual networks are called autonomous systems (AS). Each AS is free to select routes internally based on different performance metrics, such as latency, bandwidth, loss rate, etc. The AS can also choose whether or not to publicize the availability of certain routes to other AS's. Each AS also has different economic arrangements with other AS's

to route data from some networks, but not others. As a result, paths between hosts in the Internet are not optimized for end-to-end performance.

Since end-to-end performance is not optimized for routing in the Internet, and hosts have no control over the routing of data, quality of service (QoS) is a difficult problem. Many applications would like to have performance and reliability guarantees. Unfortunately, the Internet provides zero guarantees. Many solutions have been proposed to add QoS to the infrastructure of the Internet. These solutions have little chance of being accepted due to the inertia of the existing system. However, a popular approach that has been recently proposed is the idea of an *overlay routing network*, or overlay network.

1.2 Overlay Network Theory

An overlay network is an *application-layer* network composed of logical links over the existing Internet infrastructure, Figure 1.1. In an overlay network, the hosts route data through intermediate notes in the overlay network according to application specific performance metrics. Surprisingly, the end-to-end routing performance can be significantly improved by routing through intermediate nodes[2] [1]. This is a result of the hierarchical organization and tremendous redundancy that is in the Internet.

Obviously, one of the fundamental problems in overlay networks is deciding which intermediate nodes to use. Typically, overlay networks construct a virtual topology of logical links toward other nodes in the overlay network. Common routing algorithms, such as the link-state or distance vector algorithm, are then used to route data according to an application-specific performance metric. This thesis focuses on a particular method of constructing the overlay network topology.

1.3 Contribution

The primary contribution of this work is the introduction and investigation of a game-theoretic model for constructing overlay network topologies that considers the traffic-demand

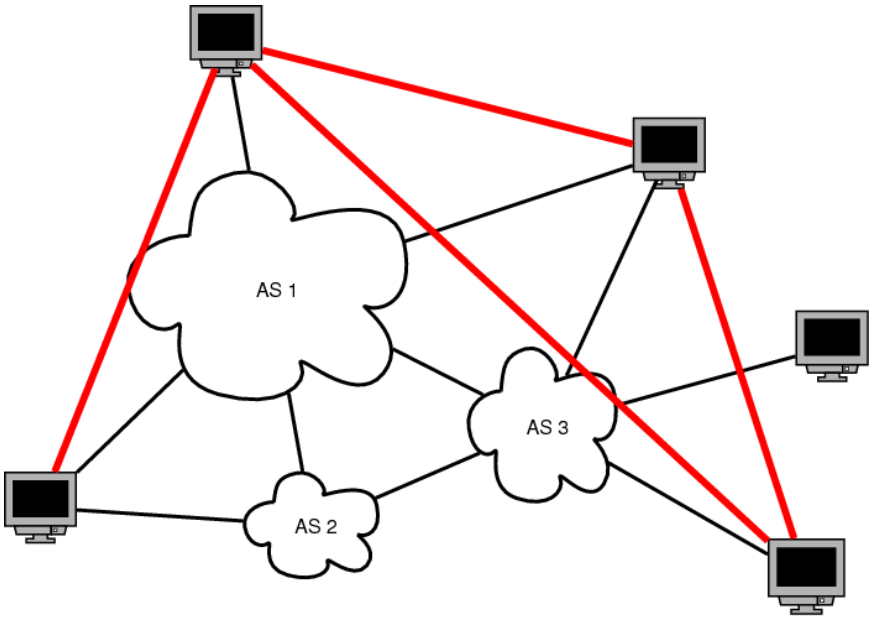


Figure 1.1: Example overlay network. Hosts are connected to different autonomous systems (AS). Each AS has arrangements with the other AS's to route traffic. As a result, end-to-end routing is not optimal. By routing through intermediate nodes, end-to-end performance can be improved.

between nodes. Previous work in selfishly constructed overlay network topologies has ignored the traffic-demand between nodes. This thesis shows that consideration of the traffic-demand between nodes has a significant effect on the topologies that are constructed.

Additionally, this thesis proves that constructing minimal-cost overlay network topologies using the traffic-demand aware cost model is intractable. This result suggests that heuristic approaches are necessary. A greedy hill-climbing search is also introduced to construct low-cost overlay network topologies. Initial results show that the greedy hill-climbing search produces topologies that closely approximate the minimal-cost topologies.

Unfortunately, even the greedy hill-climbing method for constructing low-cost overlay network topologies does not scale well to large networks. As a result, I propose a machine learning approach to construct low-cost overlay network topologies that is computationally more efficient than the greedy hill-climbing search. This thesis compares the topologies formed using the machine learning approach with the minimal-cost overlay network topolo-

gies and the topologies formed by the greedy hill-climbing search. The results show that the machine learning approach has promise and is viable for forming networks with many different parameter values. However, further refinement in the methodology and approach is needed for the approach to be completely viable.

Chapter 2

Background and Related Work

In this chapter I review previous and related work to this thesis. We begin with a look at overlay networks in general with a focus on the network topology and network formation differences that characterize different overlay networks. We examine structured overlay networks that are formed by a global protocol for creating links between peers. We then address overlay networks that are formed in a more ad-hoc manner. Finally, I examine prior research on selfishly constructed overlay networks.

2.1 Structured Overlay Networks

Overlay networks are commonly classified by how the peers are linked to each other. Structured overlay networks are formed using a consistent, global protocol for creating links between peers. This structure ensures that data can be efficiently located and routed over the overlay network. Two common types of structured overlay networks are Resilient Overlay Networks (RONs) and Dynamic Hash Table (DHT) networks.

2.1.1 Resilient Overlay Networks

The classic example of a structured overlay network is the Resilient Overlay Network (RON) architecture [4]. The primary goal of the RON architecture is to allow the communication between RON nodes in "the face of problems with underlying Internet paths connecting them" [4]. This seminal work in overlay networks allows for the discovery and recovery of

path failures and degradation. Nodes aggressively probe and monitor the links in the overlay network. When a path between nodes has an outage or extreme performance degradation, data is routed through an alternate path in the overlay network. The exchange of path quality and routing information allows RON to quickly discover path outages and recover from them. The RON results show that RON was able to detect and recover from all the significant path outages in less than twenty seconds on average.

The secondary goal of the RON architecture is to integrate the routing and path selection with the application [4]. Traditionally, the routing and path selection has been entirely contained in the Internet core. Hosts had no way to select or even prefer alternate paths. Using a RON, however, allows application specific metrics to be used in the path selection. An application can select paths based on a variety of metrics such as loss rates, latency, or throughput. The RON results show that dramatic improvements in both TCP throughput and loss rate can be achieved by using alternate paths. This result is significant from a QoS standpoint. This result also opens the possibility for an "Overlay ISP" where a RON is formed between different ISPs and improved QoS is provided to the customer [4].

In the RON architecture, each node has a logical link to all other RON nodes. This means that the number of logical links and, consequently, the probing costs grow quadratically with the number of nodes. Because of this RONs do not scale beyond about fifty nodes. A RON of this size is certainly beneficial, but the lack of scalability limits the applications. Another consequence of the RON topology is that many of the links contain duplicate physical links in the underlying network. This finding has been used to build RONs with sparser topologies and a larger number of nodes [5].

2.1.2 Distributed Hash Table

Distributed Hash Table (DHT) overlay networks are another type of structured overlay network that address the problem of scale. Systems that employ a DHT include Pastry [6], Tapestry [7], CAN [8], and CHORD [9]. All DHT overlay networks view the overlay network

as a distributed data structure that governs both the network topology and routing. Each node requires routing tables on the order $O(\log n)$ and guarantees that messages will be routed in at most $O(\log n)$ hops. This makes the DHT systems massively scalable.

Each node in a DHT is given a unique identifier. Given a message and a key, routing is performed by iteratively routing the message closer to the key in the node identifier space[6]. While most DHT overlay networks achieve similar state and routing efficiency the underlying topologies realized differ significantly [10]. These topological differences are important when considering the performance and the resilience of the overlay network. For example, Tapestry realizes a tree topology that is not very resilient. On the other hand, CAN's topology is similar to a hypercube and consequently is highly resilient to node failure. The CHORD topology is a ring, while Pastry's geometry is a hybrid between a tree and a ring. Other topologies that are realized in DHT overlay networks are butterfly networks[11] and the XOR geometry[12].

This focus on massive scalability comes at the cost of decreased performance. This is because the routing is based primarily on a random addressing scheme and secondarily on maximizing the route performance. While the routing scheme ensures that data can be routed to every node, there is little guarantee on the performance of the selected path. Additionally, DHT overlay networks can not easily integrate with application-specific performance requirements. As a result DHT systems are primarily used in P2P applications where scalability and object location and routing are the primary design goals.

2.2 Unstructured Overlay Networks

In contrast to structured overlay networks, unstructured overlay networks are formed when overlay links are created arbitrarily. Each peer determines its neighbors in the overlay network. Data is located by a flooding query. This flexibility can result in high overhead for locating and routing data, but allows the network to easily adapt to peers joining and leaving the network. Unstructured overlay networks are commonly used in peer-to-peer (P2P)

networks, like Gnutella, where this flexibility is critical. Unfortunately, this flexibility in the overlay network topology makes the topologies harder to characterize. For the purposes of characterizing the network topologies, I will use node degree information and other network attributes to discriminate between different topologies [13].

2.2.1 Gnutella

One of the more popular P2P systems that employ overlay networks is Gnutella[14]. The Gnutella network topology is composed of two-tiers of nodes. Nodes are either ultra-peers or leaf peers. The ultra-peers form a top-level overlay network, while leaf peers connect to one or more ultra-peers[15]. Only the ultra-peers forward messages in the network. Routing and object location queries are flooded outward between ultra-peers in the network. This flooding can limit the scalability of the network if not restricted.

Early efforts at characterizing the topology of the Gnutella network [16]showed that Gnutella was primarily a power-law network [17]. Although, later results showed that there were too few nodes with low degree to form a pure power-law network. It was observed that this multi-modal distribution results in higher resilience to node failure and attack [16].

More recent efforts have refuted the idea that the Gnutella network can be characterized as a power-law network[15]. These efforts show that the power-law characteristics observed are a result of slow network crawlers. These slow crawlers show that peers with long uptimes have high degree when in reality they have many short-lived peers that report them as a neighbor, but the short-lived peers are not all present at the same time. Using a more efficient network crawler shows that the Gnutella network exhibits characteristics of a small world network[18], such as clustering and short path lengths[15]. These results show that the Gnutella network is "onion-like" where long-lived peers form a stable core that is highly resilient to node failure and attack[15].

2.2.2 Connect and Improve

Many unstructured overlay networks take a "connect and improve" approach to constructing the network topology. These approaches first construct a dense overlay network or "mesh" and then use local improvement techniques to adapt the topology by removing "bad" links. The Narada[19] protocol constructs its overlay topology in this type of two-step process. Each node continually probes the other nodes in the network looking to add or drop links based on the perceived utility of the link. Unfortunately, this procedure has significant probing costs and in the case of Narada, each node stores information about all other nodes.

A similar approach uses interleaved spanning trees to construct the overlay topology[20]. Like Narada, a dense mesh is first created. Then a distributed algorithm is used to construct minimum spanning trees[21] (MST) over the edges of the mesh. Different performance metrics can be considered for the edge weights, insuring many alternate paths between nodes in the network. By using a topology composed of k trees, k edge disjoint paths between any two nodes are guaranteed. This promotes resilience to node failures and attacks as well as to extended periods of performance degradation.

Criticisms of these approaches include scalability and convergence time. These types of "connect and improve" strategies and tree-based connection strategies tend to scale poorly. Consequently, most of the applications for these networks are ones that require resilience and high performance guarantees, like A/V multicast, streaming video applications, and multi-path routing. Additionally, the slower convergence times for the overlay network topology means that applications where nodes frequently join and leave the network are discouraged.

2.2.3 Selfishly Constructed Overlay Networks

Recently, a new approach from game theory has been proposed for the formation of network topologies. These studies propose a network creation model where each node tries to minimize its own "cost", which is defined according to a cost model, without consideration

for the global interests of the network. Each node will continue to minimize its own cost until an equilibrium is reached where no node can decrease its own cost without some other node first increasing its cost. At this point the network is stable since nodes always act in a selfish or greedy fashion. This equilibrium is called a *Nash equilibrium*. This thesis is based on this unstructured overlay network approach. We discuss some of the seminal and closely related work with selfishly constructed overlay networks in this section.

A game-theoretic approach

In a game-theoretic approach to network creation, each node in the network acts as an independent agent, selecting its neighbors, and paying for logical links to those neighbors according a cost model[22]. The subset of nodes that form the neighborhood of a node is called a *strategy*. Each node acts selfishly, selecting a strategy that reduces its overall cost. Once a logical link is paid for, any node in the network can use that link to route data. The union of these logical links forms a network that is created without central design or coordination.

Obviously, the resulting network topologies are largely dependent on the specific cost model used. The cost for node i to select strategy B_i as proposed by Fabrikant, et al is[22]:

$$C_i = \alpha|B_i| + \sum_{j \in N} d_G(i,j) \tag{2.1}$$

where $d_G(i,j)$ is the cost distance between nodes i and j in the overlay network G and α is a parameter balancing the cost of creating links to other nodes and the distance to other nodes in the overlay network. These different cost terms can be seen as hardware and quality of service costs or probing and routing costs, depending on the application. It turns out that the topology of the resulting networks is dependent on the balance between these two cost terms.

By varying the value of α, a wide range of network topologies can be constructed. When $\alpha < 1$, the Nash equilibrium is a complete graph. When $1 \leq \alpha < 2$, a Nash equilibrium is of diameter at most 2. The worst Nash equilibrium in this case is the star topology,

but other equilibriums do exist that are more densely connected. When $\alpha < 2$, the star is a Nash equilibrium, but other worse equilibriums may exist. As α becomes larger, most Nash equilibriums are trees. While the Nash equilibriums exist, they may not be the same topology as the social optimum solution. The difference between the Nash equilibrium and the social optimum solution is called the *price of anarchy*. Fabrikant, et al continue their analysis of this game by providing upper and lower bounds on the price of anarchy. The important result of this work is that a large variety of network topologies can be formed in a decentralized manner by nodes acting in a selfish manner.

A generalized approach

Chun, et al, generalized the cost model proposed by Fabrikant, et al, and studied the selfishly constructed networks formed by the non-cooperative game[23]. They generalized the cost model in three areas:

1. The link cost is no longer constant, but rather is a function of the node j being connected to.

2. The distance function is generalized to be any performance metric between nodes.

3. The possible neighbors that a node can connect to is constrained.

The resulting cost model is:

$$C_i = \alpha \sum_{j \in B_i} h_{i,j} + \sum_{j \in N} d_G(i,j) \tag{2.2}$$

where $h_{i,j}$ is the cost to create a logical link between i and j.

Chun, et al, first consider small network topologies formed with varying values of α and different link cost functions. An iterative exhaustive search, where each node in turn finds its minimal cost strategy given the rest of the network, is used until the network reaches equilibrium. They demonstrate that a wide range of network topologies can be constructed: complete graphs, densely connected graphs, sparsely connected graphs, stars,

k-core stars, and trees. When considering a more realistic search, where a greedy search is done over a constrained neighborhood set, they find that networks can be constructed that have desirable *global* properties. They demonstrate that power-law networks can be formed as well as networks that are highly resistant to node failure and attack. They also conclude that there is a fundamental tradeoff between the performance of a network and its resilience.

Others

Moscibroda, Schmid, and Wattenhofer[24] have proposed a cost model for network creation that is very similar to the model presented by Fabrikant, et al[22]. Their cost model exploits locality properties by using the stretch between nodes in place of the distance in the overlay network. The stretch is the ratio of the routing distance in the overlay network and the routing distance in the underlay network. The cost model presented is:

$$C_i = \alpha|B_i| + \sum_{j \in N} stretch_G(i,j) \tag{2.3}$$

where $stretch_G(i,j) = d_G(i,j)/d(i,j)$. Moscibroda, Schmid, and Wattenhofer then present upper and lower bounds on the price of anarchy for the cost model. They also investigate the Nash equilibrium that exist. Unfortunately, unlike the model presented by Fabrikant, et al., there does always exist a Nash equilibrium.

Christin and Chuang[25] present a cost model based on the resources each node has. Their model is a function of the experienced load of forwarding traffic on behalf of other nodes and node connectivity. They consider four cost components in their analysis of different overlay network topologies: latency cost, service cost, routing cost, and maintenance cost. Unlike this work, they do not consider network formation. They consider the cost of a network as a fundamental characteristic of the network itself.

Anshelevich, Dasgupta, Tardos, and Wexler[26] present a network design game where each edge needs to connect to a set of terminal nodes. Unlike the model by Fabrikant, et al., nodes can share the cost of edges and can pay for non-adjacent edges. They do not consider

the topologies formed from the network design game.

Fabrikant, et al.[22] actually suggest the extension to their cost model to consider traffic congestion. Instead of weighing distances between all node pairs equally, they suggest multiplying the distance term by the traffic between the two nodes. Albers, et al.[27], provide bounds on the price of anarchy for this weighted network creation game. Our work builds on this extension and the work of Chun, et al[23].

Chapter 3

Problem Formulation

I begin this chapter by presenting basic definitions relevant to overlay network creation. I then present the traffic-demand aware cost model for overlay network creation. Finally, a proof is provided that shows the problem of finding a node's minimal cost strategy is NP-hard.

3.1 A Traffic Demand Aware cost model

I assume that each node needs to select its neighbors in a distributed and decentralized fashion. Additionally, nodes have no knowledge of other nodes' neighbors. That is, nodes have imperfect information.

Definition Let $G = (N, L)$ be an undirected graph representing the *overlay* network and let $G_u = (N, E)$ be the graph representing the *underlay*, or physical, network where N is the set of nodes that are in both the overlay and physical network, while the set of logical links L can be different from the set of physical links E.

Definition A *logical link* $l \in L$ between nodes $i, j \in N$ is constructed on a path composed of physical links $e \in E$.

Definition Let $t_{i,j}$ be the *traffic-demand* between nodes i and j. Each node $i \in N$ has a traffic-demand toward a subset of nodes $S_i \subseteq N$.

Definition The *strategy* for node i is the subset of nodes, $B_i \in 2^{N-\{i\}}$, that logical links are made toward, that is, for each $j \in B_i$ there exists a logical link between nodes i and j that is paid for by node i.

We define the cost using two components:

1. The *link cost* is the cost to create and maintain a the logical links in the node's strategy.

2. The *transport cost* is the cost to route, or transport, the traffic-demand through the overlay network.

The overall cost for a node is then the sum of the link cost and the transport cost:

$$C_i = \alpha \sum_{j \in B_i} h_{i,j} + \sum_{j \in S_i} d_G(i,j) t_{i,j} \tag{3.1}$$

The link cost is the sum of the linking costs, given by the function $h_{i,j}$, over all the nodes in the strategy. The transport cost is the summation, over all the nodes that i has a traffic-demand towards, of the distance in the overlay network between nodes, $d_G(i,j)$ (∞ if j is unreachable from i), multiplied by the traffic-demand between nodes. The relative weight between the link cost and the transport cost is controlled by α. The total cost of the overlay network is defined as:

$$C_G = \sum_{i \in N} C_i \tag{3.2}$$

It is important to note that the linking cost between i and j, is a general function that can represent a wide variety of metrics. The transport cost term in (3.1) can also be thought of as a generalized distance function [23]. Additionally, once a logical link has been established from i to j any node in the network can use the link. We do not consider the link to be directed in terms of its use and for calculation of d_G.

Objective The objective for each node is to find its minimal cost strategy, the strategy that minimizes the nodes cost according to Equation 3.1.

3.2 NP-hardness proof

We use proof by restriction to show that finding a node's minimal cost strategy is NP-hard[28]. We begin by assuming that there is a constant, unit traffic-demand between all nodes, $t_{i,j} = 1$ for all nodes i and j. Consequently, Equation 3.1 reduces to:

$$C_i = \alpha \sum_{j \in B_i} h_{i,j} + \sum_{j \in N} d_G(i,j)$$

This is the cost model presented by Chun, et al[23], see Equation 2.2. Let us further restrict Equation 2.2 by setting the overlay creation cost function to one, $h_{i,j} = 1$ for all nodes i and j. Equation 2.2 then reduces to the cost model presented by Fabrikant, et al, Equation 2.1:

$$C_i = \alpha |B_i| + \sum_{j \in N} d_G(i,j)$$

The remainder of this proof is taken from a sketch given by Fabrikant, et al[22]. We start with a basic definition.

Definition A *dominating set* of a graph $G = (V, E)$ is a subset $V' \subseteq V$ such that for all $u \in V - V'$ there is a $v \in V'$ for which u and v are adjacent.

We restrict α to the range $1 < \alpha < 2$. and assume there are no incoming logical links to node i. The minimal cost strategy for node i is now a minimal dominating set for the rest of the graph.

Claim 1. *The diameter of G is at most two.*

Proof. Suppose the minimal cost network had a node j at a distance greater than two hops from any node i. If a logical link was added between i and j, the link cost would increase by α, while the distance to j, and consequently, the transport cost, would be reduced by at least two. Since $\alpha < 2 \leq \Delta d_G(i,j)$, a lower cost strategy would exist for i, contradicting the supposition that the minimal cost network had a node j at a distance greater than two hops from any node i. □

Claim 2. *The cardinality of the strategy is minimal, that is the number of links are minimal.*

Proof. Suppose a lower cost strategy with greater cardinality exists. Since it has already been shown that the diameter of G is at most two, each additional link would decrease $d_G(i,j)$ by at most one. However, α is greater than one and consequently, $\alpha > \Delta d_G(i,j)$. This means that the increase in the link cost would be greater than the corresponding decrease in transport cost, resulting in a strategy that has *greater* cost. Thus the number of links are minimal. □

Since finding the minimal dominating set for a graph is NP-complete[28], the problem of finding the low cost strategy for a node is NP-hard.

Chapter 4

Characterizing Traffic Demand Aware Overlay Network Topologies

In this chapter, I present the methods used to characterize traffic-demand aware overlay network topologies. The algorithms used to search the node strategy space are introduced. Then I present the traffic-demand distributions used. Finally, results are given for topologies of varying size.

4.1 Methodology

4.1.1 Search algorithms

There are two search algorithms that are primarily used to create traffic-demand aware topologies. One is an exhaustive search of the strategy space that is used on smaller networks since it is intractable. The second is a greedy search that is used on larger networks. In this section, details of both search techniques are presented.

Iterative Exhaustive Search

The first search algorithm used is an iterative exhaustive search (IES) that is suggested by Chun, et al[23]. In the IES procedure, each node, in turn, chooses a strategy that minimizes its cost with respect to the rest of the graph. This process is repeated until a Nash equilibrium is reached, that is, no node can lower its cost without some other node

first increasing its cost. Algorithm 4.1.1 is an implementation of the IES procedure.

Algorithm 4.1.1 Iterative Exhaustive Search algorithm

```
atEquilibrium ← ∅
while atEquilibrium ≠ N do
  for each n ∈ N do
    B_n ← minimum cost strategy
    if strategy has changed then
      atEquilibrium ← ∅
    else
      atEquilibrium ← atEquilibrium ∪ n
    end if
  end for
end while
```

Since the IES procedure is an exhaustive search through the strategy space, this algorithm is intractable, as proven in Section 3.2.

Iterative Greedy Search

The second search algorithm used is iterative greedy search (IGS). IGS is similar to the IES procedure in that each node, in turn, chooses a strategy until a Nash equilibrium is reached. However, instead of doing an exhaustive search of the node strategy space, a greedy, hill-climbing search is used. The greedy search starts with the current strategy set. It then calculates the cost improvement for changing the membership in the strategy set for each node. The action that results in the largest cost improvement is selected. This process is repeated until no more cost improvements can be made.

In the worst case, IGS calculates the cost of $O(N^2)$ strategies. Each cost calculation involves a shortest path calculation with complexity on the order of $O(N^2)$ at worst, possibly $O(N \log(N))$ if a Fibonacci heap is used for the min-priority queue[21]. This results in a complexity of $O(N^3 \log(N))$ in the worst case. Additionally, convergence to the Nash equilibrium is not guaranteed. Because of this, the number of iterations over all nodes is limited to ten.

4.1.2 Traffic demand distributions

For the baseline comparison, the traffic-demand between nodes is assumed to be a constant, unit value. Thus the traffic-demand aware cost model is reduced to the traffic-demand ignorant model[23]. I then compare the topologies formed using the traffic-demand aware cost model when several different traffic-demand distributions are used. The first traffic-demand distribution used is a normal distribution with mean and variance of one. Negative traffic-demand values are rounded to zero.

