

Math 164 Final Report: Virus-Spreading in Scale-Free Networks

Alex Popkin

April 19, 2004

Abstract

Computer viruses are increasing in number and severity, and current tactics for preventing their spread frequently fail. To understand the computer virus phenomenon, I used a model of the internet called scale-free networks. I collected data from three different types of epidemic simulation. The results suggest that a concentrated effort to use anti-virus techniques at the busiest e-mail servers with frequent updates would significantly reduce the spread of e-mail viruses.

1 Introduction

1.1 Scale-Free Networks

For the past few generations, the standard approach to modeling networks has been to use random graphs. In a random graph, every node in the vertex set has an equal probability of being connected to every other node by an edge. Random graphs tend towards having very uniform topologies, where the degrees of vertices fall in a standard normal distribution with low standard deviation. Vertices of very high degree or very low degree are extremely rare.

In recent years, researchers have discovered that random graphs are not always adequate for modeling certain networks. For instance, consider the internet as a graph where each web page is a vertex and an edge indicates a hyperlink between two documents. In such a representation, it turns out that most vertices in the resulting graph have a degree that is below average, indicating that they are linked to only

a small number of other documents. A small number of vertices have degree that is several orders of magnitude above average. Thus, the entire internet does not conform to the model predicted by a random graph. [2]

To model the internet, researchers created a new type of graph known as a scale-free network, or scale-free graph. The creation of a scale-free network proceeds as follows. We begin with a small connected graph and then iteratively attach new nodes. However, when a new node is added, it does not have an equal opportunity of being connected to each existing node. Instead, the probability of the new node connecting to each existing vertex is proportional to the degree of that vertex. [2] For example, suppose we add a new vertex n to the graph in figure 1:

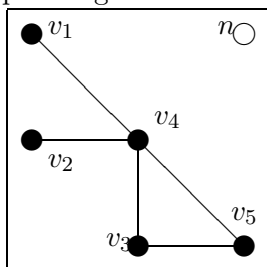


Figure 1

Here v_1 and v_2 have degree 1 and there are 5 vertices in the graph, so the probability of an edge forming between n and either v_1 or v_2 is $\frac{1}{5}$. v_3 and v_5 have degree 2, so the probability of an edge forming to either of those vertices is $\frac{2}{5}$. Lastly, v_4 has degree 4, so the probability of a connection between n and v_4 is $\frac{4}{5}$. This process of adding nodes can be repeated an indefinite number of times; some models have included scale-free networks of $\sim 10^9$ nodes. [5]

1.2 Properties of Scale-Free Networks

Does this model of scale free networks meet the properties observed in many real-world networks? Researchers investigating the subject plotted the number of vertices exhibiting every whole number degree. They found that for a degree k , the number of vertices with degree k was proportional to $k^{-\gamma}$, where $\gamma = 2.3 \pm .1$. In other words, the degrees of vertices in the network exhibited a power law distribution, rather than a normal distribution. The practical implications of this result is that this model of scale-free networks does exhibit the desired properties. The majority of nodes have low degree, but a few nodes

have very high degree. This result proved to be very robust, holding true when the number of vertices was as small as one hundred or as large as one billion. [5] Vertices of a scale-free network with very high degree are referred to as hubs.

Another interesting property of scale-free networks is that they fall into the classification of small-world networks. Given a graph G , we define the bandwidth of G as the maximum length of the shortest path between any pair of vertices in G . A small-world network is one where the bandwidth b is much smaller than the total number of nodes N .

1.3 Applications

The study of scale-free networks began in 1998, and it grew out of attempts to model the entire internet. It had previously been assumed that the internet resembled a random graph, with links forming between web pages without preference. However, a computerized analysis showed that the internet featured a large majority of web pages with very few links and a small number of hubs with a huge number of links. In other words, the internet proved to be a scale-free network.

Scientists soon found applications of scale-free networks across a variety of other fields. Biologists studying the spread of epidemics found that a scale-free network was often the best model for the interactions among potential carriers of a disease. Other applications showed up in fields ranging from physics to molecular biology. Social scientists also found scale-free networks appearing in unexpected places. For instance, the set of all scientific papers can be viewed as a graph, with papers being nodes and edges indicating when one paper contains a citation to another. This proved to be a scale-free network. [2] [6]

1.4 Current Research

The recent discovery of scale free networks was particularly interesting to people working in the field of computer viruses and virus protection. A computer virus is a program designed with malicious intent, which attempts to spread across a network and duplicate itself on as many different computers as possible. Although there are many different types of computer viruses, among the most common and damaging ones are e-mail viruses, which spread by hijacking computers to send out e-mails containing copies of the virus.