I next introduce the idea of a *popular* node. A popular node has traffic-demand towards it that is significantly higher than the mean traffic-demand between nodes. A popular node distribution is created using the previously described normal traffic-demand, but with an increase of the mean traffic-demand towards node 4 by a factor of five. Node 4 was arbitrarily selected from the leaf nodes in the network to be popular.

I also examine the effect of *greedy* nodes on the network topology. A greedy node has significantly higher traffic-demand towards other nodes than the mean traffic-demand between nodes. The normal traffic-demand distribution was modified such that the mean traffic-demand of node 4 toward other nodes was increased by a factor of 5.

Finally, the effects of high traffic-demand links on the network topology are considered.

4.1.3 Constraining the node degree

Two link cost functions are considered in this work. The first is where the link cost between two nodes is always one. For the second link cost function, we desire to constrain the maximum node degree. To do this the link cost function that was presented by Chun, et al[23] is used:

$$h_{i,j} = \begin{cases} 1 & \text{if } degree(i) < maxDegree \text{ AND} \\ & \ degree(j) < maxDegree \\ \infty & \text{otherwise} \end{cases}$$

This function could also be used when the maximum degree is unconstrained by setting $maxDegree = N$.

4.2 Results

This section presents the topologies that are formed using various cost models, search algorithms, and traffic-demand distributions. A fully-connected underlay, where the distance between each node is one, is assumed. The topologies are created from Python[29] simulations. Graph statistics are gathered using the NetworkX[30] library and the network visualizations are created using Graphviz[31].

Iterative exhaustive search results are only presented for 20-node networks because of the computational complexity. Partial results are presented for the iterative greedy search results. Additional IGS results are included in Appendix B. IGS results for 100-node networks are touched on as well, with additional results in Appendix C.

For discussion on the graph metrics used please refer to the work by Mahadevan, et. al[13].

4.2.1 IES Results

We first compare the topologies created when traffic-demand between nodes is ignored with those created when normal traffic-demand between nodes is considered. For these comparisons the maximum node degree is not constrained. Figure 4.1 shows the topologies formed when $\alpha = 0.5$. When traffic-demand between nodes is ignored, the topology is a complete graph. For the traffic-demand aware cost model, the topology is densely connected. Considering the traffic-demand increases the transport cost slightly, but the link cost is greatly reduced as the number of edges in the graph is reduced to 132.

On the other hand, when $\alpha = 1$, as in Figure 4.2, the link cost is higher when traffic-demand is considered as the number of edges increases from 19 to 89. However, the transport cost is reduced when the traffic-demand is considered. It should be noted that our results differ from Chun et al[23] with respect to $\alpha = 1$. We found the equilibrium topology to be a star rather than a partially connected graph when the traffic-demand between nodes is ignored. However, it is noted by Fabrikant et al[22] that the star topology is also a Nash

equilibrium, albeit the worst one.

When $\alpha = 5$, both cost models create star topologies as shown in Figure 4.3. When $\alpha = 60$, both cost models produce trees with diameter of 4, as shown in Figure 4.4. The node degree distributions in Figures 4.5, 4.6, 4.7, and 4.8 provide an alternate view of the topologies that are formed.

We next constrained the maximum degree of a node to four. For $\alpha \leq 5$, there was little change in the overall topologies formed when considering normal traffic-demands. When $\alpha = 60$, both cost models produced trees with a diameter of six and equivalent node degree distributions. However, the traffic-demand aware topology had a slightly higher characteristic path length, 3.43 and 3.31, respectively, than the traffic-demand aware topology. The differences between the two cost models becomes more interesting when different traffic-demand distributions are considered.

We next considered the effects of popular and greedy nodes on the topologies formed. We expect that an increased number of logical links will be made towards popular nodes, while greedy nodes will make more logical nodes towards others. We also expect that the topologies formed will adapt to take advantage of these types of nodes.

For these experiments node 4 is either a popular or greedy node, depending on the experiment. All other traffic-demand values were taken from the Normal traffic-demand distribution. For both the popular and greedy nodes the degree of node 4 has increased, Table 4.1 and Table 4.2, respectively. However, the increase in node degree occurs differently. For the popular node, a change in the node in degree is the cause of the increased degree. Other nodes are creating logical links towards the popular node. On the other hand, the greedy node has an increased out degree as a result of creating extra links to other nodes. When $\alpha < 60$, these effects are clearly seen. Figure 4.9 shows that the popular and greedy nodes have indeed become the graph center. While not shown, the popular and greedy node becomes a member of the graph center for $\alpha < 5$ as well.

When $\alpha = 60$ the effects of the popular and greedy nodes on the topology are not clearly

seen. In fact, when the maximum node degree is not constrained, there is no change in the topology with the inclusion of the popular node. When the maximum node degree is constrained, the popular node becomes a member of the graph center, rather than the graph periphery, as shown in Figure 4.11. Greedy nodes do something very different though. Instead of becoming a member of the graph center, the greedy node, Figure 4.10, creates a logical link towards the graph center! This reduces the transport cost for the greedy node dramatically, while minimizing the link cost.

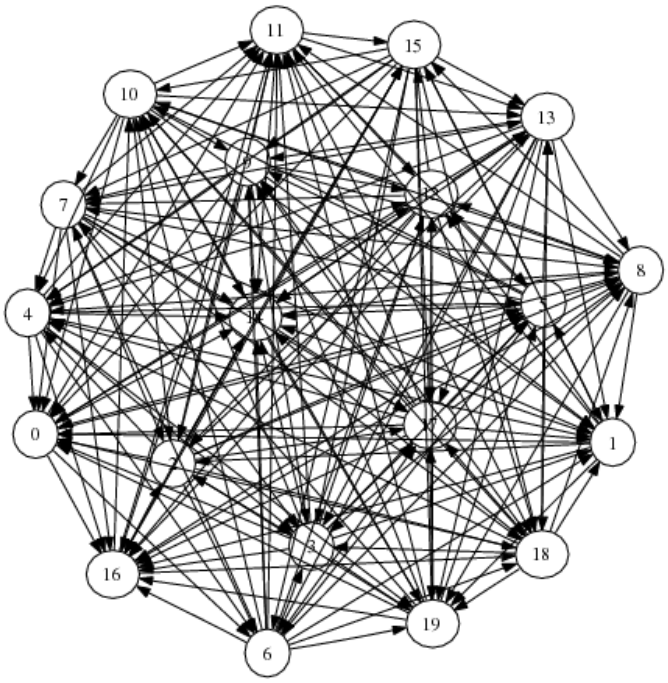
We next considered increased traffic-demand between just two nodes. This differs from the popular and greedy nodes in that the traffic-demand remains the same towards the other nodes; there is just a specific node towards which there is high traffic-demand. Figure 4.12 shows the network topology for such a scenario. Surprisingly, the graph is no longer a tree. The basic star topology can clearly be identified, but additional logical links have formed where there is high traffic-demand between the nodes.

Table 4.1: Degree of popular node for varying values of α

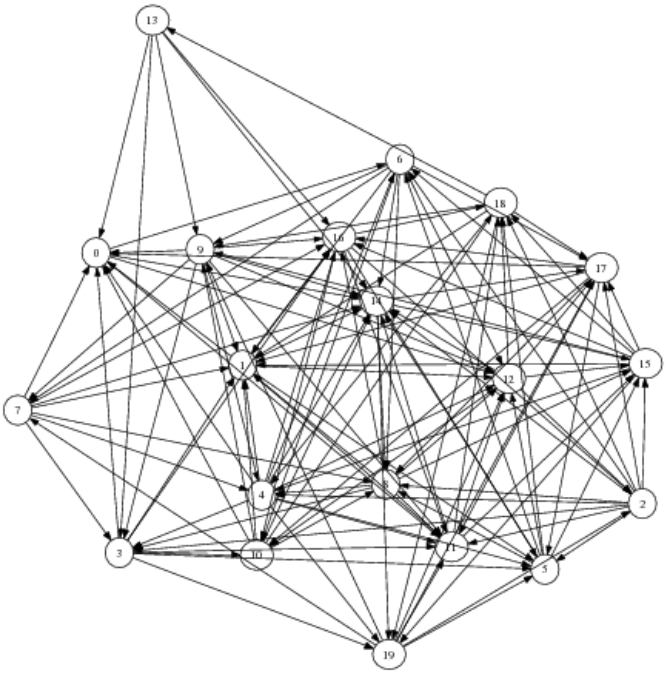
	Normal traffic-demand		Node 4 is *popular*	
α	degree	in degree	degree	in degree
0.5	16	6	19	9
1	12	5	19	12
5	1	0	19	19
60	1	1	1	1

Table 4.2: Degree of greedy node for varying values of α

	Normal Traffic Demand		Node 4 is *greedy*	
α	degree	out degree	degree	out degree
0.5	16	10	19	13
1	12	7	19	14
5	1	1	19	16
60	1	0	1	1

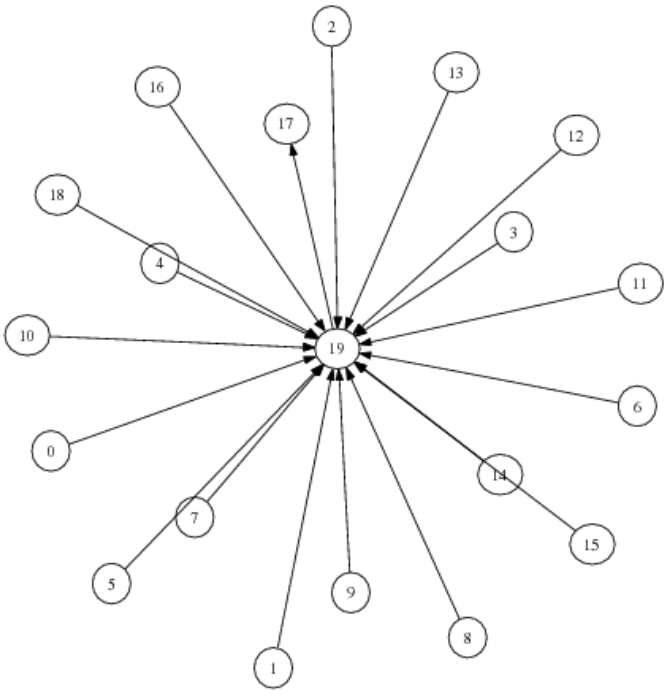


(a) Traffic demand ignorant

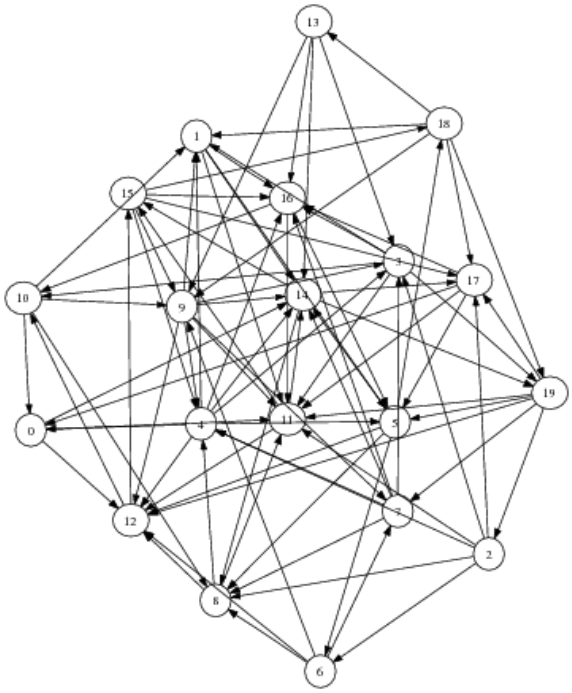


(b) Traffic demand aware

Figure 4.1: Topology comparison for $\alpha = 0.5$. Figure (a) is created using the traffic-demand ignorant cost model. The resulting topology is a complete graph. Figure (b) is formed using the traffic-demand aware cost model. The graph is densely connected, with average node degree of 13.2.

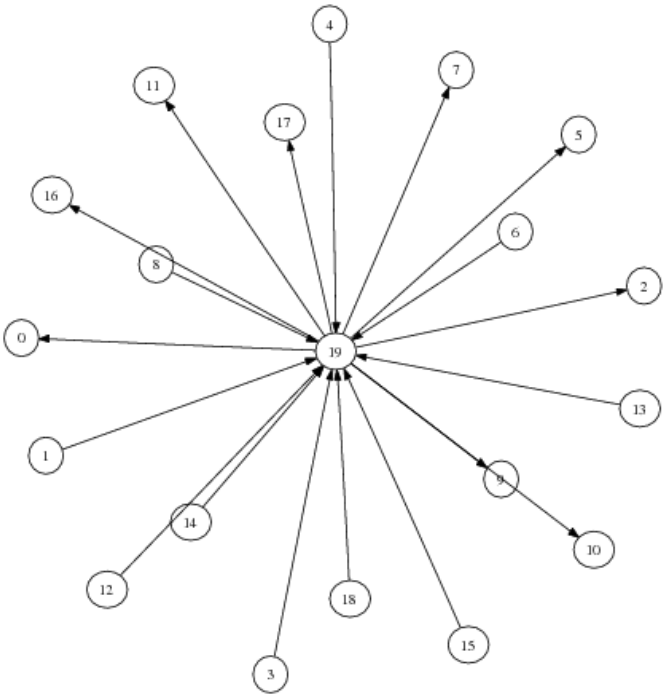


(a) Traffic demand ignorant

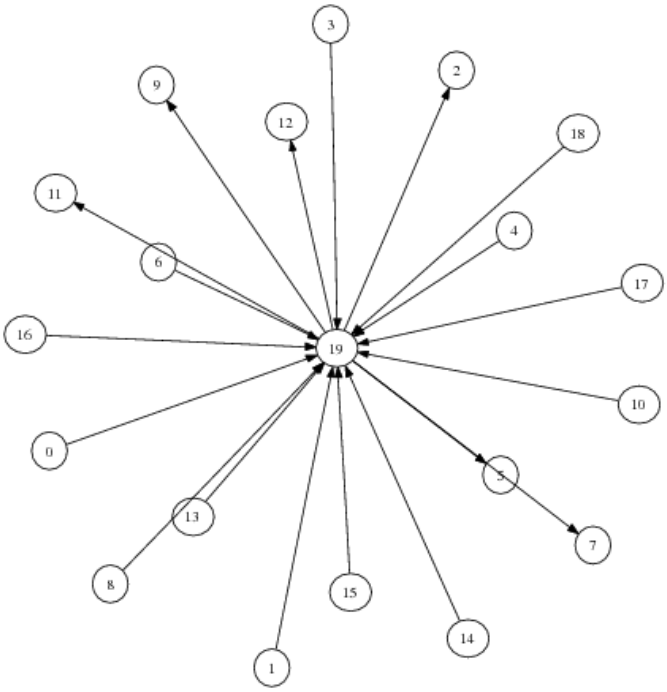


(b) Traffic demand aware

Figure 4.2: Topology comparison for $\alpha = 1$. Figure (a) is created using the traffic-demand ignorant cost model. The star is not the only Nash equilibrium in this case. A partially connected graph, with diameter at most 2, is also a Nash equilibrium. The graph in Figure (b) has an average node degree of 8.9 and diameter of 2.

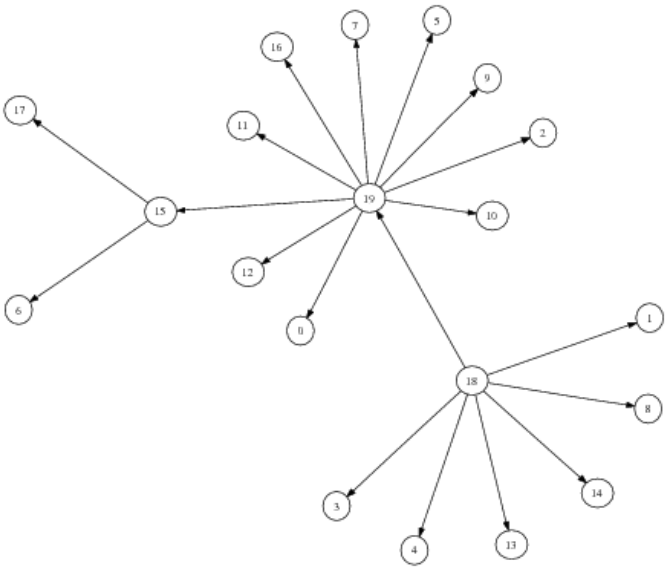


(a) Traffic demand ignorant

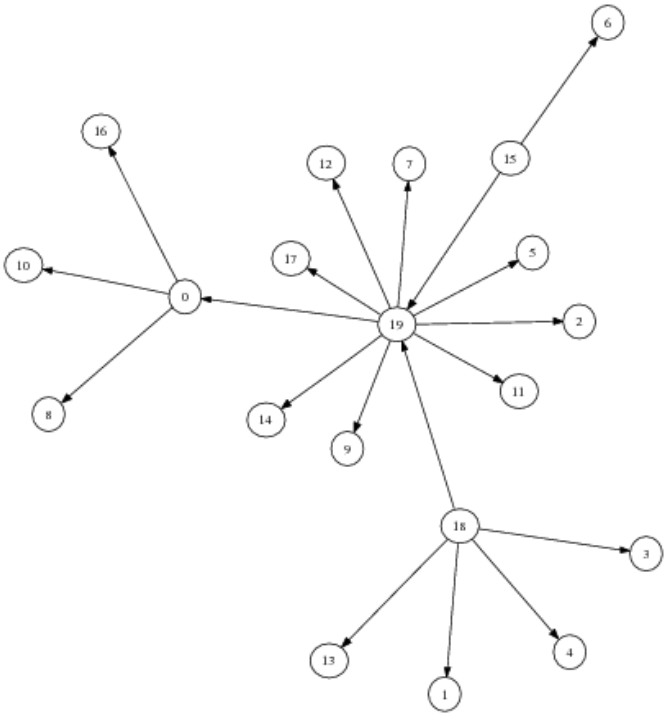


(b) Traffic demand aware

Figure 4.3: Both the traffic-demand ignorant, Figure (a), and the traffic-demand aware, Figure (b), cost models give star topologies when $\alpha = 5$. However, they are slightly different with respect to the strategies that are formed.

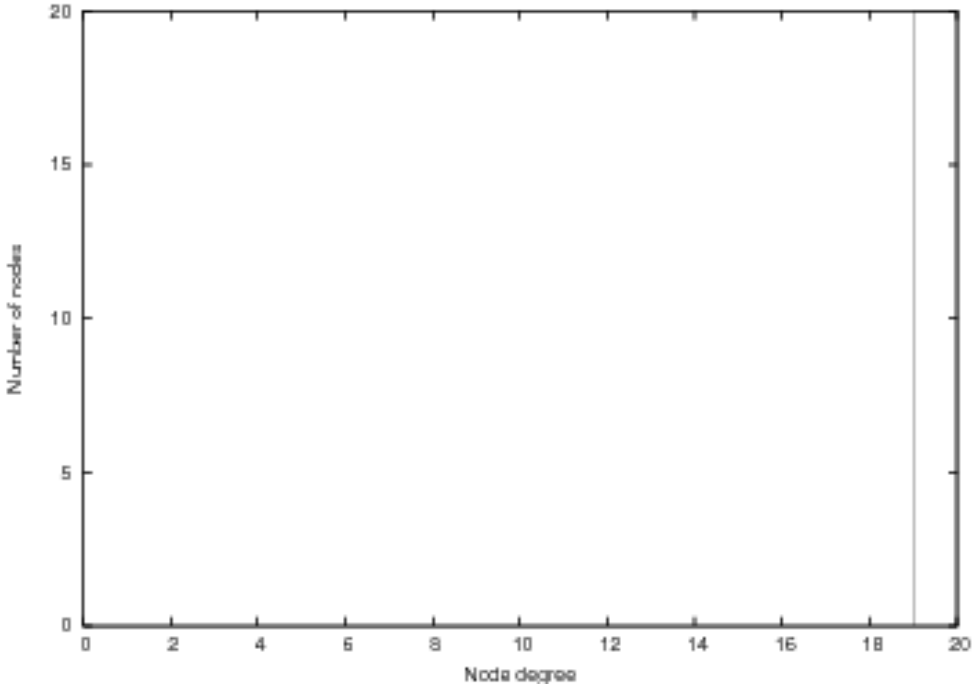


(a) Traffic demand ignorant

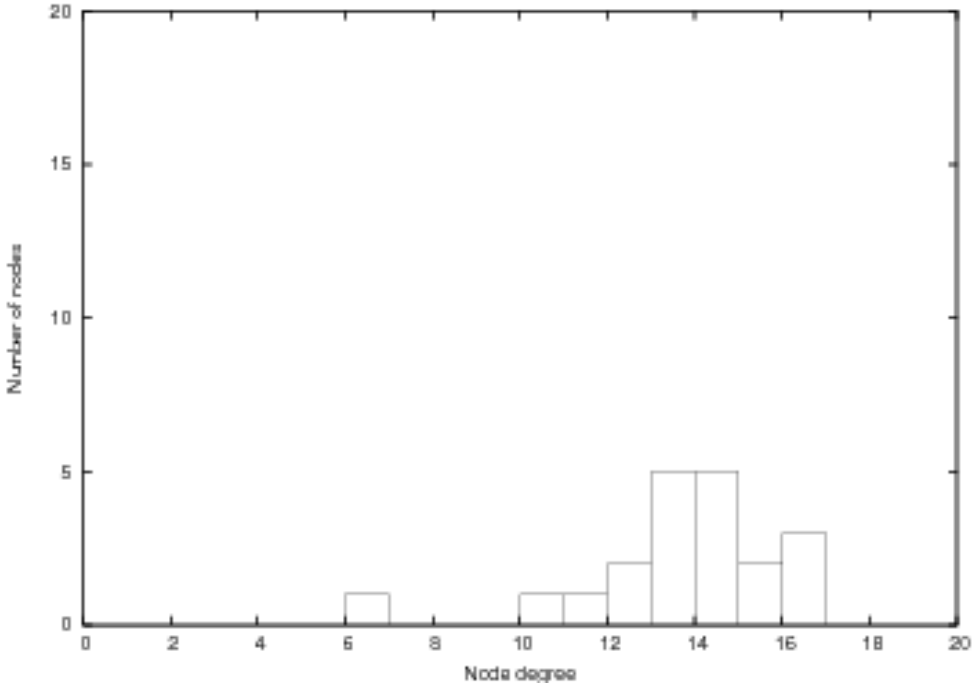


(b) Traffic demand aware

Figure 4.4: Both traffic models produce trees when $\alpha = 60$. The graph in Figure (a) has a characteristic path length of 2.45 compared to 2.52 for the graph in Figure (b).

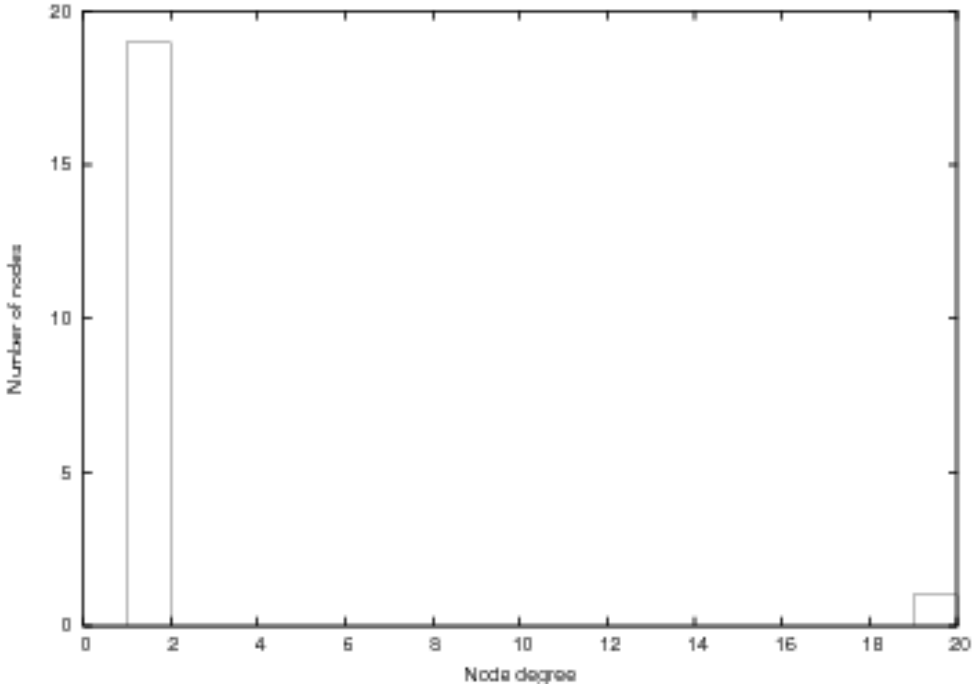


(a) Traffic-demand ignorant

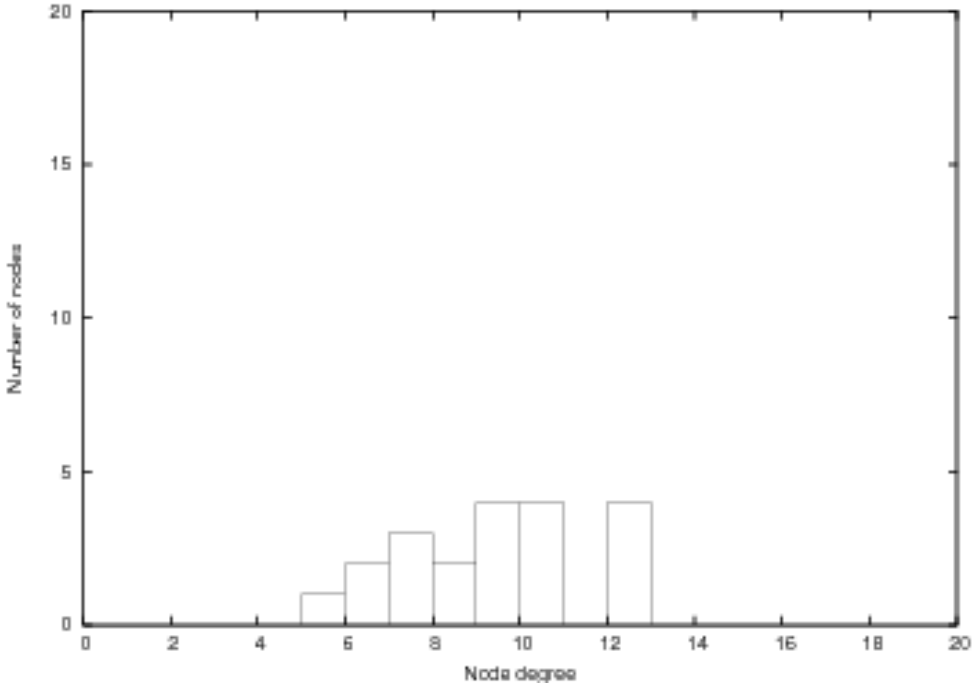


(b) Traffic-demand aware

Figure 4.5: Node degree distribution, $\alpha = 0.5$

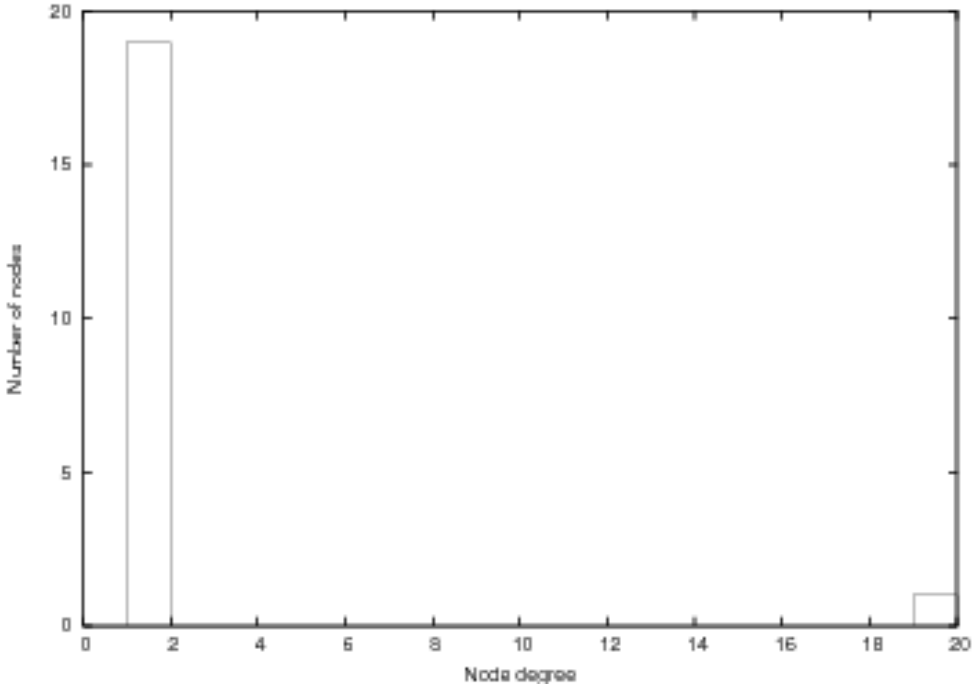


(a) Traffic-demand ignorant

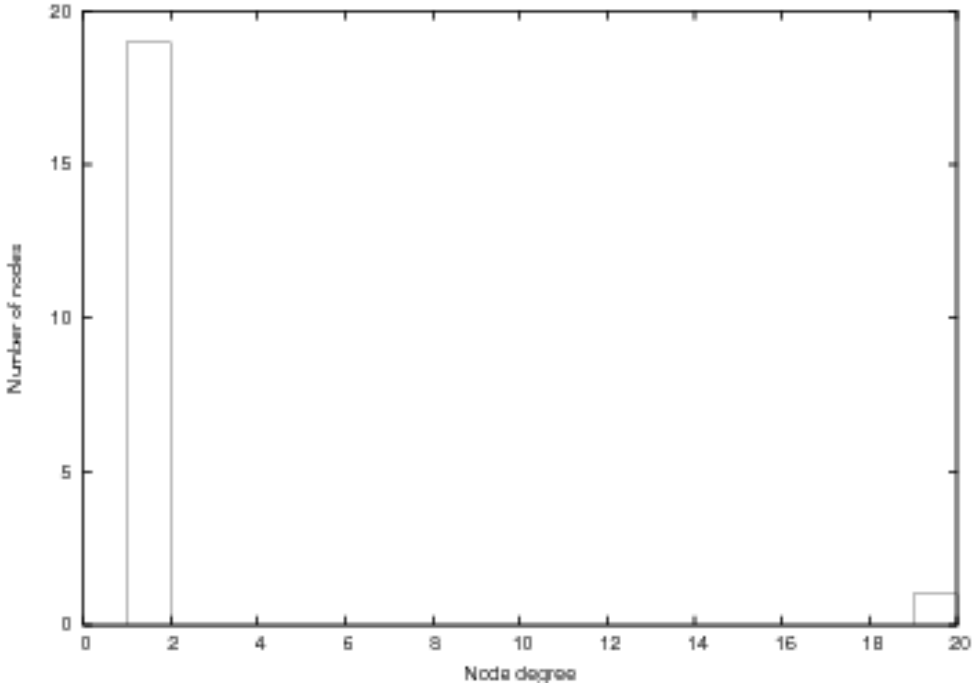


(b) Traffic-demand aware

Figure 4.6: Node degree distribution, $\alpha = 1$

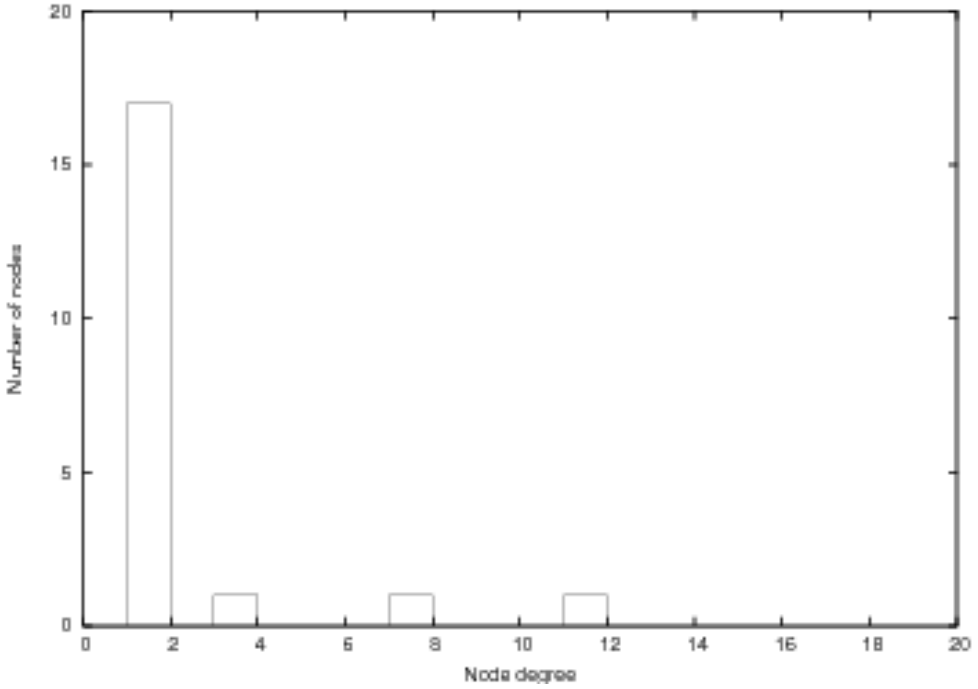


(a) Traffic-demand ignorant

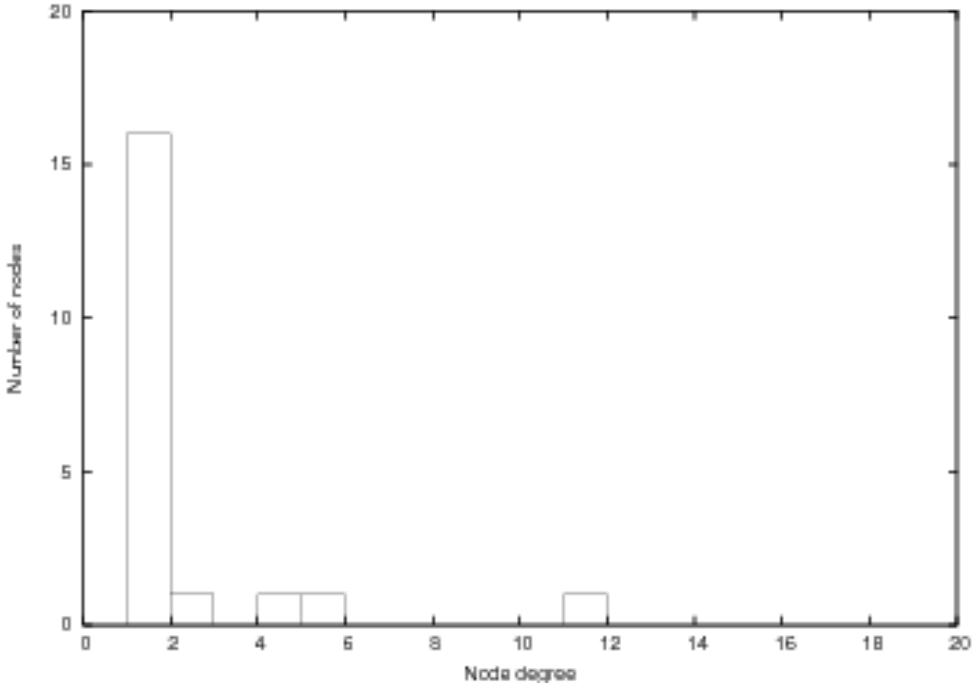


(b) Traffic-demand aware

Figure 4.7: Node degree distribution, $\alpha = 5$

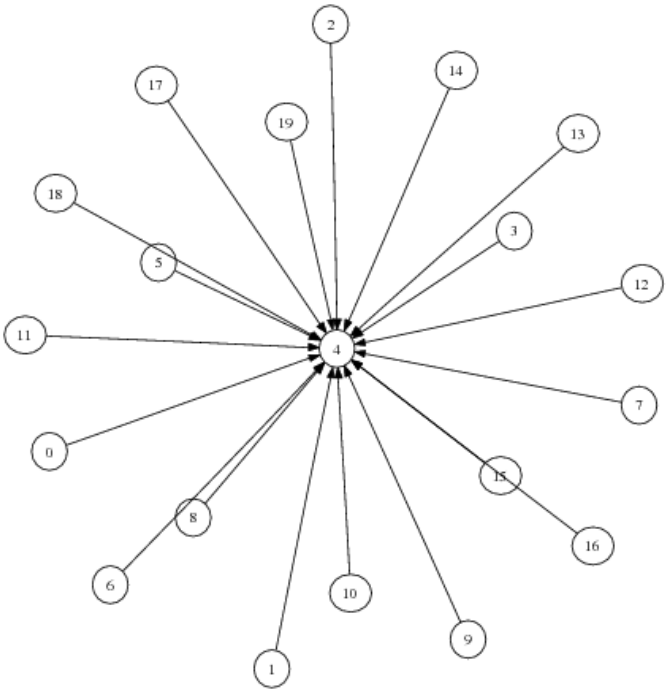


(a) Traffic-demand ignorant

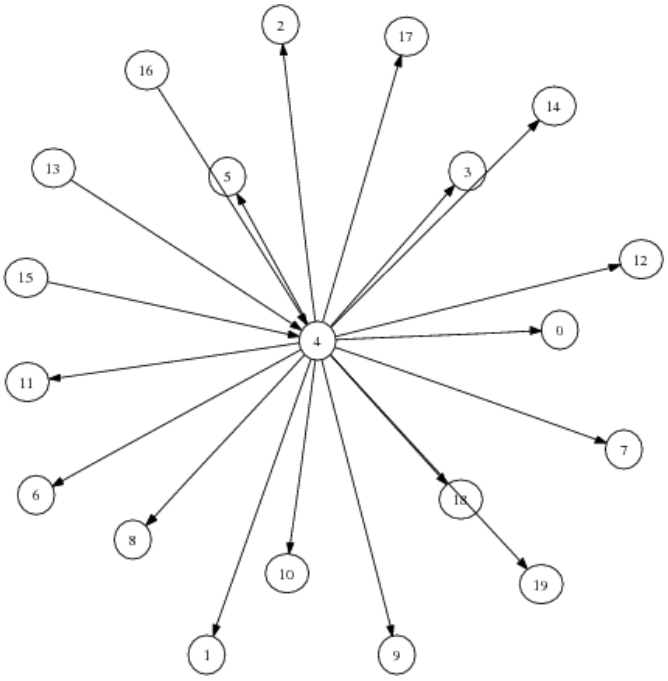


(b) Traffic-demand aware

Figure 4.8: Node degree distribution, $\alpha = 60$

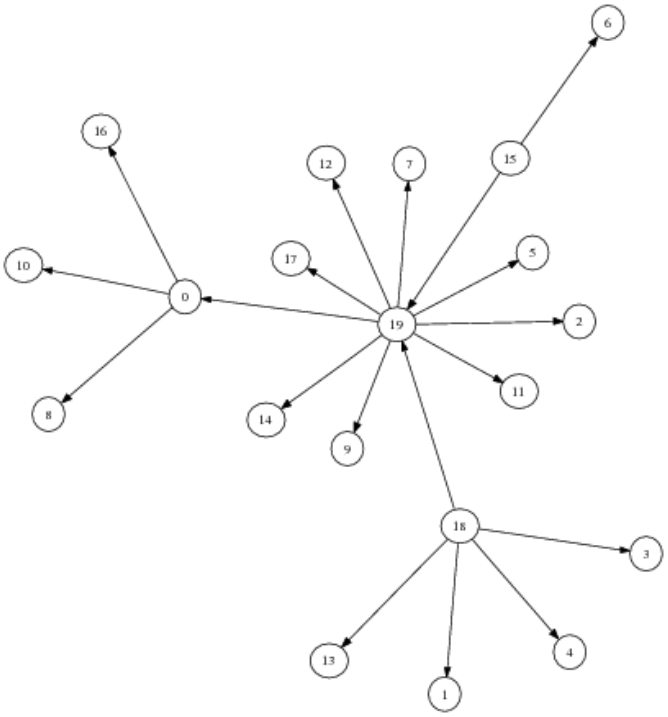


(a) Node 4 is popular

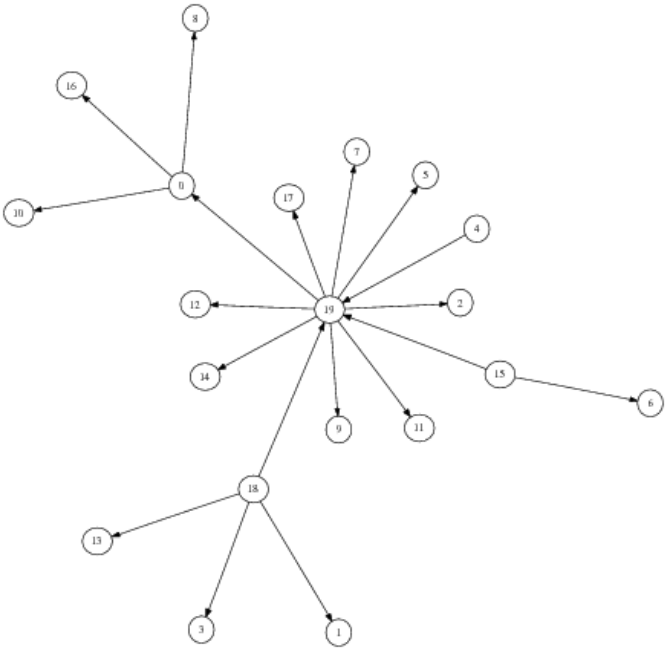


(b) Node 4 is greedy

Figure 4.9: Figures (a) and (b) show the network topology when node 4 is popular and greedy, respectively, and $\alpha = 5$. When node 4 is popular it becomes the center node because other nodes create logical links towards it. On the other hand, when node 4 has a high traffic-demand towards other nodes, it becomes the center node by creating logical links towards the other nodes.

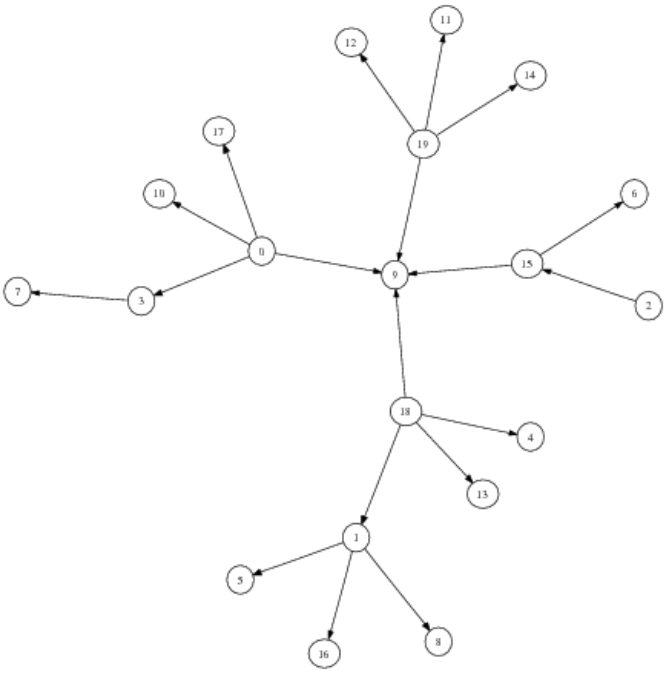


(a) Normal traffic-demand

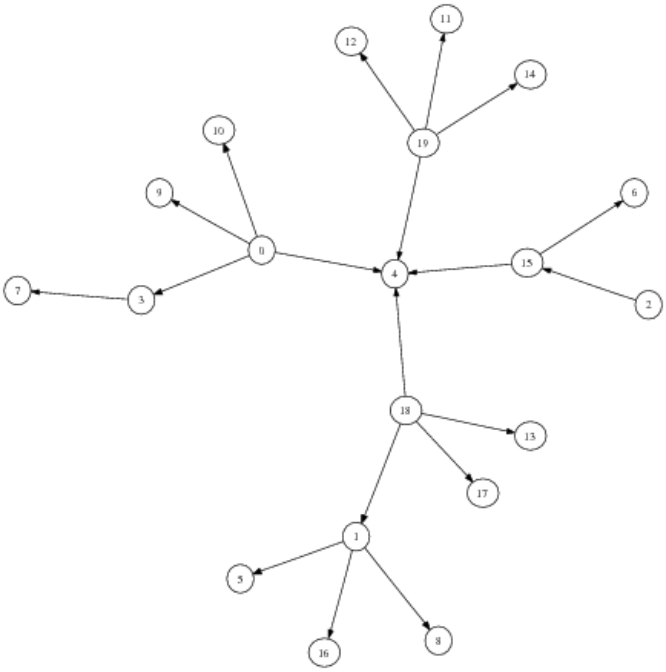


(b) Node 4 is greedy

Figure 4.10: Notice that node 4 is a periphery node that creates no logical links toward other nodes when normal traffic-demand is considered and $\alpha = 60$, Figure (a). However, when node 4 is greedy, it exploits the existing network topology and creates a logical link towards the center node, Figure (b).



(a) Normal traffic-demand



(b) Node 4 is popular

Figure 4.11: Network topologies for different traffic-demand models when the maximum node degree is constrained and $\alpha = 60$. Node 4 is a center node, Figure (b), rather than a periphery node, Figure (a), when the traffic-demand towards it is increased.

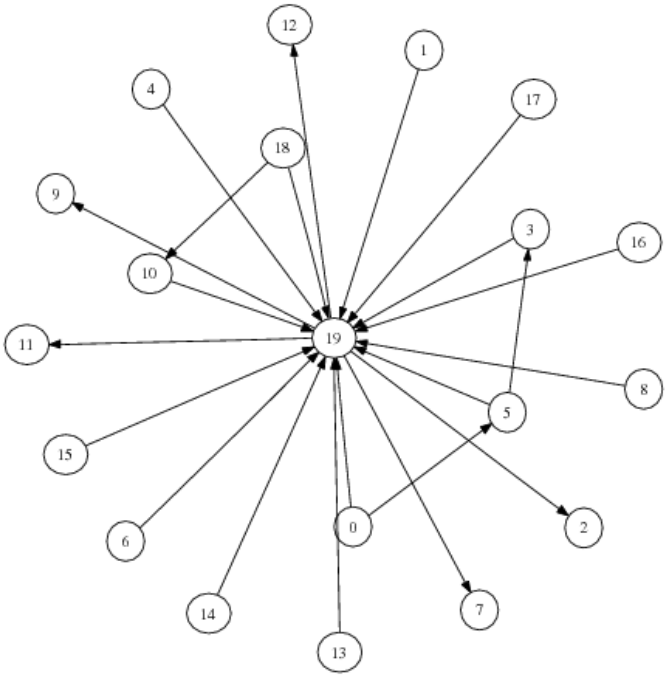


Figure 4.12: Network topology for $\alpha = 5$. Nodes 0, 5, and 18 have high traffic-demand towards nodes 5, 3, and 10, respectively, and therefore, create additional logical links towards them when traffic-demands are considered in the cost model.

4.2.2 IGS Search

This section presents different graph metrics comparing the topologies formed using the IES and IGS procedures. The number of edges, graph transport cost, diameter, characteristic path length, and spectral radius values are given in Table 4.3, Table 4.4, Table 4.5, Table 4.6, and Table 4.7, respectively. Results for the IGS procedure are the mean values over ten simulations. Normal traffic-demand was used and the maximum node degree was unconstrained, unless otherwise noted. The presence of popular and greedy nodes had a similar effect as with IES as shown in Table 4.8 and Table 4.9. Figures showing the resulting topologies and node degree distributions for $N = 20$ are provided in Appendix B. These results show that the IGS procedure performs comparably to the IES procedure when $N = 20$.

We also present some basic results for topologies formed using IGS when $N = 100$. Additional results for $N = 100$ are provided in Chapter 6 and Appendix C.