Researchers who study computer viruses tried modeling their spread using scale free networks. To do so, they created a dynamic model using graphs where each node represented an e-mail user and an edge represented e-mail communication between users. At any given time, each node was qualified as either susceptible or infected. For any two connected nodes where one was infected, they gave a probability λ that the other node would become infected during each time step. This model explained some of the most puzzling real-world explanations about the spread of computer viruses, including their long lifetime and their tendency to spread even when λ was very small. [4]

After verifying that scale-free networks were useful for accurately modeling the spread of computer viruses, the next question was whether they could actually be used to devise new strategies to prevent viruses from spreading. The first model explored was simply to label a certain percentage g of randomly selected nodes as immune, modeling the distribution of anti-virus software to internet users at random. This strategy did not prevent viruses from spreading, even when g was quite large, a finding that corresponded with the fact that most real-world attempts to block viruses have failed.

Intuitively, it seems that the most effective strategy would be to focus on making the hubs of the network immune. Researchers modeled a second strategy where anti-virus software was distributed not at random, but rather to the g fraction of nodes with highest connectivity. This strategy proved to be more successful, and it was found that for $g \geq .16$, all virus outbreaks were stopped. [3]

In considering the use of scale-free networks to model computer virus spread, we need to remember that scale-free networks are a very recent discovery. The first research on the topic was published in 1999, and the concept was only applied to viruses beginning in 2001. Thus, reserachers have only covered the basics of the topic, and many aspects of the problem have not been investigated. In my project, I intend to first verify the results about virus-blocking strategies and then create several variations on the model and test whether those variations provide any more useful information about strategies for fighting viruses.

2 Methods

2.1 Simulation

In order to run my own epidemic simulations on a scale-free network, I decided to create my own model. It is widely agreed that object-oriented languages are best suited for graph simulations. Consequently, I used the language that I am most familiar with, C++. There are several libraries available that take care of most of the details related to creating and using large graphs. I researched several of these libraries, but decided that none of them would be easy enough to modify for my purpose. Instead, I created the network from the ground up.

Fortunately, this proved to be relatively simple. I implemented the graph as an adjacency list. This is an array of all the **Nodes** in the graph, where each vertex object represents edges with a linked list containing all of its neighbors. In addition, each vertex object contained a degree counter and a variable indicating the status of the node (Infected, susceptible, or immune). The **Node** class also contained the functions necessary to change the node's status or to add a neighbor.

The **SF-Network** class encompassed all necessary data for the network, including the list of nodes. The **SF-Network** constructor takes parameters describing the desired network, including the number of nodes, and iteratively adds new nodes to the network under the rules described above. It generates random numbers for each edge to be added, and then compares those numbers to the percentage of total degree represented by each vertex to determine which vertex the new node will connect to. Other functions in the **SF-Network** class allow the user to run a simulation based on various parameters, and to print data about the network.

Some initial runs of the network creation code showed that on Turing, it takes about thirty seconds to create a network of 50,000 nodes. After that, an epidemic simulation can run through about five time steps per second, although the number does vary considerably based on the conditions of the simulation. In any case, even the longest simulations, which lasted roughly three-hundred generations, completed in under five minutes, so it was perfectly reasonable to do multiple trials with a wide range of parameters.

An epidemic simulation on a scale-free network, as described in

[4], is fairly simple. Nodes can be classified as either susceptible or infected. A certain percentage of all the nodes on the network are randomly chosen to be infected at the start. In each timestep, the program notes which vertices are adjacent to an infected vertex. Each such vertex then has a probability λ of becoming infected. Researchers are primarily interested in the lifetime of a virus during a simulation, so the program prints the number of infected nodes at each time step and records the time steps before that number shrinks to zero. When a node is immunized, that simply means it cannot become infected under any circumstances.

2.2 Assumptions

In order to reduce both the necessary code and the computation time for my simulations, I made several simplifying assumptions. The first concerned the status of infected nodes. In epidemic simulations on a graph, one must decide whether an infected node remains infected or moves back to the susceptible category after some time. For computer virus simulations, we assume that an infected node remains infected for some number ν time steps before it gets cured. In this model I set $\nu = 1$. This reflects the fact that once a computer system is infected, the owner or user will typically move to deal with the problem quickly. [3]

In researching computer virus spread, people are interested in a wide range of data. The most common data that gets collected are the lifetime of a virus, meaning the number of timesteps before zero nodes are infected. Some researchers also collect data on the number of nodes that are infected over time. They may also record what types of nodes are infected, for instance how many hubs versus non-hubs have the virus at each time step. [4] In my project, I decided to record only the duration of the virus during each run. While in some cases collecting more detailed data might have provided some additional insight, it would also have taken more time to compute and analyze.