20-node results

Table 4.3: Number of edges

α	exhaustive	greedy	rules
0.5	132	132.0	144.1
1	89	89.0	115.5
5	19	20.3	20.2
60	19	19.0	19.0
200	19		19.0

Table 4.4: Graph transport cost

α	exhaustive	greedy	rules
0.5	387	387	381
1	449	449	416
5	714	710	883
60	930	1199	932
200	930		932

Table 4.5: Graph diameter

α	exhaustive	greedy	rules
0.5	2	2.0	2.0
1	2	2.0	2.0
5	2	2.2	4.9
60	4	6.2	4.3
200	4		4.3

Table 4.6: Graph characteristic path length

α	exhaustive	greedy	rules
0.5	1.30	1.30	1.24
1	1.53	1.53	1.39
5	1.90	1.91	2.48
60	2.52	3.23	2.48
200	2.52		2.48

Table 4.7: Graph spectral radius

α	exhaustive	greedy	rules
0.5	13.58	13.58	14.77
1	9.35	9.35	11.93
5	4.35	4.37	3.74
60	3.45	2.99	3.55
200	3.45		3.55

Table 4.8: Degree of popular node for varying values of α, using Greedy search

	Normal Traffic Demand		Node 4 is *popular*	
α	degree	in degree	degree	in degree
0.5	16.0	7.3	19.0	10.3
1	12.0	5.4	19.0	12.4
5	1.1	0.0	19.0	19.0
60	1.2	1.2	1.3	1.3

Table 4.9: Degree of greedy node for varying values of α, using Greedy search

	Normal Traffic Demand		Node 4 is *greedy*	
α	degree	out degree	degree	out degree
0.5	16.0	8.7	19.0	11.7
1	12.0	6.6	19.0	13.6
5	1.1	1.1	19.0	13.8
60	1.2	0.0	1.7	0.9

100-node results

For larger overlay network topologies, $N = 100$, the results are similar to those found in smaller networks. When α is small, densely connected graphs result, as shown by the node degree distribution in Figure **??**. As α gets larger, more star-like topologies are formed and when $\alpha \geq 200$ more tree-like topologies are formed, as shown in Figure 4.13.

The presence of popular and greedy nodes have a similar effect on the topologies as well. Popular and greedy nodes tend to move closer to the graph center on average. This is especially true when the maximum node degree is constrained. Additional results showing this are given in Appendix C.

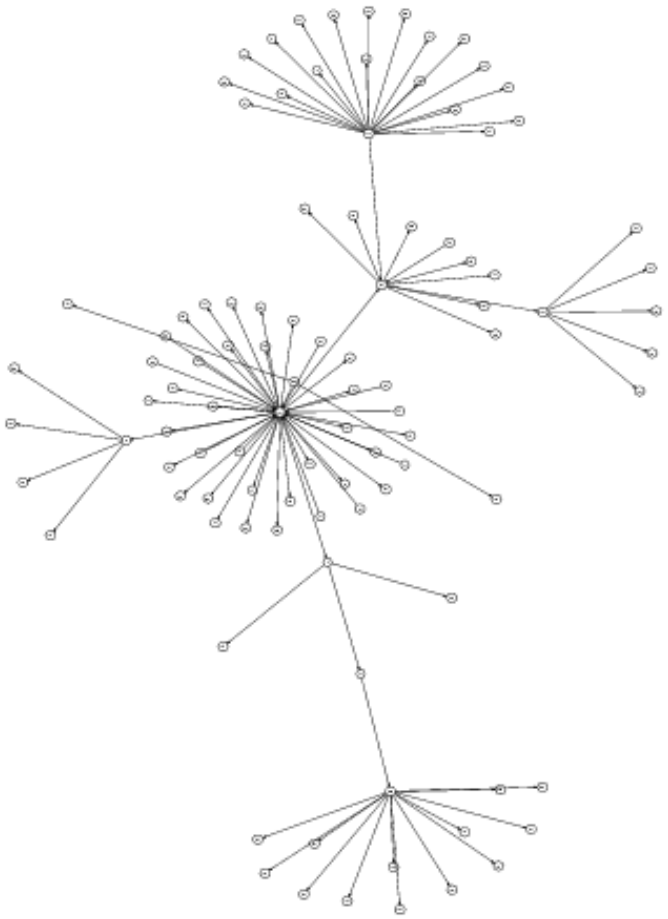


Figure 4.13: A representative topology formed using IGS with Normal traffic-demand, $\alpha = 200$, and unconstrained maximum node degree

Chapter 5

A Machine Learning Approach

I begin this chapter by touching on the motivation for using a machine learning approach to overlay network construction. The method for acquiring the training data is then discussed. Next, the process of selecting and creating the attributes that are used in the learning algorithm is detailed. The process and justification for selecting the JRip rule learning algorithm is given. Finally, I show how the learned rules are used in the context of overlay network construction.

5.1 Motivation

As discussed in Section 4.1.1, searching the node strategy space with the greedy hill-climbing search is $O(N^3 \log N)$. As the number of nodes in the overlay network increases, the runtime complexity becomes prohibitive. A algorithm with runtime complexity that is linear in the number of nodes is desirable. Clearly a heuristic or approximation approach is needed. Additionally, the number of distance calculations must be limited as calculating the single source shortest path length is $O(N \log N)$. Under these constraints a good heuristic is not obvious.

An approach that could characterize *good* nodes to make logical links towards is desired. A decision could then be made for each node, classifying whether a logical link is made or not. This is clearly a good domain for a supervised classification problem. A good definition of the learning task is given by Mitchell[32]:

Definition A computer program is said to *learn* from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

For the overlay network learning problem the task is clearly whether to create a logical link to a node or not. The performance measure is the number of correctly classified nodes and the experience is gathered from topologies formed using exhaustive and greedy search.

5.2 Data Acquisition

In order to get the training data for the learning algorithm, the decisions that each node makes using exhaustive or greedy search needs to be recorded. At each iteration of the IES or IGS search, when a strategy for a node has been selected, the algorithm records various statistics for each node and whether the node was selected as a member of the strategy. This process was repeated until the network reached equilibrium. For this thesis, the data was collected while running the IGS search for 20-node networks. Various values of α, different traffic distributions, and different maximum node degree constraints were used. This approach is somewhat flawed, as IGS is itself a heuristic approach. As a result, the learning algorithm can only be expected to perform as well as the IGS search. An improvement to this approach, would use results obtained from the IES procedure. This was not done, due to the simulation time needed to rerun the IES procedure.

5.3 Attribute Selection

After the raw data was collected, I did some attribute creation and selection. The first task was to create new attributes that were thought to be important. The first attribute created was $maxDegreeRatio$. This attribute is the ratio of a node's degree to the $maxDegree$ constraint. This attribute is really just a normalized version of the node degree. The second attribute created was $TrafficDemandToAlpha$. This also can be viewed as a way

of normalizing the traffic-demand and the α parameter. Table 5.1 gives the names and brief descriptions of all the attributes used.

Table 5.1: Attributes and their description in the full dataset

Attribute name	Type	Description
N	numeric	Number of nodes in the network
alpha	numeric	α
maxDegreeExceeded	boolean	Would the *maxDegree* constraint be violated by adding a link?
maxDegreeRatio	numeric	Ratio of the degree of node j to the *maxDegree*
ijTrafficDemand	numeric	Traffic demand between nodes i and j
jMeanDistance	numeric	Average distance from j to other nodes.
Degreej	numeric	The degree of node j
OutDegreej	numeric	The out degree of node j
InDegreej	numeric	The in degree of node j
jiLinkState	boolean	Does a logical link exist from j to i?
TrafficDemandToAlpha	numeric	Ratio of the traffic-demand to alpha
linkState	boolean	Is a logical link made towards node j?

The complete set of attributes represents the *full* dataset. As mentioned previously, it is desirable to remove all distance calculations from the classification task. We also wanted to reduce the number of attributes that needed to be tested in order to reduce the complexity of the learning problem. We ran two attribute selection algorithms, *CfsSubsetEval* and *InfoGainAttributeEval*, from the Waikato Environment for Knowledge Analysis (WEKA) [33] to determine the most relevant attributes.

The information gained from the different attributes is given in Table 5.2. As it turns out the mean distance from node j to other nodes does has a low information gain. This is promising since it is desirable to neglect the distance attribute. The attributes that were selected by the *CfsSubsetEval* algorithm were: *maxDegreeExceeded*, *maxDegreeRatio*, *Degreej*, *jiLinkState*, and *TrafficDemandToAlpha*. Once again the distance attribute was not selected. However, for both selection algorithms *Degreej* was selected. This is not desirable since it is not normalized. Instead we would like to use the normalized attribute *maxDe-*

Table 5.2: Information gain of attributes

Attribute name	Information Gain
maxDegreeRatio	0.1454
Degreej	0.1229
maxDegreeExceeded	0.05826
TrafficDemandToAlpha	0.05772
ijTrafficDemand	0.0571
InDegreej	0.03242
jMeanDistance	0.01837
alpha	0.01784
jiLinkState	0.01637
OutDegreej	0.00977
N	0

greeRatio. Based on these results the *reduced* dataset was created that included the following attributes: *maxDegreeExceeded*, *maxDegreeRatio*, *jiLinkState*, and *TrafficDemandToAlpha*.

5.4 Algorithm Selection

The next step in designing a learning system is to choose the target representation. The representation is closely tied to the choice of learning algorithm. Ideally, the representation would be human-readable. This rules out various neural network approaches like multi-layered perceptrons, radial basis functions, and support vector machines. For performance reasons, lazy learning algorithms such as nearest neighbor approaches were also ruled out. This left primarily decision tree and rule learning approaches. Since much of our data is numeric, the learning algorithm must be capable of handling numeric as well as nominal valued attributes. Based on preliminary experiments the following learning algorithms were selected for further investigation: *J48*, *JRip*, *PART*, and *Ridor*.

WEKA's Experimenter was used to compare the four learning algorithms against fifteen data sets. Five datasets were formed using random sampling from the *full* dataset. Five datasets were formed using random sampling from the *reduced* dataset. Five datasets were formed by random sampling, plus class balancing, from the *reduced* dataset. These datasets

are called *balanced*. Each dataset contained 10 percent of the examples from the complete dataset. They were reduced in size because of memory constraints associated with WEKA and the Java Virtual Machine.

Table 5.3 gives the percentage of correctly classified instances for each data set over each algorithm. Each algorithm was run on each dataset ten times. The mean values and standard deviations are given. As observed the Ridor algorithm performs statistically worse than J48, JRip, and PART. The percentage of correctly classified instances does not tell the whole story though, especially when considering an unbalanced dataset. Precision and recall values provide a different view of the predictive accuracy.

Definition *Precision* is the number of true positive examples compared to the number of examples that are classified as positive.

Definition *Recall* is the number of true positive examples compared to the number of examples that are actually positive.

Table 5.4 and Table 5.5 provide the precision and accuracy results of the learning algorithms. It is interesting to note that the Ridor algorithm has poor precision results, but high recall results. The final metric used in deciding which learning algorithm to use was the number of generated rules. Table 5.6 shows that the JRip algorithm produces a much smaller set of rules than either J48 or PART, while still maintaining comparable predictive accuracy. Consequently, the JRip learning algorithm was selected.

Table 5.3: Percent Correct

Data Set	J48	JRip	PART	Ridor
full1	97.67±0.50	97.58±0.48	97.51±0.53	97.00±0.67•
full2	97.60±0.47	97.53±0.54	97.46±0.63	97.04±0.65•
full3	97.14±0.50	97.35±0.55	97.20±0.54	96.97±0.82
full4	97.47±0.57	97.39±0.58	97.29±0.60	96.86±0.74•
full5	97.22±0.53	96.91±0.52•	96.99±0.60	96.56±0.75•
reduced1	97.37±0.43	97.33±0.47	97.43±0.44	97.12±0.63
reduced2	97.34±0.56	97.29±0.55	97.30±0.55	96.84±0.73•
reduced3	97.23±0.51	97.33±0.52	97.32±0.50	97.02±0.63
reduced4	97.32±0.53	97.33±0.57	97.43±0.54	96.91±0.76•
reduced5	96.99±0.55	97.01±0.52	97.01±0.52	96.42±0.68•
balanced1	93.30±0.94	93.15±0.86	93.16±0.95	92.31±1.24•
balanced2	93.41±0.88	93.15±0.87	92.98±0.97•	92.45±1.13•
balanced3	93.32±0.85	92.88±0.87•	92.90±0.90	92.08±1.46•
balanced4	93.04±0.91	93.08±1.02	92.57±1.04•	92.33±1.28•
balanced5	93.29±0.84	93.18±0.85	93.08±0.87	92.79±1.06

○, • statistically significant improvement or degradation from J48

Table 5.4: Precision

Data Set	J48	JRip	PART	Ridor	
full1	0.91±0.04	0.89±0.05	0.88±0.06	0.81±0.07	•
full2	0.92±0.04	0.92±0.05	0.90±0.06	0.83±0.06	•
full3	0.89±0.05	0.88±0.05	0.88±0.07	0.81±0.07	•
full4	0.90±0.05	0.88±0.05	0.87±0.06	0.80±0.08	•
full5	0.89±0.04	0.85±0.04•	0.86±0.07	0.80±0.07	•
reduced1	0.93±0.05	0.89±0.05•	0.93±0.04	0.86±0.08	•
reduced2	0.93±0.04	0.92±0.04	0.93±0.04	0.82±0.08	•
reduced3	0.88±0.05	0.90±0.05	0.92±0.05○	0.85±0.07	
reduced4	0.89±0.05	0.89±0.05	0.91±0.04	0.83±0.08	•
reduced5	0.87±0.04	0.87±0.04	0.87±0.05	0.81±0.06	•
balanced1	0.94±0.01	0.93±0.01	0.93±0.02	0.89±0.03	•
balanced2	0.94±0.01	0.93±0.01•	0.93±0.02	0.90±0.02	•
balanced3	0.93±0.01	0.93±0.01	0.93±0.02	0.89±0.03	•
balanced4	0.92±0.01	0.93±0.02	0.92±0.03	0.89±0.02	•
balanced5	0.93±0.01	0.94±0.01○	0.93±0.02	0.90±0.02	•

○, • statistically significant improvement or degradation from J48

Table 5.5: Recall

Data Set	J48	JRip	PART	Ridor	
full1	0.78±0.05	0.78±0.06	0.78±0.07	0.82±0.06	○
full2	0.77±0.05	0.76±0.06	0.78±0.07	0.81±0.06	
full3	0.74±0.06	0.78±0.06○	0.76±0.07	0.83±0.06	○
full4	0.75±0.07	0.76±0.07	0.75±0.09	0.80±0.07	○
full5	0.77±0.06	0.78±0.07	0.79±0.07	0.82±0.07	○
reduced1	0.71±0.06	0.75±0.06○	0.72±0.05	0.76±0.07	○
reduced2	0.72±0.06	0.73±0.06	0.72±0.06	0.79±0.07	○
reduced3	0.76±0.06	0.75±0.06	0.73±0.06	0.77±0.06	
reduced4	0.73±0.06	0.73±0.06	0.73±0.06	0.76±0.06	
reduced5	0.76±0.06	0.77±0.05	0.77±0.06	0.78±0.07	
balanced1	0.93±0.01	0.93±0.01	0.93±0.02	0.96±0.02	○
balanced2	0.93±0.01	0.93±0.01	0.93±0.02	0.96±0.02	○
balanced3	0.94±0.01	0.93±0.02●	0.93±0.02	0.96±0.02	○
balanced4	0.94±0.01	0.93±0.02	0.94±0.03	0.96±0.02	○
balanced5	0.93±0.01	0.92±0.02●	0.93±0.02	0.96±0.01	○

○, ● statistically significant improvement or degradation from J48

Table 5.6: Number of Rules

Data Set	J48	JRip	PART	Ridor
full1	38.71±4.99	8.75±1.65○	32.91±5.83○	19.73±3.14○
full2	30.40±5.55	9.27±2.09○	34.83±6.23	20.19±3.33○
full3	39.50±7.03	8.78±1.69○	33.05±6.51	19.73±3.20○
full4	33.37±7.94	8.36±1.72○	35.46±5.80	19.75±3.11○
full5	39.02±6.16	9.09±2.00○	33.78±7.06	20.18±3.14○
reduced1	16.29±2.09	6.68±1.26○	12.67±2.25○	19.98±4.08●
reduced2	18.53±2.86	5.31±1.04○	13.50±2.65○	22.84±4.18●
reduced3	14.20±2.13	4.99±0.90○	11.86±2.11○	21.61±4.12●
reduced4	14.44±2.83	5.60±1.13○	14.35±2.11	22.11±4.07●
reduced5	20.92±4.99	6.29±1.28○	14.42±2.20○	22.05±4.18
balanced1	35.51±4.37	10.36±1.84○	18.80±2.98○	24.63±4.86○
balanced2	44.30±8.40	10.14±2.60○	17.63±2.33○	26.16±4.77○
balanced3	56.98±7.02	10.50±1.77○	18.55±2.34○	27.60±5.97○
balanced4	42.96±7.15	12.16±2.42○	19.64±2.35○	26.79±5.41○
balanced5	45.66±8.42	10.25±1.80○	19.85±3.09○	29.27±6.21○

○, ● statistically significant improvement or degradation from J48

5.5 Using the knowledge

The rules that were learned were then used in place of the greedy and exhaustive search algorithms. Like with the IGS algorithm, only ten iterations were allowed because convergence to an equilibrium is not guaranteed. The initial results were mixed. For small values of α the topologies formed were comparable to the greedy search algorithm. Figure 5.1 shows an example topology formed using the rules. Unfortunately, as α got larger, the graphs that were formed were disconnected, as see in Figure 5.2.

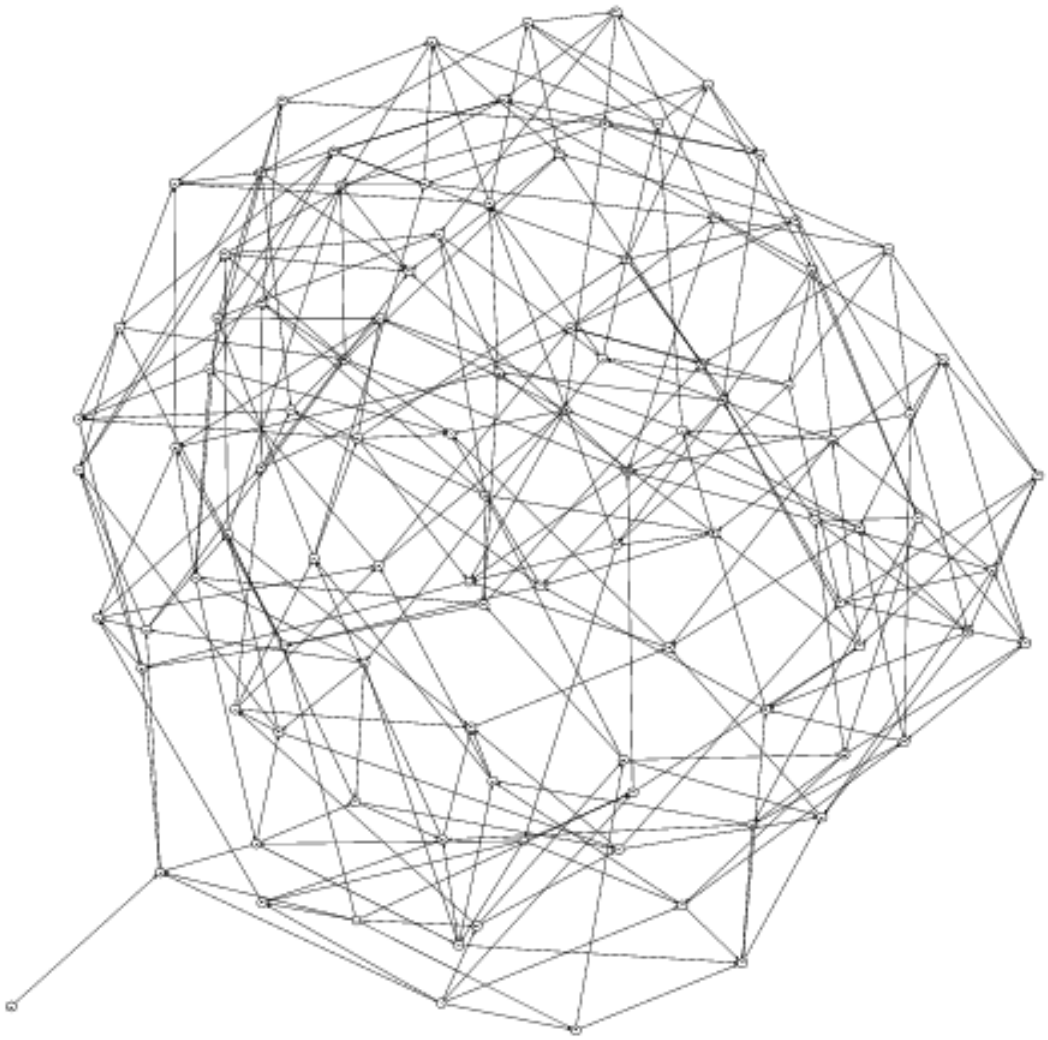


Figure 5.1: Example topology formed using rules, $\alpha = 0.5$, $maxDegree = 6$.

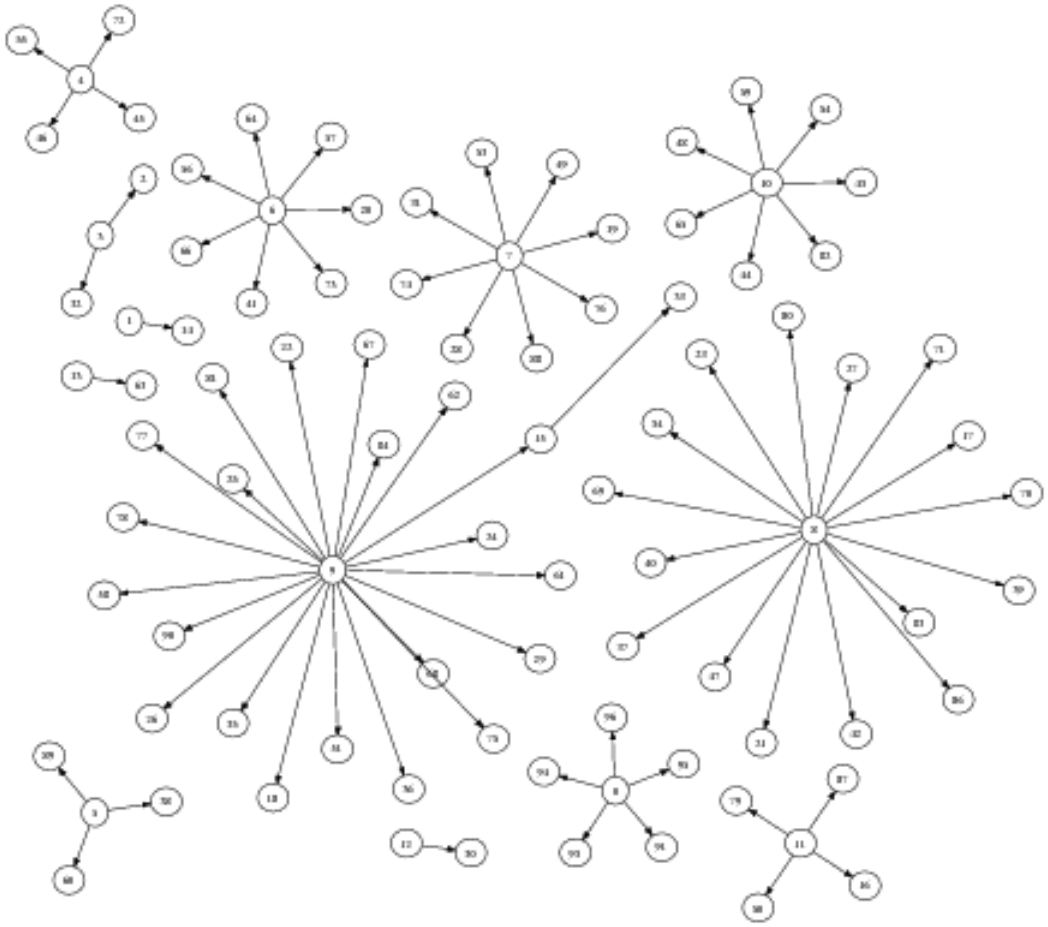


Figure 5.2: Network topology formed using rules, $\alpha = 200$.

To remedy this problem, a simple, greedy connection algorithm was implemented. After a node had selected its strategy according the rules, if the network was disconnected it would attempt to connect to nodes with high degree in other subgraphs so far as the maximum node degree constraints were not violated. For most cases, this approach works well. However, when a $maxDegree$-regular subgraph is formed there is no way for the network to become connected, as seen in Figure 5.3, because the greedy connection heuristic will not remove links. This scenerio is not common though, and only happens when the $maxDegree$ is relatively small. Discussion of the results obtained using the rules with greedy connection are given in Chapter 6.

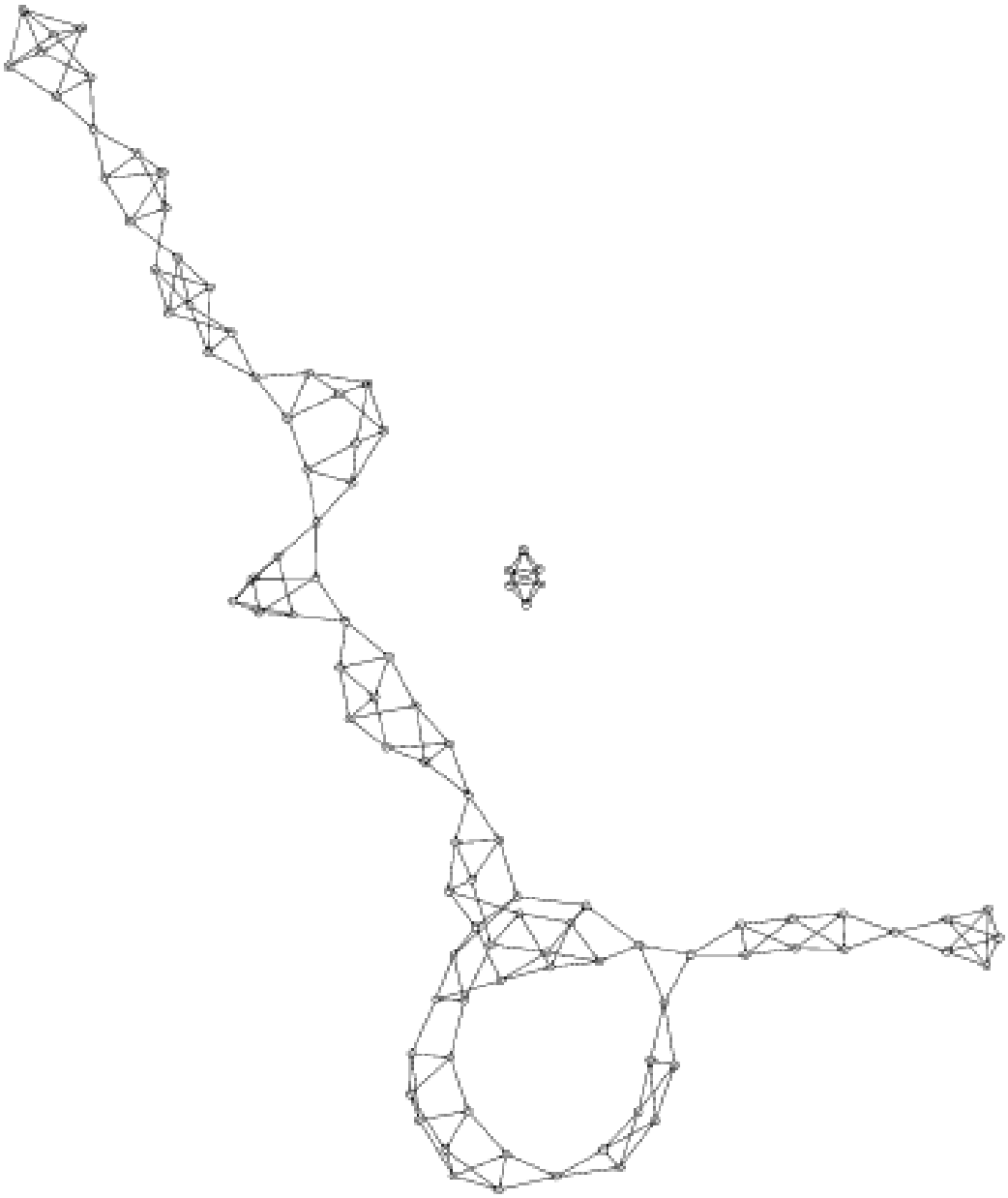


Figure 5.3: Topology formed using rules, $\alpha = 0.5$, $maxDegree = 4$. The greedy connection approach can not resolve problems when a $maxDegree$-regular subgraph is formed. Since the greedy connection approach will not violate the maximum node degree constraints nor will it remove any nodes, the resulting graph is disconnected. Fortunately, this is only a problem when $maxDegree$ is relatively small and α is low.

Chapter 6

Topologies formed by Rules

In this chapter I compare the topologies that are formed using the rule-based approach with a greedy connection strategy. Results are given for 100-node networks. Comparisons with exhaustive and greedy search for 20-node networks are shown in Section 4.2.2. Several different topologies are also given for varying values of α. Appendix B also provides topologies formed by rules in 20-node networks. Results for popular and greedy traffic-demand distributions are also provided in Appendix B All results presented for the rules approach with greedy connection strategy are mean values of ten separate simulations. Greedy search results and rules without the greedy connect approach are only given for one simulation due to computational complexity.

6.1 100 node networks

The number of edges, graph transport cost, graph diameter, characteristic path length, and spectral radius data is given in Table 6.1, Table 6.2, Table 6.3, Table 6.4, and Table 6.5, respectively. As can be seen, the topologies formed with rules approach are similar to those formed by the greedy approach. When $\alpha \leq 1$ and the maximum node degree is unconstrained, the topologies are densely connected graphs, like those formed with greedy search.

The node degree distributions, in Figures 6.1, 6.2, 6.3, and 6.4, provide additional insight. When $\alpha = 0.5$ the rules approach results in more edges being created, but the shape of the

distribution is similar to the greedy approach. When $\alpha = 60$, the greedy approach results in a star topology. The rules approach does not result in a strict star topology, but still has a graph center with high node degree. One result that is unexplained is the spike in nodes with degree of 55 in the rules approach for $\alpha = 1$.

However, there are some differences as α gets larger when the maximum node degree is unconstrained. For the topologies formed using rules, the center node tends to have a higher degree than topologies formed using greedy search, see Figure 6.5. This results in lower graph diameters and characteristic path lengths. This also results in a larger spectral radius. However, when the maximum node degree is constrained, similar topologies to the greedy search result, as in Figure 6.6.

The situation gets worse when the maximum node degree is constrained and alpha is low. Figure 6.7 shows the topology formed when $\alpha = 5$ and $maxDegree = 10$. It is a reasonable topology for those parameters. However, as the maximum node degree constraint is tightened, the problems increase, as in Figure 6.8. When α is further reduced to 1, the topology has an incredibly high diameter for the number of nodes, Figure 6.9.

At this point I have little explanation for this result, except I hypothesize that it has to do the greedy connection heuristic as the initial rules results showed favorable topologies for small values of α even when the maximum node degree was constrained, as in Figure 5.1. Figure 6.10 provides a comparison of the graph diameter to the maximum node degree when $\alpha = 1$. For reference, the graph diameter using greedy search is 5 when $maxDegree = 6$ and 3 when $maxDegree = 10$.

Another shortcoming is that the popular and greedy nodes seem to have little effect on the topologies formed when α is large. This is not surprising as the rules approach struggled to distinguish popular and greedy nodes for smaller networks as well. The reasons for this are currently unclear.

Table 6.1: Number of edges in graph

α	greedy	rules, first attempt	rules+greedy-connect
0.5	3399	4000	3973.6
1	2438	2910	2897.6
5	99	62	101.4
60	99	81	99.0
200	99	81	99.0
400	99	81	99.0
1000	99	80	99.0

Table 6.2: Graph transport cost

α	greedy	rules, first attempt	rules+greedy connect
0.5	11089	10718	10734
1	12559	11940	11967
5	21205		25486
60	21205		28431
200	39143		28453
400	46558		28410
1000	46558		28278

Table 6.3: Graph diameter

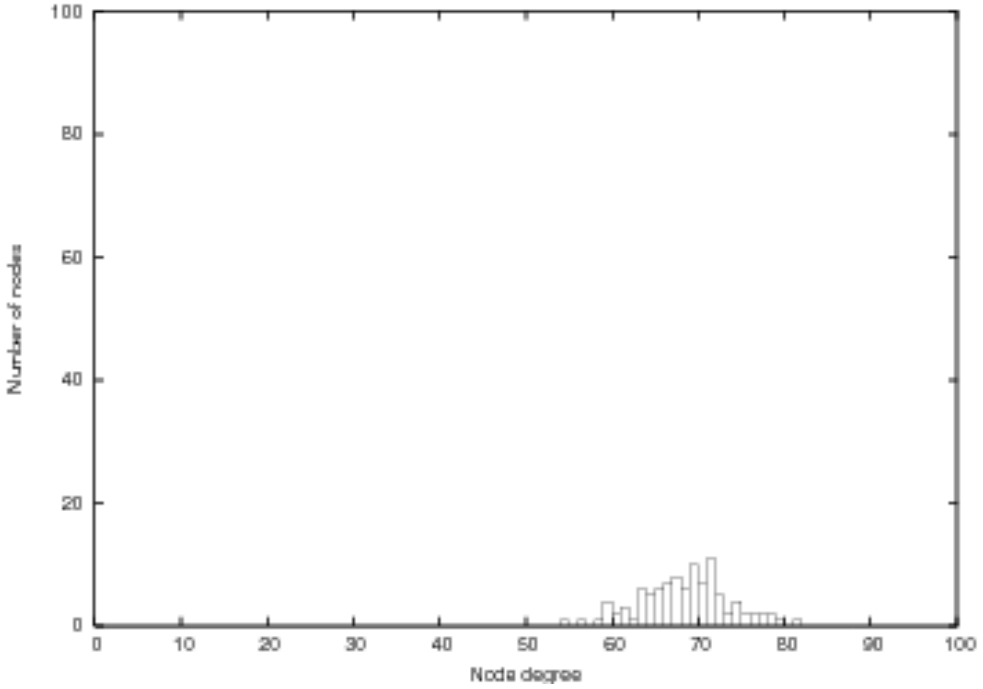
α	greedy	rules, first attempt	rules+greedy connect
0.5	2	2	2.0
1	2	2	2.0
5	2		4.6
60	2		5.2
200	7		5.2
400	9		5.2
1000	9		5.0

Table 6.4: Graph characteristic path length

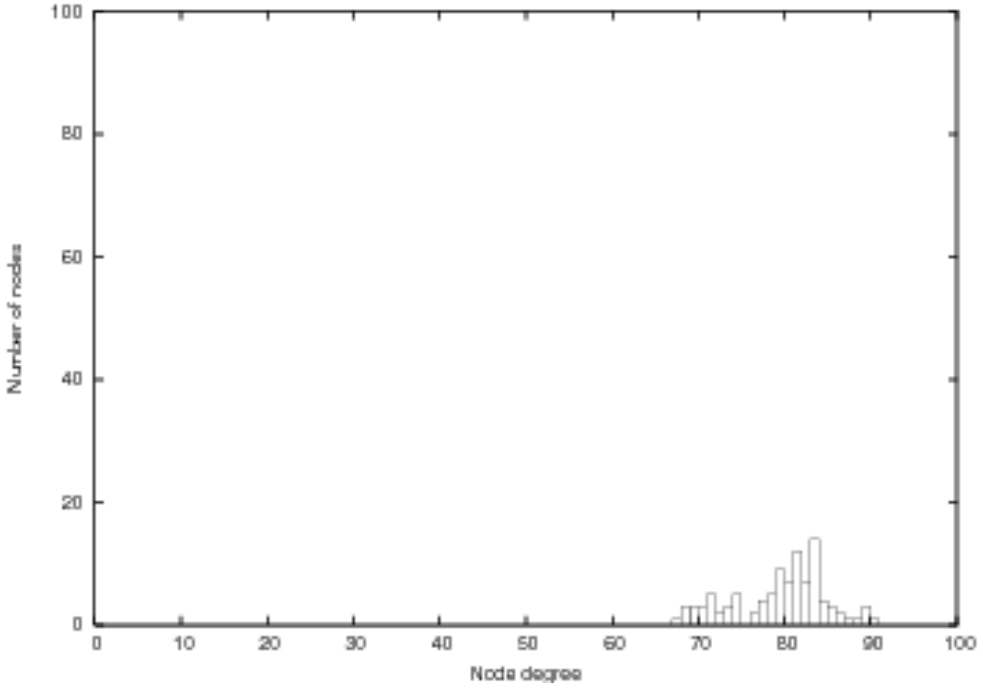
α	greedy	rules, first attempt	rules+greedy connect
0.5	1.31	1.19	1.19
1	1.50	1.41	1.41
5	1.98		2.39
60	1.98		2.66
200	3.67		2.66
400	4.35		2.65
1000	4.35		2.64

Table 6.5: Graph spectral radius

α	greedy	rules, first attempt	rules+greedy connect
0.5	68.36	80.21	79.76
1	49.27	59.12	58.76
5	9.94		8.89
60	9.94		8.08
200	6.52		8.07
400	5.46		8.08
1000	5.46		8.09

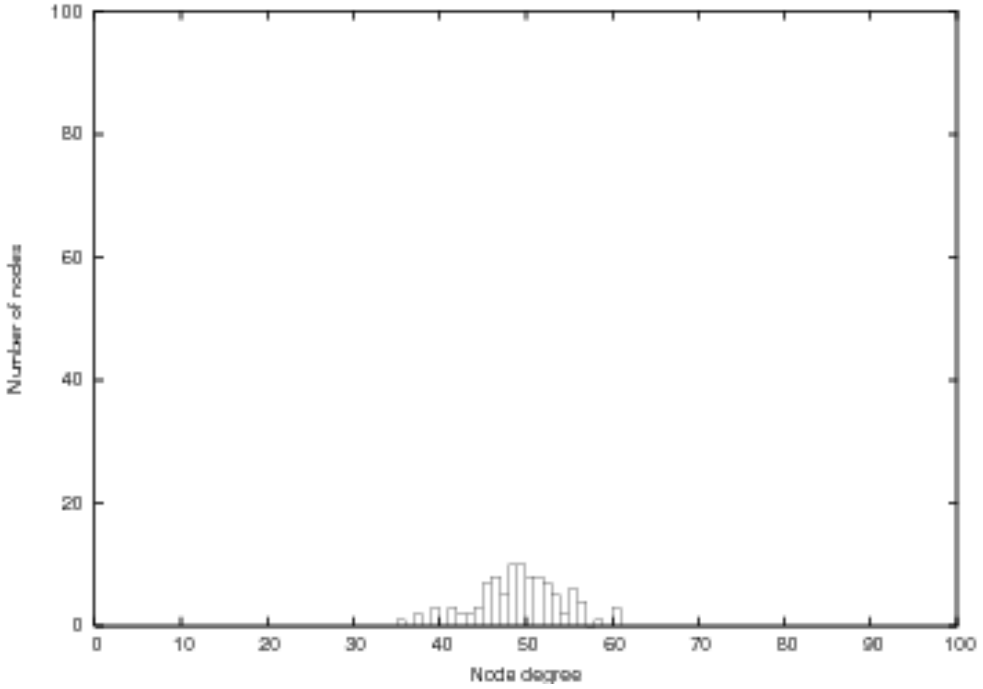


(a) Greedy search

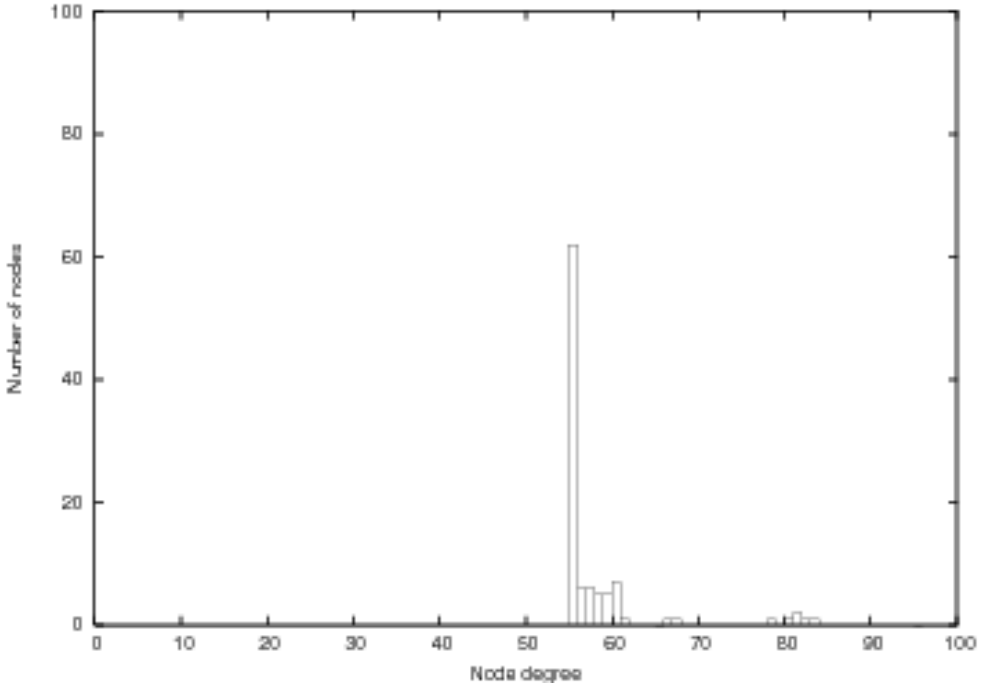


(b) Rules + Greedy Connect

Figure 6.1: Node degree distribution, $\alpha = 0.5$, $N = 100$, IGS versus Rules

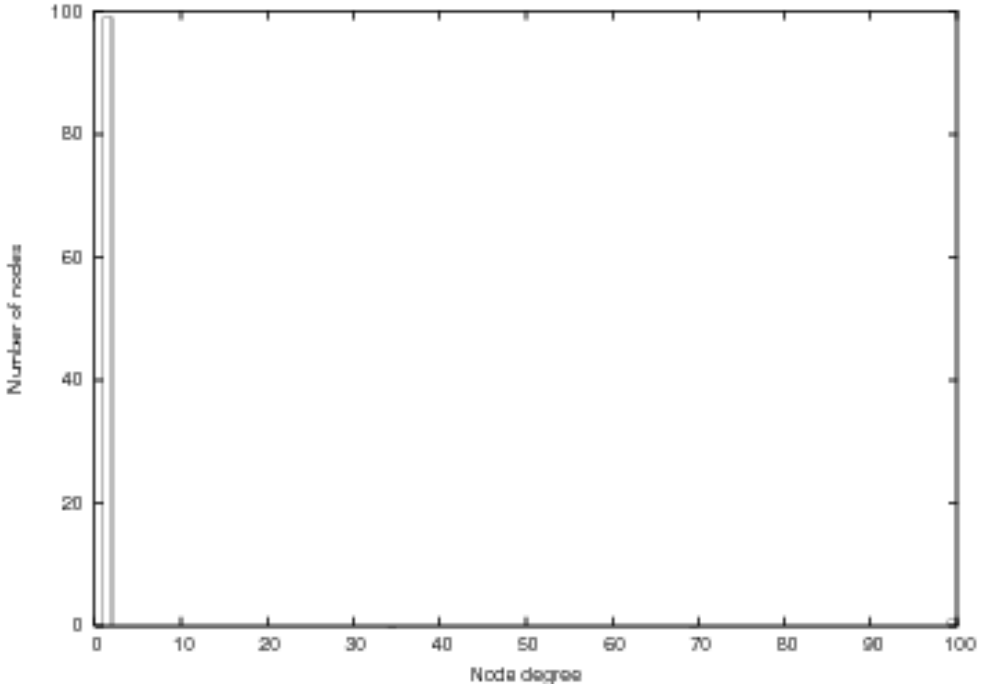


(a) Greedy search

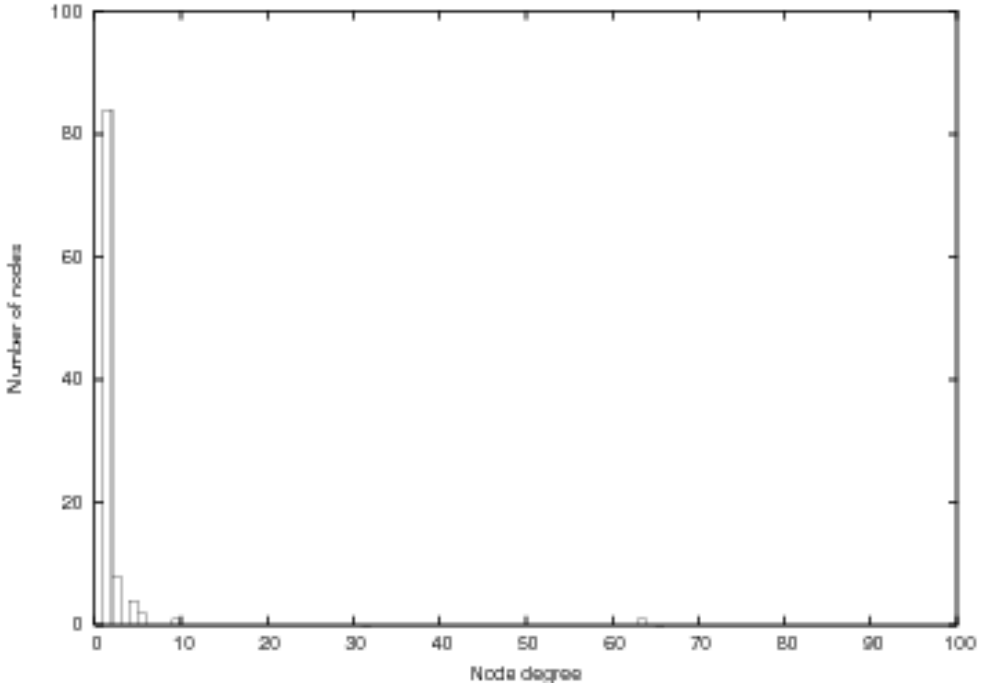


(b) Rules + Greedy Connect

Figure 6.2: Node degree distribution, $\alpha = 1$, $N = 100$, IGS versus Rules

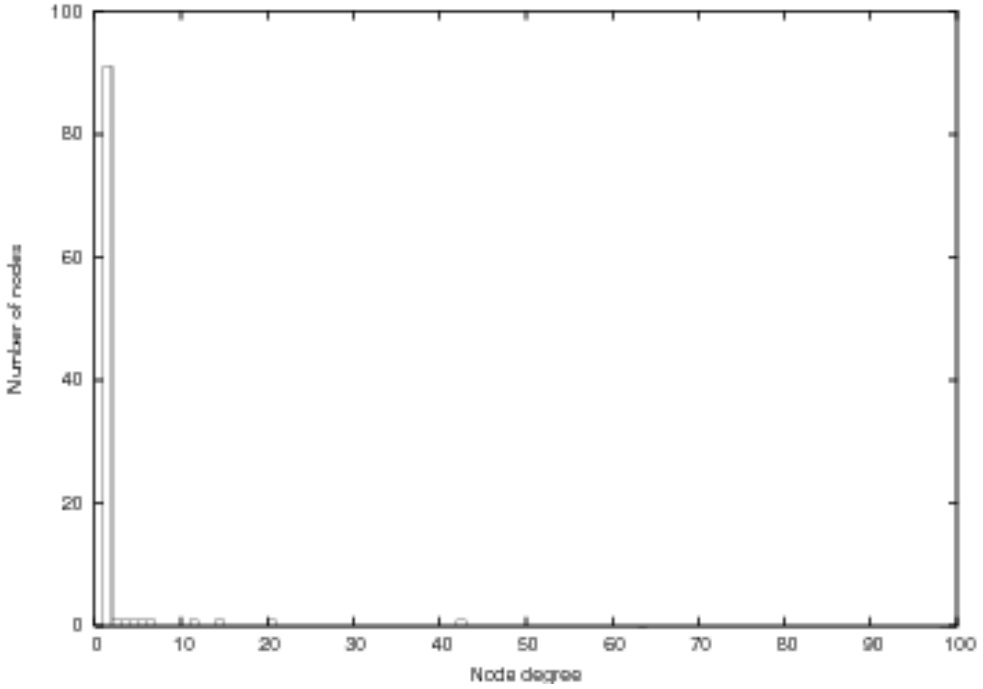


(a) Greedy search

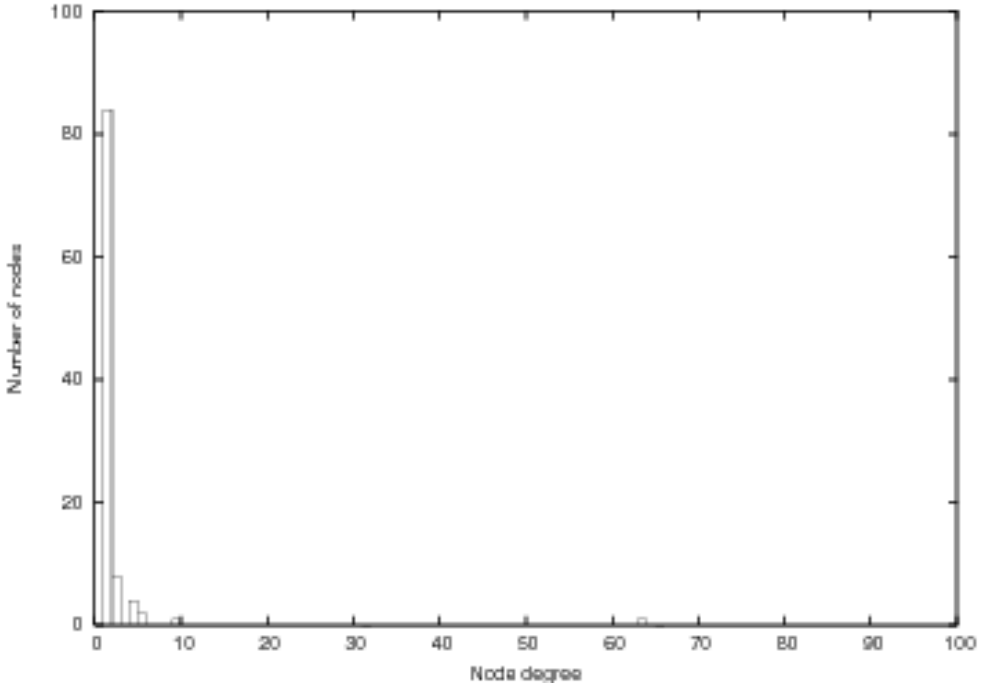


(b) Rules + Greedy Connect

Figure 6.3: Node degree distribution, $\alpha = 60$, $N = 100$, IGS versus Rules



(a) Greedy search



(b) Rules + Greedy Connect

Figure 6.4: Node degree distribution, $\alpha = 200$, $N = 100$, IGS versus Rules