In each simulation, there must be a chosen set of nodes infected at the beginning to allow the virus to spread. In real life, a programmer may distribute their virus from just one computer or from many. Thus, a simulation could begin with one infected vertex, or with a certain percentage of nodes randomly infected. The problem with the first approach is that the results will depend heavily on which node is

chosen. A virus that starts at a hub will spread quickly, while one that starts at a leaf may die out. I chose the second approach because it gives more uniform results.

3 Questions

3.1 Verification of Hub Immunization Strategy

My first set of test runs involved simply verifying the results previously reported in [3] about immunization of hubs as a strategy for preventing viruses. Specifically, I will tried various different values of g to determine whether $g = .16$ is actually the lowest acceptable value to stop at. To make the results more robust, I also tried various different values of λ , in order to account for the wide range of behaviors observed in computer viruses. For each set of parameters, I ran five trials and took the average lifetime of the virus.

3.2 Partial Hub Immunization

In the real world, human behavior affects strategies for combatting viruses. Even if working anti-virus software is available, some people may refuse to use it or install it incorrectly. Also, software may not work on all platforms, or malfunction under certain circumstances. Thus, I chose to investigate what would happen if instead of immunizing all hubs, I only immunized a portion of them. To do this, I added a loop to the model that went through the list of vertices and removed immunity from a certain percentage β of the hubs. Other than this modification, the details of the simulation remained exactly the same. Due to time constraints, I could only run this adjusted model with one value, $\beta = .01$. Again I tested a range of values of g and λ .

3.3 Dynamic Immunization

Both of the first two models used static immunization; every node was declared either immune or susceptible at the start, and that status never changed. However, such models fail to capture certain aspects of the computer virus problem. Once a computer virus is released,

other programmers around the world may obtain the code and modify it. A single anti-virus program may be able to identify and defend against the original virus but not the modifications. To combat this, companies may provide updates of their software that can identify the newer versions of the virus. It is worth investigating what effects these factors have on virus spreading.

To account for this, I created a dynamic immunization model. Before the simulation begins, the program creates a list of vertices that are classified as hubs and declares them immune. However, after each time step, it changes i percent of immune hubs to susceptible, and also declares j percent of susceptible hubs to be immune. If an infected node suddenly becomes immune, its infection is cured automatically. Again due to time constraints I only used one set of values, $i = .1$ and $j = .1$. In this case the number of immune hubs will remain stable over time, although the precise list of which ones are immune will change. Again I tested a range of values of g and λ .

4 Results

4.1 Verification of Hub Immunization Strategy

The results appear in Figure 1; each line represents results for a different value of λ . Virus durations were always infinite for small values of g , but grew shorter as more hubs were immunized. In all cases, viruses were eliminated immediately for $g = .17$. However, the shape of the curve depended on the chosen value of λ . For any mid-range value of g , lifetime is shorter for viruses with lower infection probabilities.

Previous research produced the value of $g = .16$ as the cutoff above which no virus could spread. In my simulation, at $g = .16$ lifetimes were very short for all viruses; at $g = .17$ the lifetimes shrank to zero. This minor difference might be explained by minor differences in topology or construction parameters for the network. Also, I was testing on a network of only 50,000 nodes, while other researchers have used much larger models; this may have played some role in the discrepancy. However, my results do confirm the previous result that total hub immunization can prevent the spread of computer viruses.