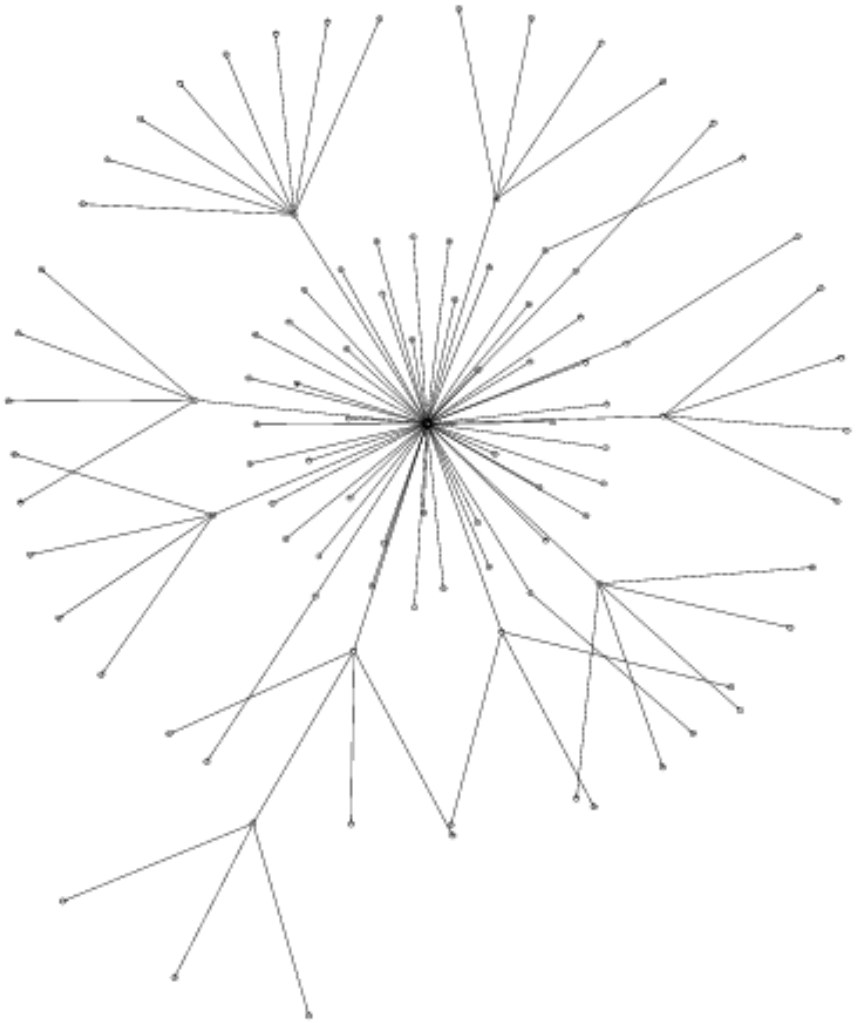


Figure 6.5: Representative topology formed with Rules + Greedy Connect, $\alpha = 1000$, Normal traffic-demand, and unconstrained maximum node degree. The tree is fairly well balanced with a center node having degree 63.

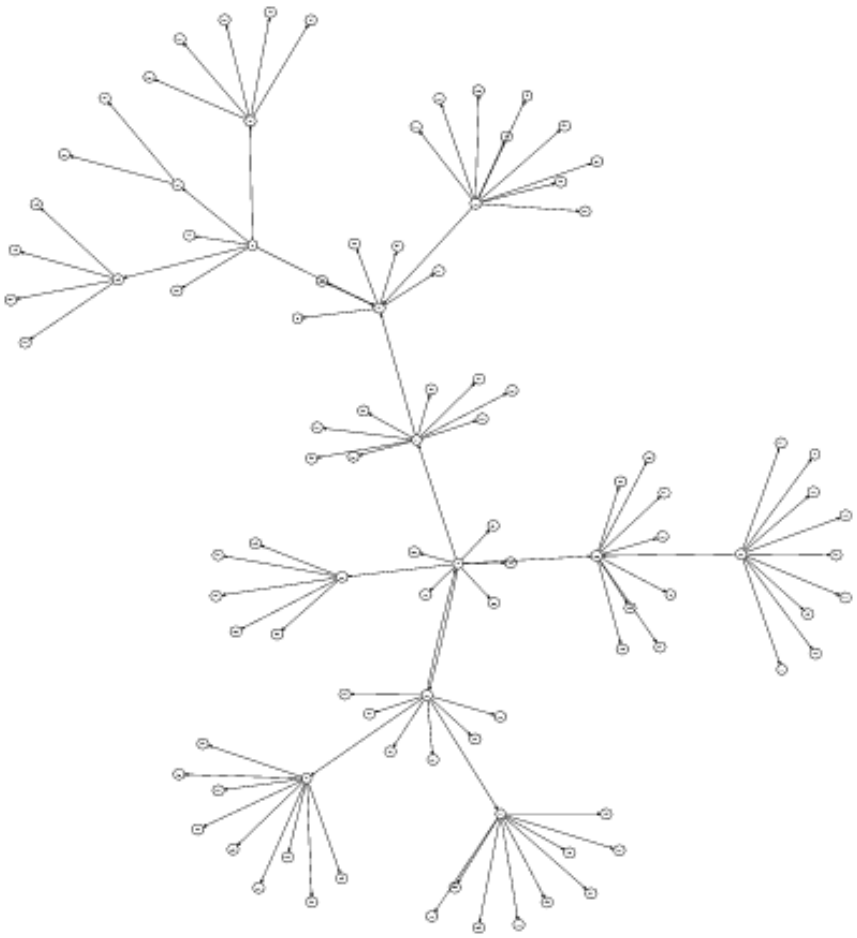


Figure 6.6: Representative topology formed with Rules + Greedy Connect, $\alpha = 1000$, Normal traffic-demand, and $maxDegree = 10$.

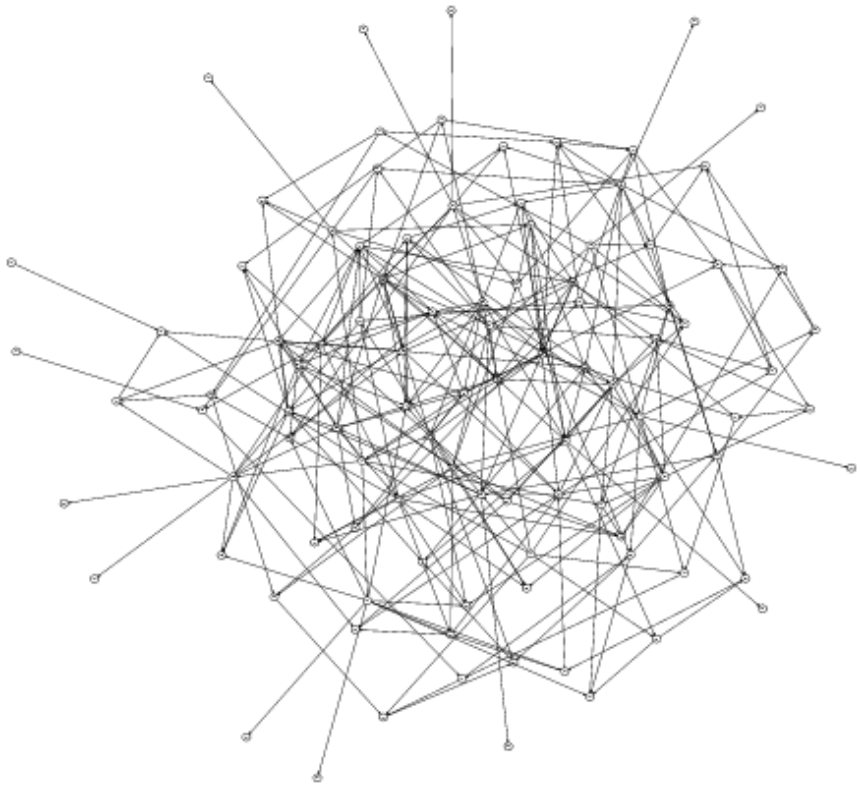


Figure 6.7: Representative topology formed with Rules + Greedy Connect, $\alpha = 5$, Normal traffic-demand, and $maxDegree = 10$.

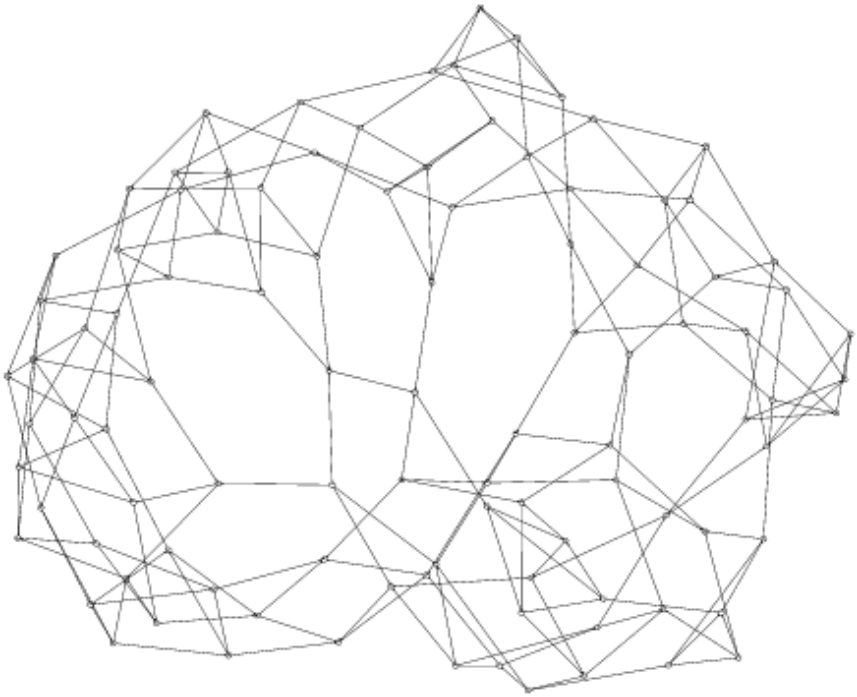


Figure 6.8: Representative topology formed with Rules + Greedy Connect, $\alpha = 5$, Normal traffic-demand, and $maxDegree = 4$. The diameter for this graph is 9.

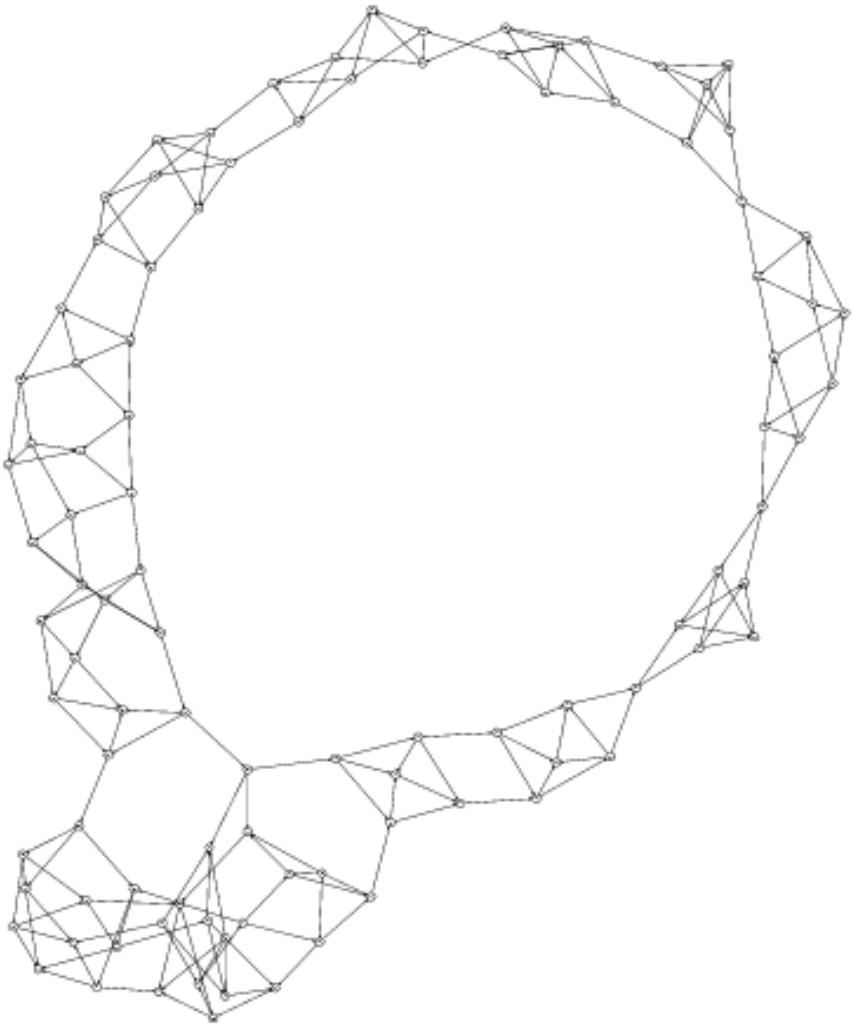


Figure 6.9: But this one is much worse! Representative topology formed with Rules + Greedy Connect, $\alpha = 1$, Normal traffic-demand, and $maxDegree = 4$. The diameter for this graph is 18!

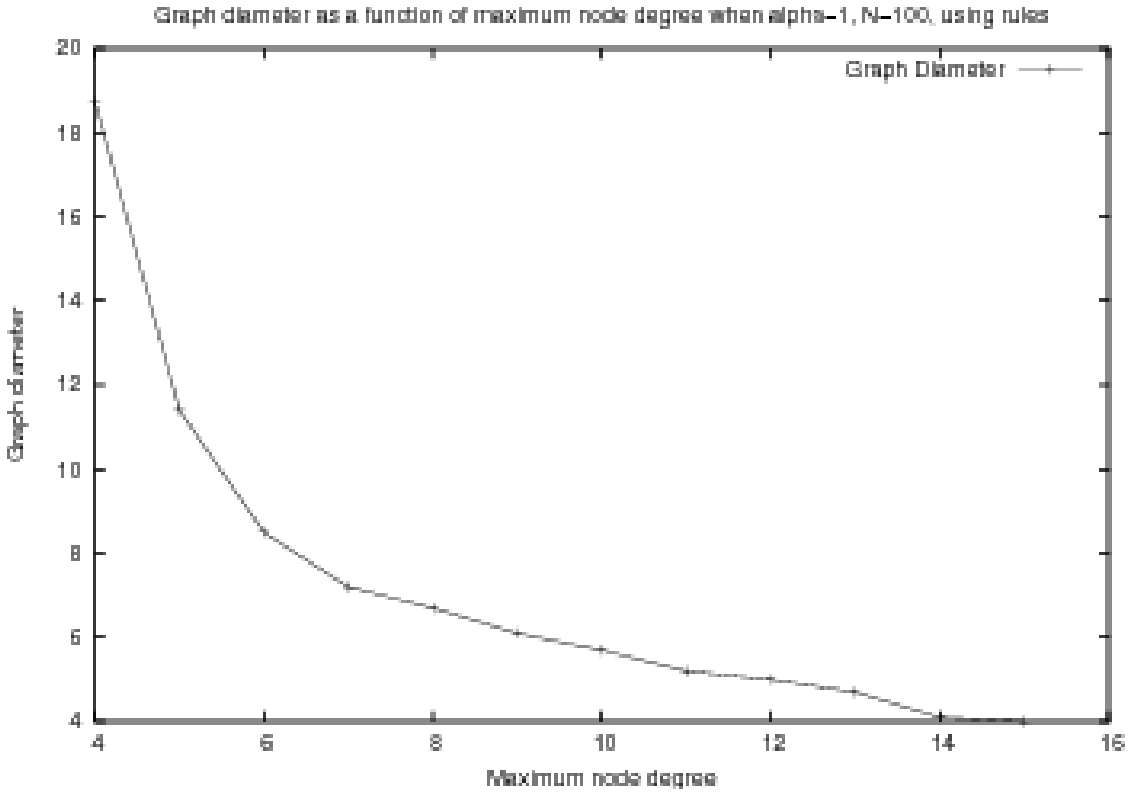


Figure 6.10: Graph diameter as a function of maximum node degree, $\alpha = 1$, $N = 100$, and Normal traffic-demand. For reference, the graph diameter using greedy search is 5 when $maxDegree = 6$ and 3 when $maxDegree = 10$.

Chapter 7

Conclusions and Future Work

In this thesis, I have introduced a game-theoretic approach to overlay network creation that considers the traffic-demand between nodes. Empirical results were also presented that show the effects of traffic-demand on the overlay network topologies formed. These results show that the traffic-demand between nodes is an important consideration when creating the overlay topology, as topologies are adapted to reflect the underlying traffic-demand distribution.

Unfortunately, using this approach to create overlay network topologies is intractable. Even a greedy hill-climbing approach is computationally expensive as the size of the network grows. To solve this problem, a machine learning approach was used to characterize the attributes of nodes that logical links were made toward. Using these attributes, a set of rules were learned that were used to decide whether to create a logical link toward a node or not. The resulting topologies were compared against those formed through the exhaustive and greedy hill-climbing approaches. These comparisons show that the rule-based approach creates similar topologies to the exhaustive and greedy hill-climbing approaches in most cases. In cases where popular and greedy nodes are present, the results are mixed. In some cases, the rules approach behaves in a similar manner to the exhaustive and greedy hill-climbing approach. But as the network size increases, the rules approach does not do as well in distinguishing the popular and greedy nodes. Another shortcoming of the rule-based approach is when the maximum node degree is highly constrained and α is small. In these

instances, the rule-based approach struggles to construct good networks. The reasons for this remain unclear and the subject of possible future work.

There are several areas of future work. The first is in the development of the rules. A dataset that is derived from the exhaustive search procedure, instead of the greedy hill-climbing approach would result in a more accurate rule set. Further refinement of the attributes used in learning through different node attributes and characteristics could lead to further improvements. A method to flag individual nodes as popular or greedy, instead of strictly relying on the traffic-demands, could also improve the ability to distinguish popular and greedy nodes. Further investigations on the scalability of the rule-based approach, especially with respect to the connection heuristic, is needed.

Another area of future work, would be considering a more realistic underlay topology, where the distance between nodes in the underlay are not constant. Other interesting areas of research, would be exploring the effects of heterogeneous nodes. What type of effect would nodes with different α and $maxDegree$ values have on the topology? How would the rules have to be changed to handle this? It would also be interesting to investigate this approach in a dynamic environment, where traffic-demand between nodes is constantly changing. How could online learning be used to adjust the rules in a dynamic environment? All of these issues would need to be investigated before a real-world implementation and deployment could be realized.

Bibliography

[1] H. Zhang, J. F. Kurose, and D. Towsley, "Can an overlay compensate for a careless underlay?" in *INFOCOM 2006: Twenty-fifth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2006.

[2] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of internet path selection," in *SIGCOMM '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication.* ACM Press, 1999.

[3] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker, "On selfish routing in internet-like environments," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, 2003.

[4] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *SOSP 2001: Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles*, 2001.

[5] A. Nakao, L. Peterson, and A. Bavier, "Scalable routing overlay networks," *SIGOPS Oper. Syst. Rev.*, vol. 40, no. 1, 2006.

[6] A. Rowstron and P. Druschel, *Lecture Notes in Computer Science.* Springer-Verlang, 2001, vol. 2218, ch. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems.

[7] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, January 2004.

[8] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, 2001.

[9] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, 2001.

[10] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica, "The impact of dht routing geometry on resilience and proximity," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, 2003.

[11] D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: a scalable and dynamic emulation of the butterfly," in *PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing*, 2002.

[12] P. Maymounkov and D. Mazieres, "A peer-to-peer information system based on the XOR metric," in *Proceedings of the IPTPS 2002*, 2002.

[13] P. Mahadevan, D. Krioukov, M. Fomenkov, B. Huffaker, X. Dimitropoulos, and K. Claffy, "Lessons from three views of the Internet topology," CAIDA, Tech. Rep., 2005, tr-2005-02.

[14] "Gnutella," http://www.gnutella.com.

[15] D. Stutzbach, R. Rejaie, and S. Sen, "Characterizing unstructured overlay topologies in modern P2P file-sharing systems," in *Proceedings of Internet Measurement Conference 2005*, 2005.

[16] M. Ripeanu, A. Iamnitchi, and I. Foster, "Mapping the Gnutella network," *IEEE Internet Computing*, vol. 6, no. 1, 2002.

[17] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *SIGCOMM '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, 1999.

[18] J. Kleinberg, "The small-world phenomenon: an algorithm perspective," in *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, 2000.

[19] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *Measurement and Modeling of Computer Systems*, 2000.

[20] A. Young, J. Chen, Z. Ma, A. Krishnamurthy, L. Peterson, and R. Y. Wang, "Overlay mesh construction using interleaved spanning trees," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 2004.

[21] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. The MIT Press, 2001.

[22] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker, "On a network creation game," in *PODC 2003: Proceedings of the Twenty-second Annual Symposium on Principles of Distributed Computing*, 2003.

[23] B.-G. Chun, R. Fonseca, I. Stoica, and J. Kubiatowicz, "Characterizing selfishly constructed overlay routing networks," in *INFOCOM 2004: Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, 2004.

[24] T. Moscibroda, S. Schmid, and R. Wattenhofer, "On the topologies formed by selfish peers," in *Proceedings of the Twenty-fifth Annual ACM Symposium on Principles of Distributed Computing*, 2006.

[25] N. Christin and J. Chuang, "A cost-based analysis of overlay routing geometries," in *INFOCOM 2005: Twenty-fourth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2005.

[26] E. Anshelevich, A. Dasgupta, E. Tardos, and T. Wexler, "Near-optimal network design with selfish agents," in *Thirty-fifth Annual ACM Symposium on Theory of Computing*, 2003.

[27] S. Albers, S. Eilts, E. Even-Dar, Y. Mansour, and L. Roditty, "On Nash equilibria for a network creation game," in *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2006.

[28] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[29] "Python programming language," http://www.python.org.

[30] "Networkx: High productivity software for complex networks," https://networkx.lanl.gov/wiki.

[31] "Graphviz - graph visualization software," http://www.graphviz.org.

[32] T. M. Mitchell, *Machine Learning*. WCB/McGraw-Hill, 1997.

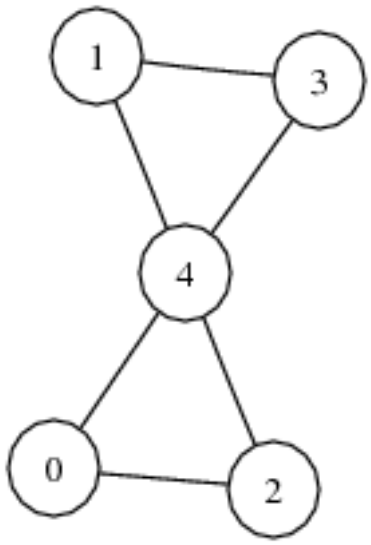
[33] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann Publishers, 2005.

Appendix A

MAPGEN file format

The MAPGEN file format is used for serializing adjacency matrices in a human-readable format. These appendix details the MAPGEN file format. An example MAPGEN file for the graph in Figure A.1 is presented in Figure A.2 for reference.

Figure A.1: Example graph



A.1 Comments

All lines that start with "|" are considered comments. Blank lines are also ignored.

Figure A.2: Example MAPGEN file

```
| This is a comment line.
| The first non-comment line should contain the number of nodes
5
* * 1 * 1
* * * 1 1
1 * * * 1
* 1 * * 1
1 1 1 1 *
```

A.2 Data

The first non-comment line signifies the start of the data section. The first non-comment line should contain a single number distinguishing the number of nodes in the graph.

Each remaining line gives the edge weights between nodes in a matrix-like format where edge weights are separated by whitespace. Entries between non-adjacent nodes are represented by an asterisk, "*".

Appendix B

Additional 20 Node Results

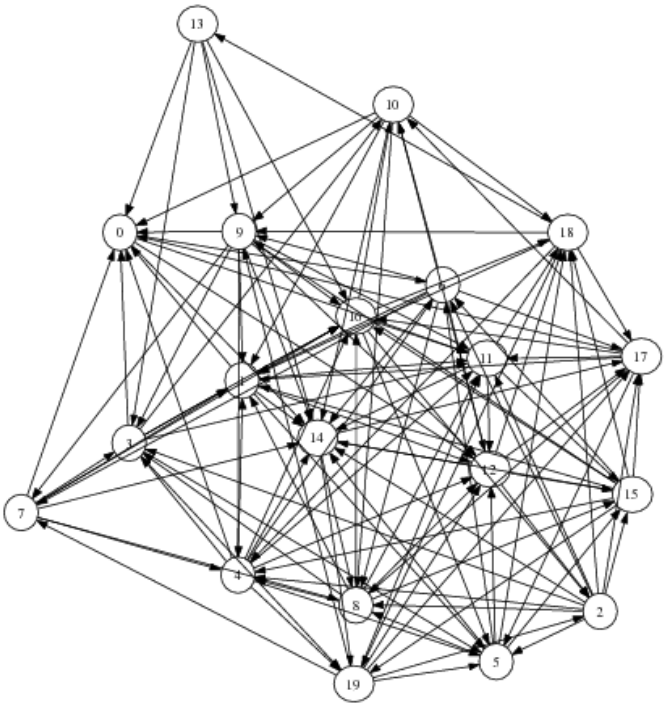
B.1 Unconstrained node degree

Table B.1: Degree of popular node for varying values of α, using Rules + Greedy Connect

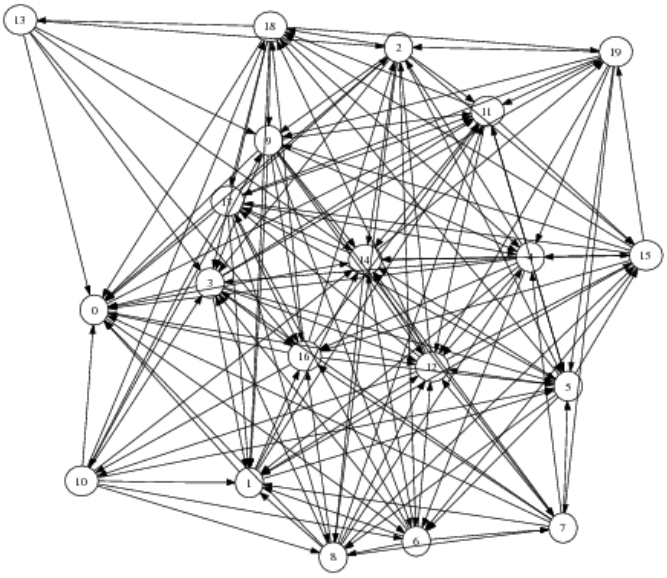
	Normal Traffic Demand		Node 4 is *popular*	
α	degree	in degree	degree	in degree
0.5	17.8	9.4	19.0	10.6
1	16.8	9.7	19.0	12.0
5	6.7	5.6	19.0	18.7
60	1.4	0.8	1.4	0.8

Table B.2: Degree of greedy node for varying values of α, using Rules + Greedy Connect

	Normal Traffic Demand		Node 4 is *greedy*	
α	degree	out degree	degree	out degree
0.5	17.8	8.4	19.0	9.8
1	16.8	7.1	19.0	9.8
5	6.7	1.1	19.0	13.0
60	1.4	0.6	1.4	0.6

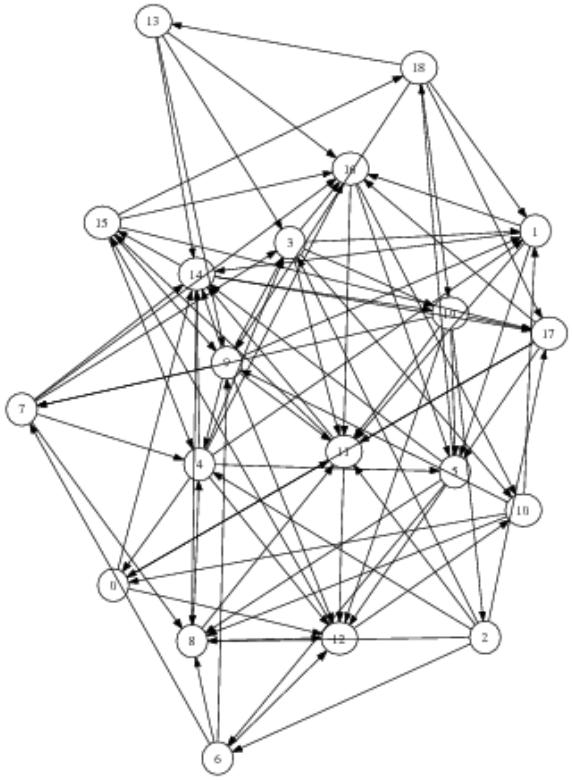


(a) Greedy search

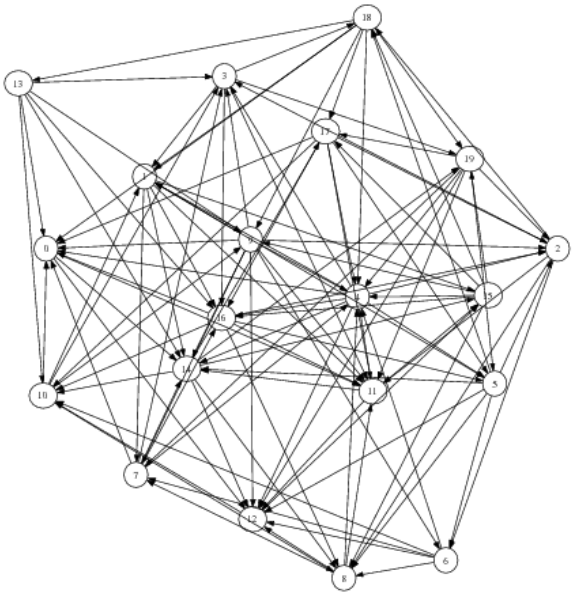


(b) Rules + Greedy Connect

Figure B.1: Topology comparison for $\alpha = 0.5$, Normal traffic-demand, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic.

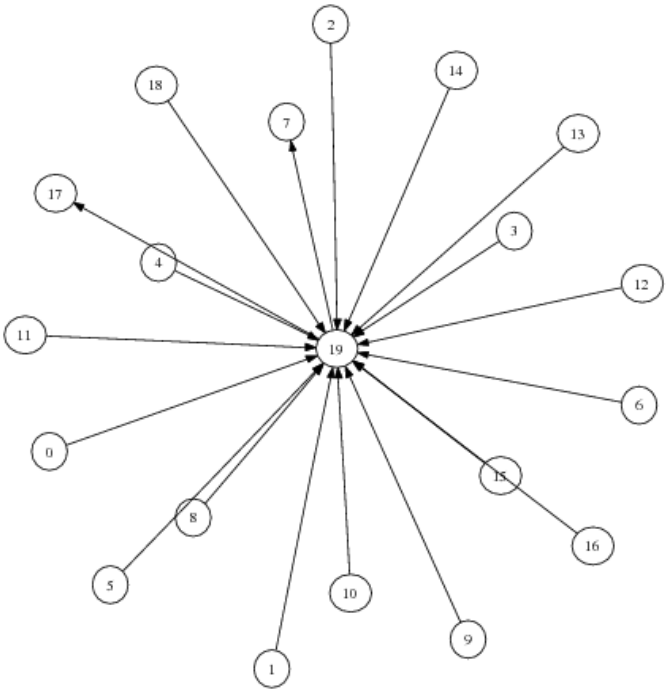


(a) Greedy search

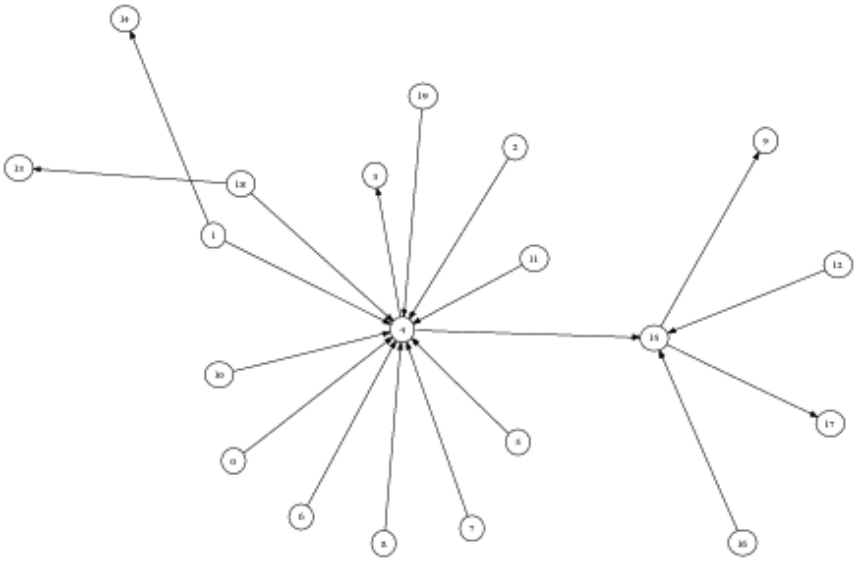


(b) Rules + Greedy Connect

Figure B.2: Topology comparison for $\alpha = 1$, Normal traffic-demand, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic.

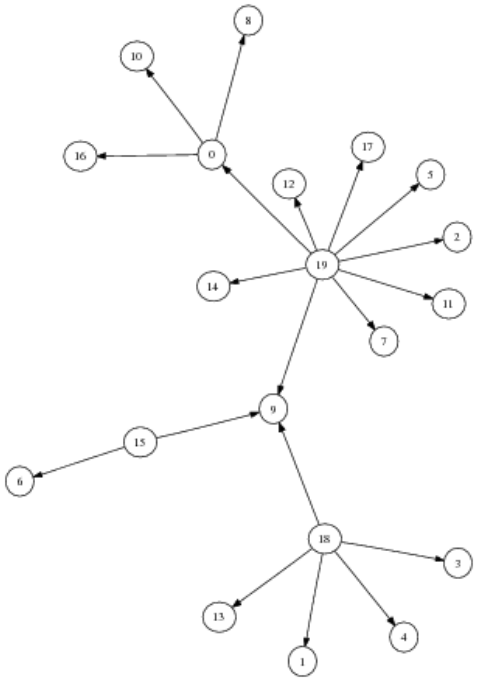


(a) Greedy search

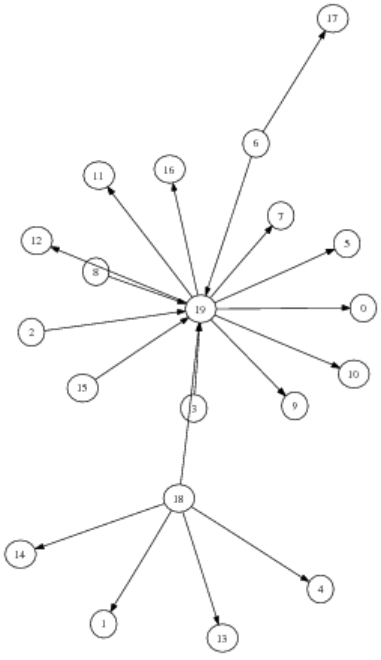


(b) Rules + Greedy Connect

Figure B.3: Topology comparison for $\alpha = 5$, Normal traffic-demand, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic. The topologies formed with rules tend to be more tree-like, with a center node of high degree.

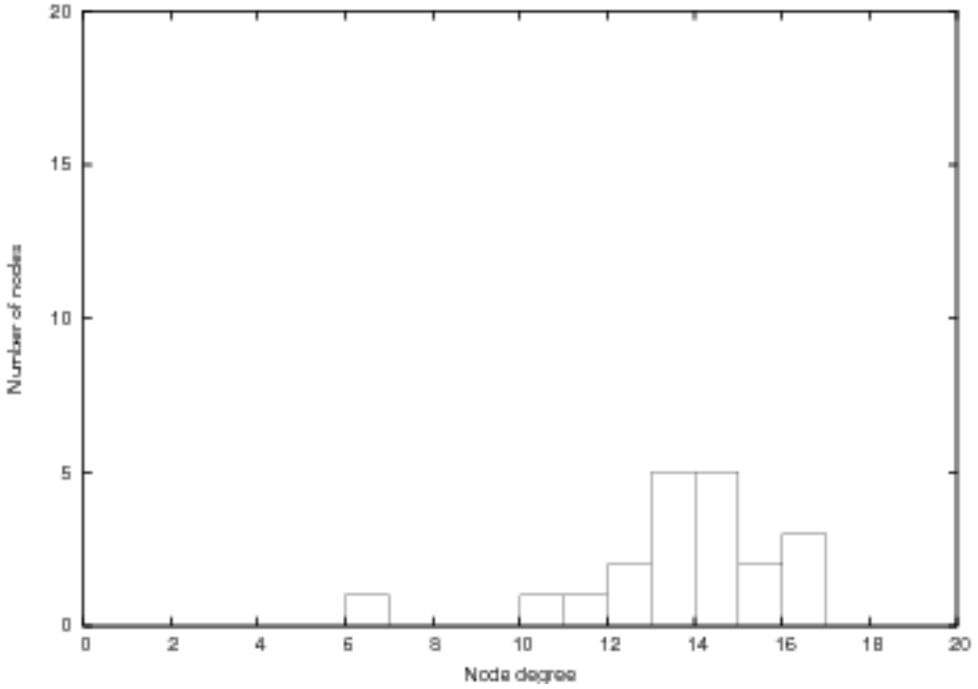


(a) Greedy search

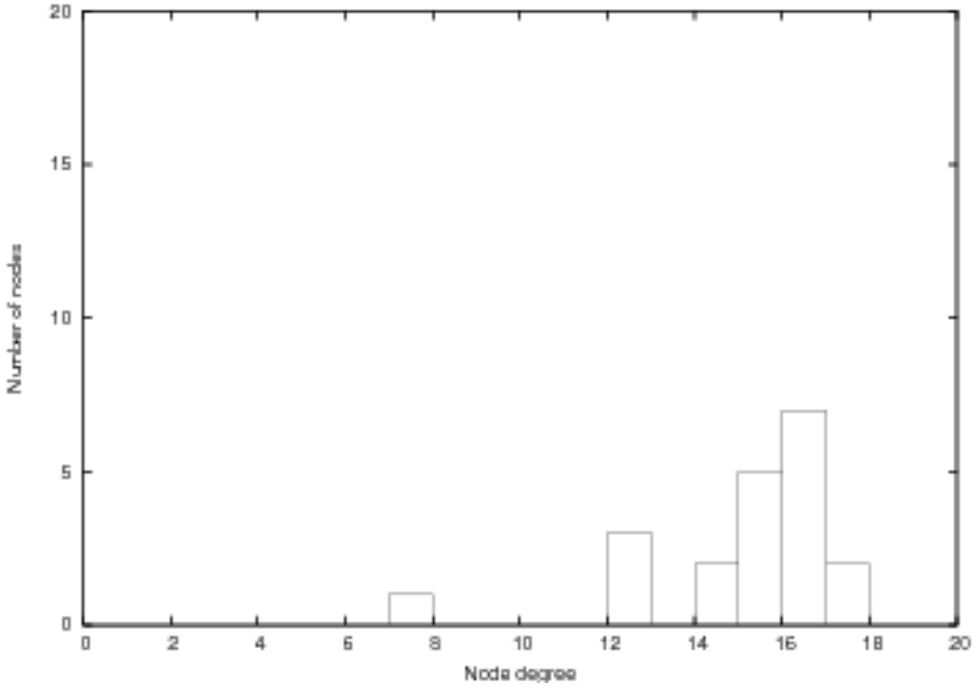


(b) Rules + Greedy Connect

Figure B.4: Topology comparison for $\alpha = 60$, Normal traffic-demand, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic. Topologies formed using the rules tend to have a higher spectral radius and smaller characteristic path length because of the center node having a higher node degree.

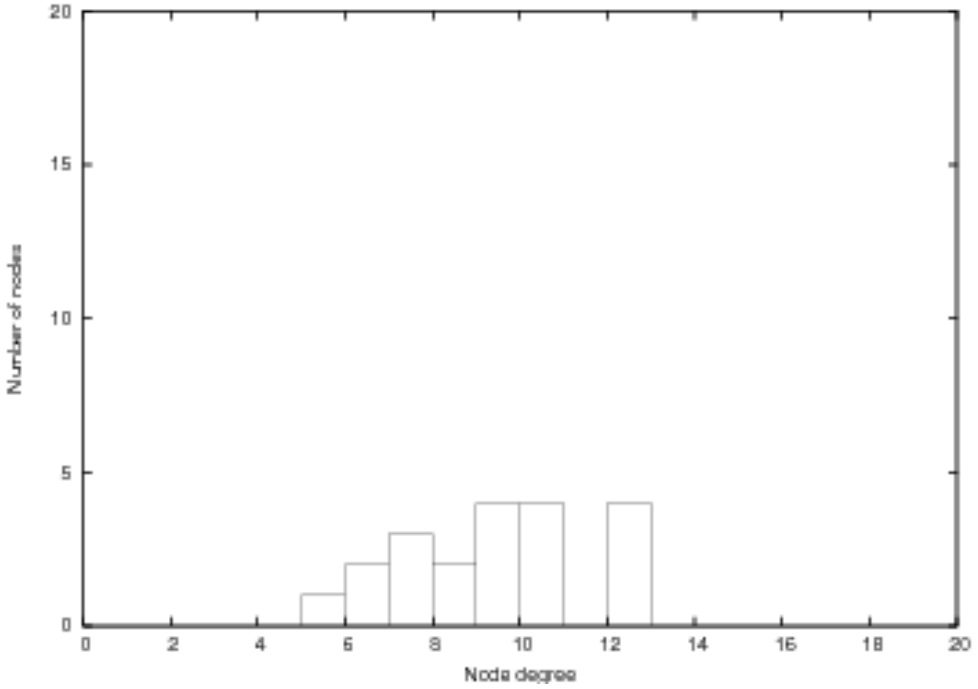


(a) Greedy search

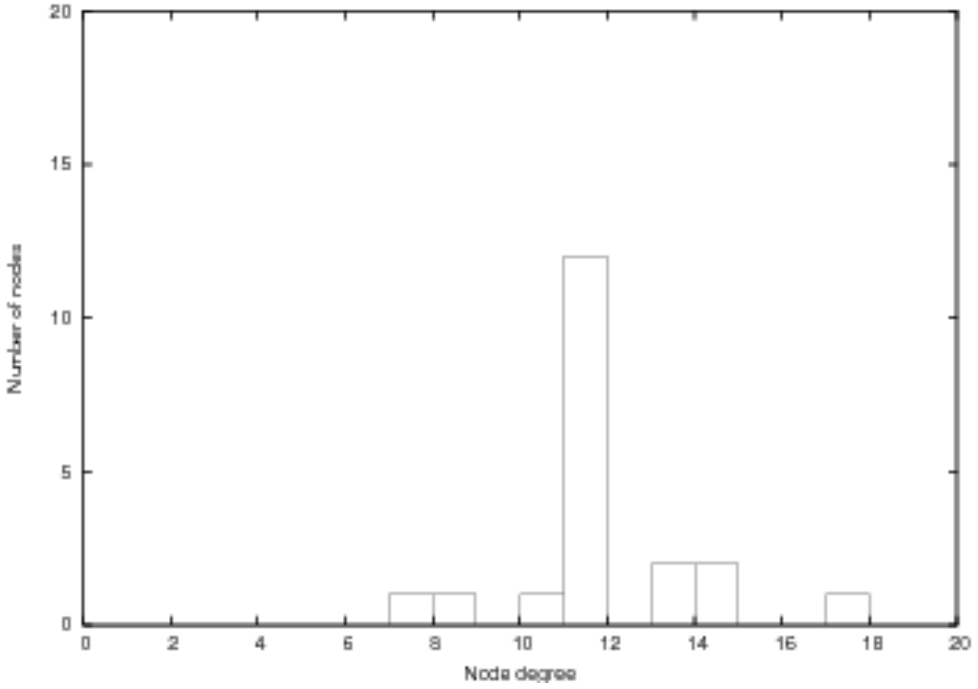


(b) Rules + Greedy Connect

Figure B.5: Node degree distribution, $\alpha = 0.5$, IGS versus Rules

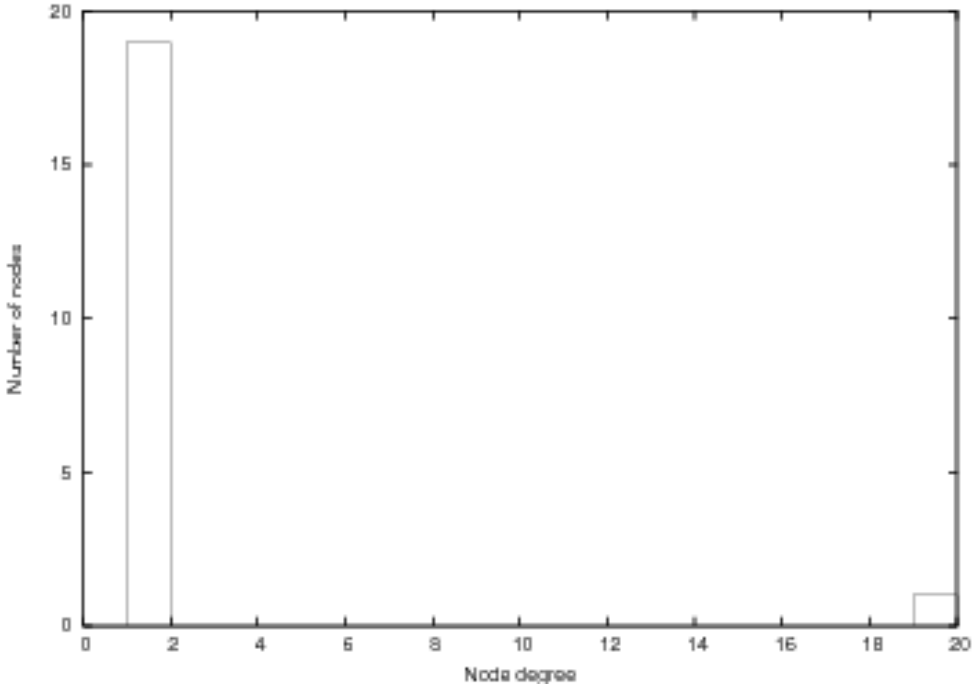


(a) Greedy search

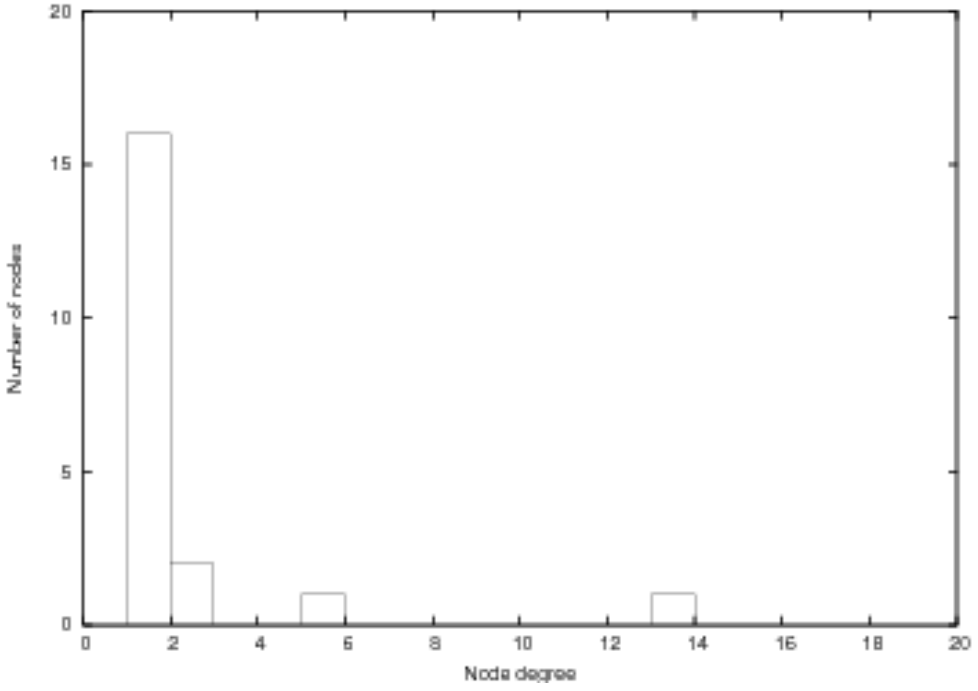


(b) Rules + Greedy Connect

Figure B.6: Node degree distribution, $\alpha = 1$, IGS versus Rules

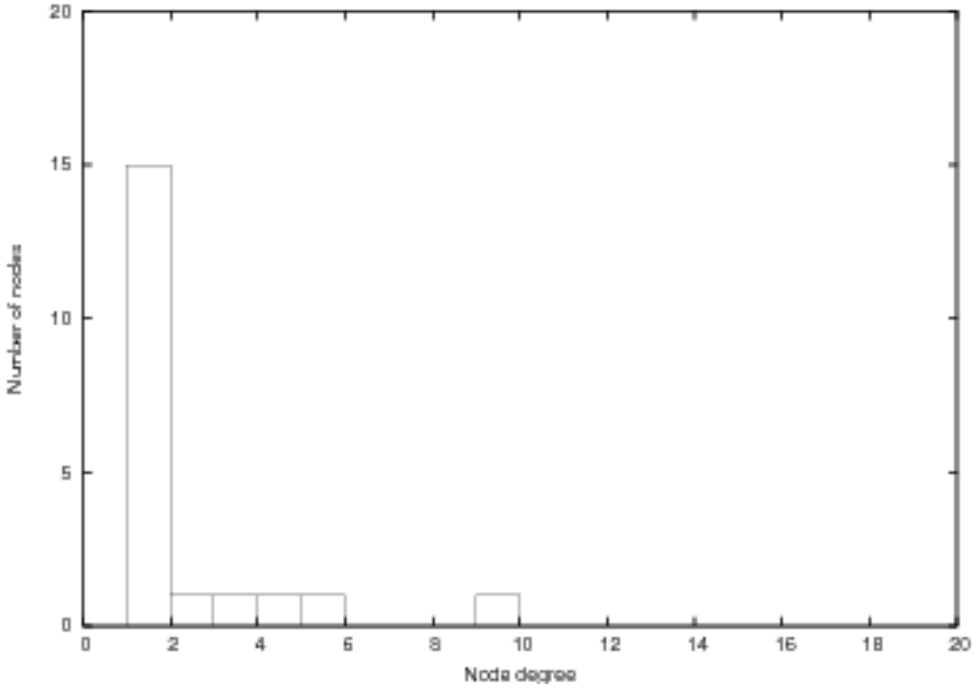


(a) Greedy search

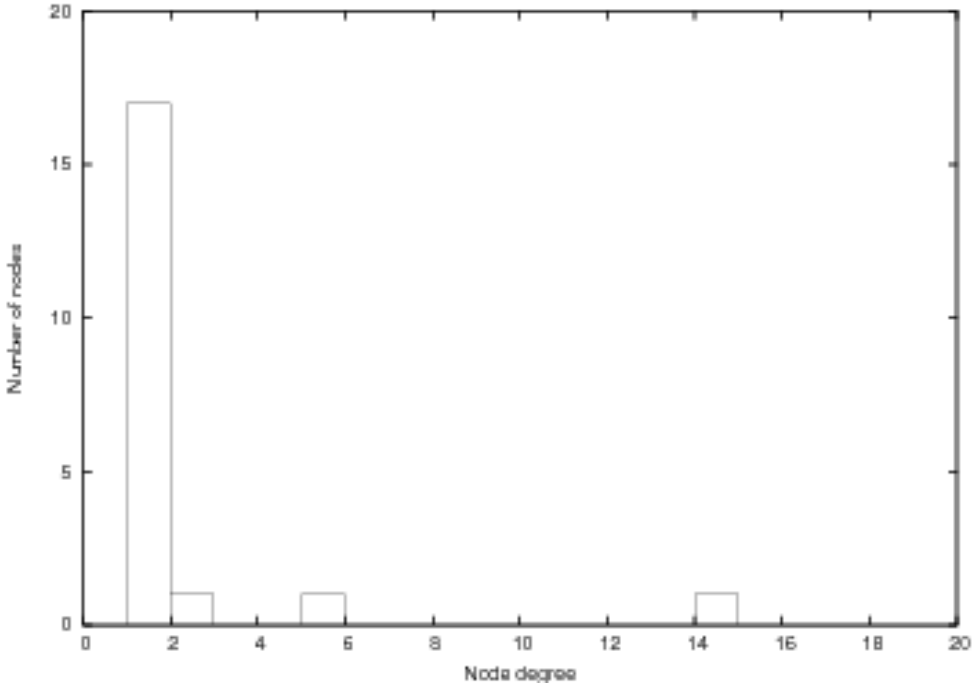


(b) Rules + Greedy Connect

Figure B.7: Node degree distribution, $\alpha = 5$, IGS versus Rules

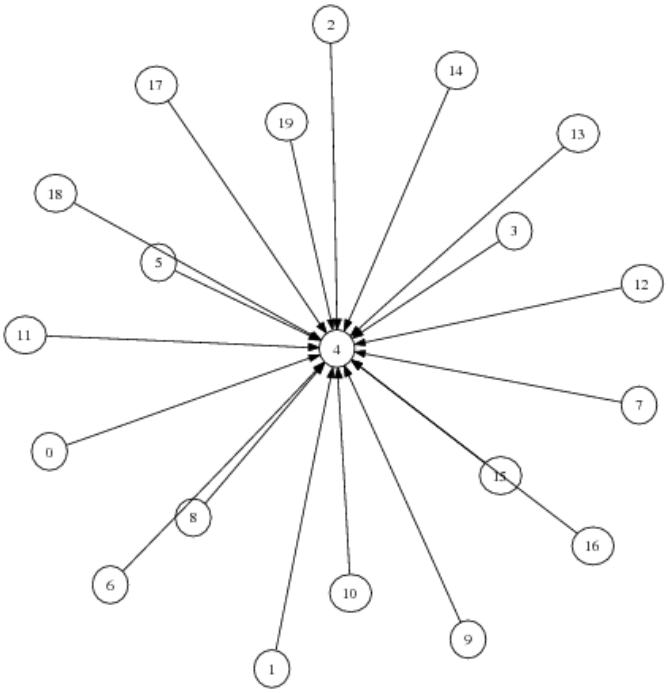


(a) Greedy search

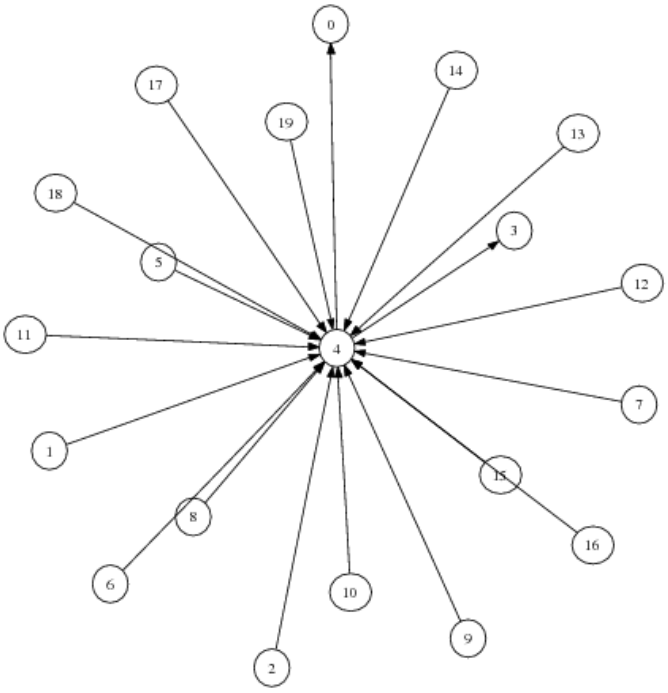


(b) Rules + Greedy Connect

Figure B.8: Node degree distribution, $\alpha = 60$, IGS versus Rules

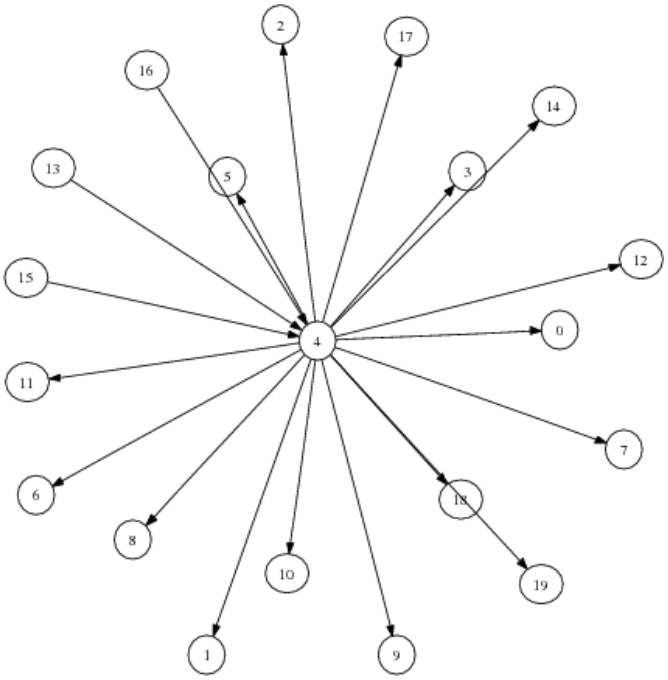


(a) Greedy

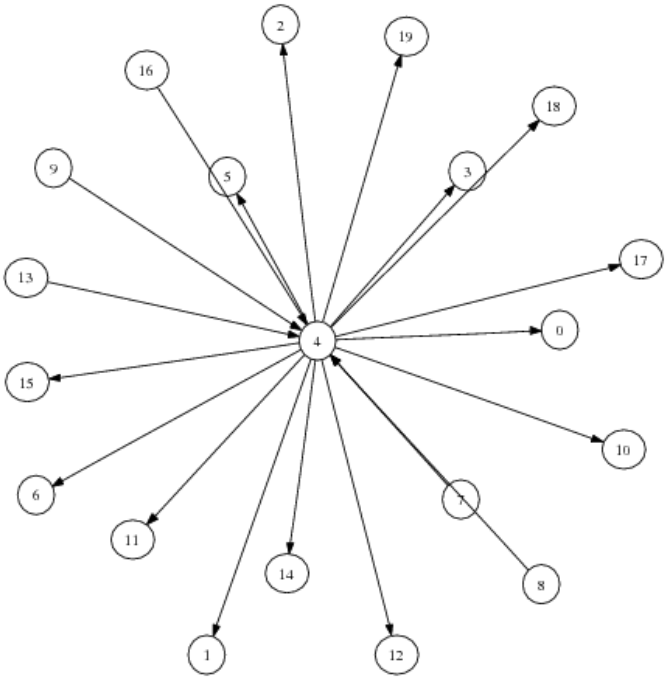


(b) Rules + Greedy Connect

Figure B.9: Topology comparison for $\alpha = 5$, with node 4 being popular, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic.

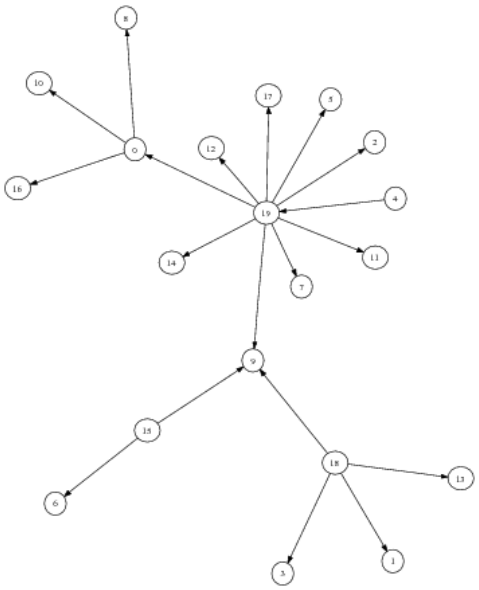


(a) Greedy

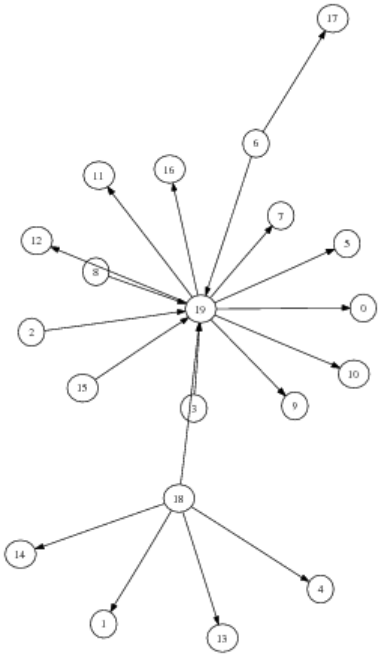


(b) Rules + Greedy Connect

Figure B.10: Topology comparison for $\alpha = 5$, with node 4 being greedy, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic.



(a) Greedy



(b) Rules + Greedy Connect

Figure B.11: Topology comparison for $\alpha = 60$, with node 4 being greedy, and unconstrained maximum node degree. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic. About half the time, using the rules, node 4 is connected to the graph center, while using the greedy search, node 4 is always either a member of the graph center or connected to a member of the graph center.

B.2 Constrained node degree, $maxDegree = 4$

Table B.3: Number of edges, $maxDegree = 4$

α	exhaustive	greedy	rules
0.5	39	39.4	39.2
1	39	38.9	38.8
5	34	34.6	38.1
60	19	19.0	19.0
200	19		19.0

Table B.4: Graph transport cost, $maxDegree = 4$

α	exhaustive	greedy	rules
0.5	712	740	810
1	712	743	805
5	712	720	737
60	1195	1323	1293
200	1195		1293

Table B.5: Graph diameter, $maxDegree = 4$

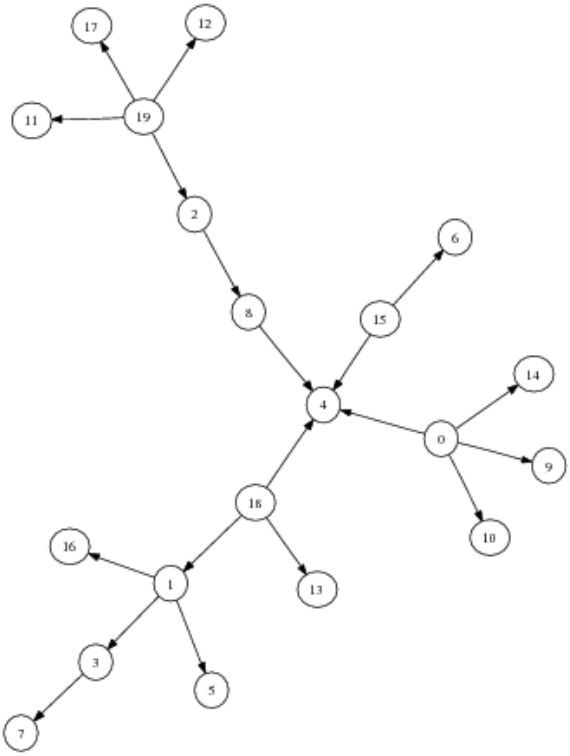
α	exhaustive	greedy	rules
0.5	4	3.9	4.2
1	4	3.9	4.3
5	4	4.0	3.9
60	6	7.1	6.4
200	6		6.4

Table B.6: Graph characteristic path length, $maxDegree = 4$

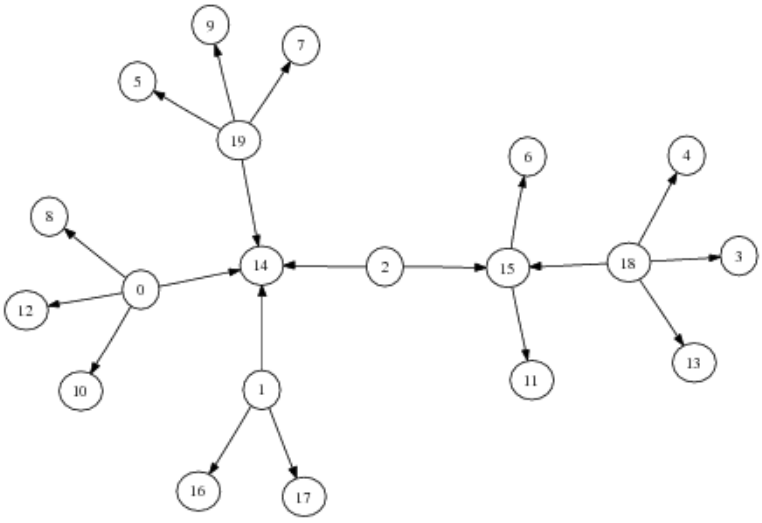
α	exhaustive	greedy	rules
0.5	2.10	2.11	2.28
1	2.10	2.13	2.29
5	2.27	2.21	2.19
60	3.30	3.62	3.49
200	3.30		3.49

Table B.7: Graph spectral radius, $maxDegree = 4$

α	exhaustive	greedy	rules
0.5	3.95	3.96	3.95
1	3.95	3.93	3.93
5	3.59	3.65	3.89
60	2.65	2.56	2.61
200	2.65		2.61



(a) Greedy search



(b) Rules + Greedy Connect

Figure B.12: Topology comparison for $\alpha = 60$, with node 4 being popular, and $maxDegree = 4$. Figure (a) shows a topology formed using iterative greedy search while Figure (b) gives a representative topology formed using rules with the greedy connection heuristic. Node 4 is *never* a member of the graph center when using the rules to create the topology. When using greedy search, node 4 is almost always connected to or a member of the graph center.

Appendix C

Additional 100 Node Results

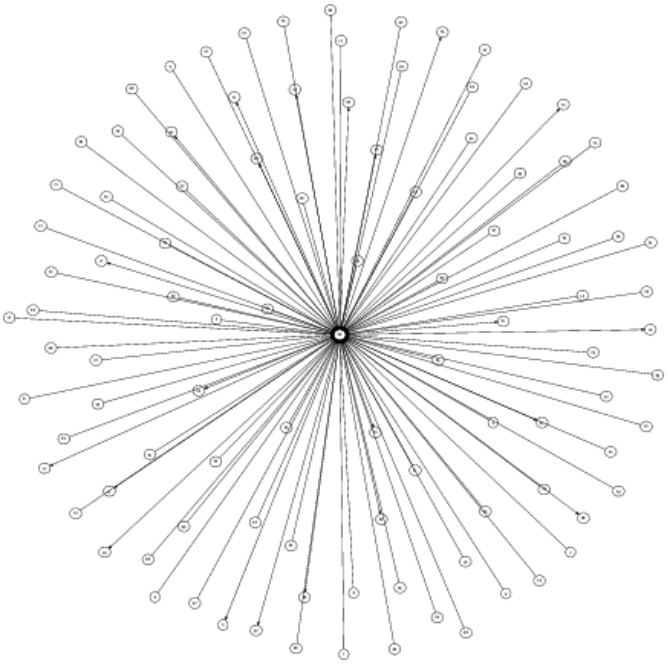


Figure C.1: Network formed using IGS, $\alpha = 60$ and Normal traffic-demand distribution.

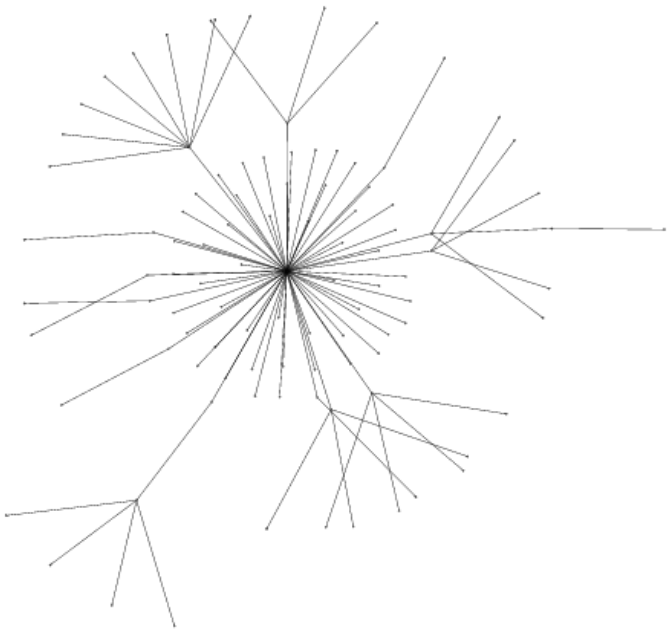


Figure C.2: Network formed using rules, $\alpha = 60$ and Normal traffic-demand distribution.

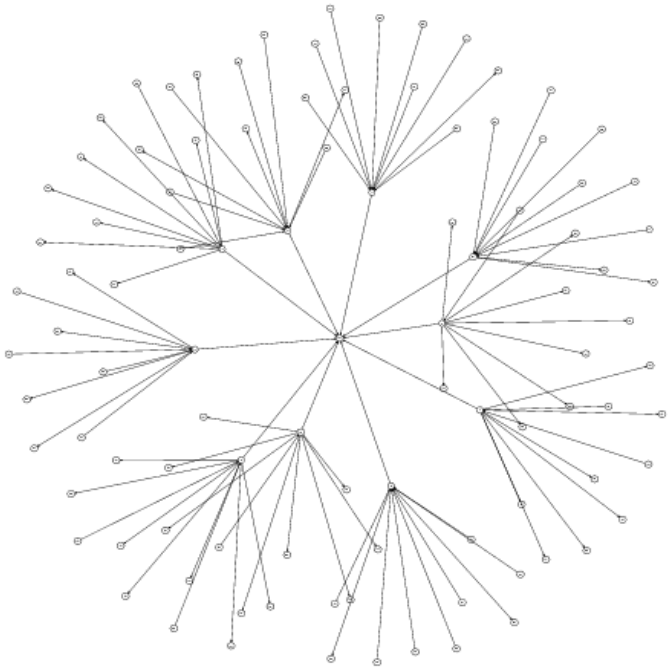


Figure C.3: Network formed using IGS, $\alpha = 60$, $maxDegree = 10$, and Normal traffic-demand distribution.

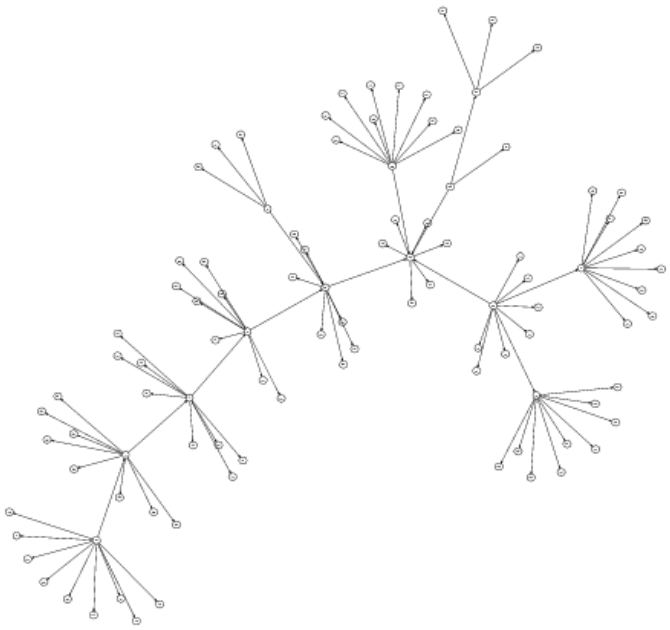


Figure C.4: Network formed using rules, $\alpha = 60$, $maxDegree = 10$, and Normal traffic-demand distribution.