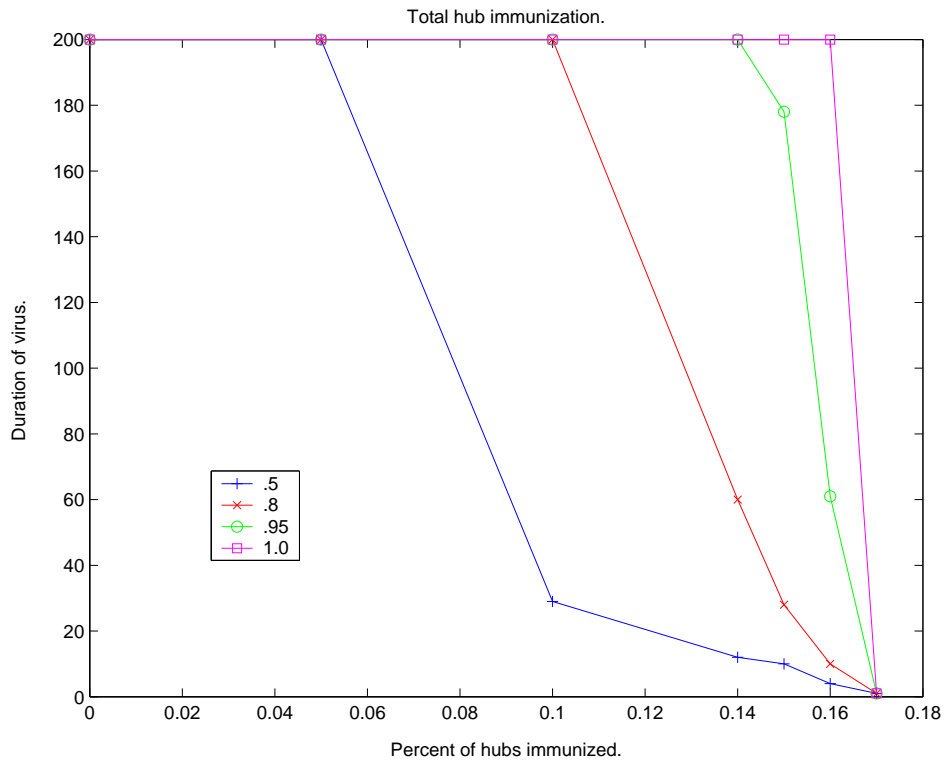


Figure 1: Virus lifetime as a function of the percentage of nodes classified as hubs, for four viruses with different values of λ . Note that infinite or very long lifetimes are represented as lifetimes of 200.

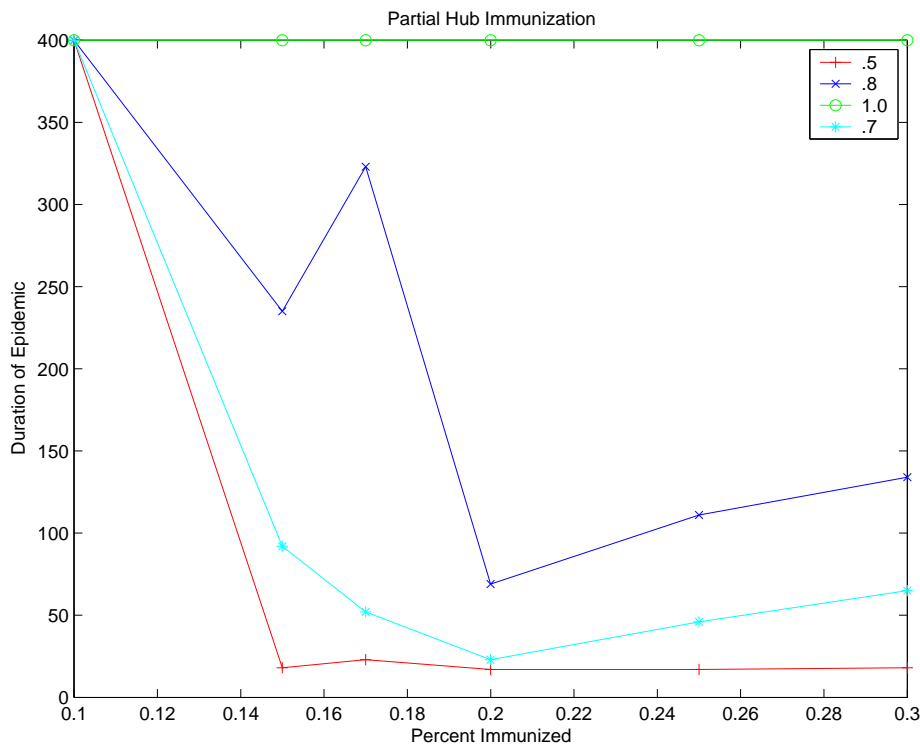


Figure 2: Virus lifetime as a function of the percentage of nodes classified as hubs, for four viruses with different values of λ . Again infinite or very long durations are represented at the top. Note that the axes are different from the previous graph.

4.2 Partial Hub Immunization

Again the results appear in Figure 2. We have four different values of λ shown. The results are obviously quite different from the total hub immunization model. In this case there is no cutoff at $g = .17$, or at any other value of g . In fact, for three of the viruses the duration appears to stay fairly constant as g increases past $.10$. For the other virus, where $\lambda = .8$, duration does decrease substantially between $g = .17$ and $g = .2$. Note that I only took data up to $g = .3$. This is because the entire justification for the hub immunization strategy is to avoid distributing immunity to too many nodes. If g gets too large, then the benefits of the strategy will be eliminated.

The results appear to indicate that a partial hub immunization strategy is not sufficient for preventing the spread of viruses. Even when we are immunizing the thirty percent of the nodes with the

highest degree, viruses still survive for unacceptably long times. One unexpected observation from this data was that in some cases the duration appears to increase when we immunize more nodes. Since removing nodes from the susceptible pool cannot help a virus spread, the initial response is that this must be caused by random variation. Ideally I would have time to take more data points and more trials at each point to get a more accurate picture of the situation. However, the data I took are sufficient to show that with partial hub immunization there is no cutoff for any small value of g .

To understand this phenomenon, consider the properties of scale-free networks. The number of nodes with a particular degree k is a power function of k . Graphically, this means that there is a long tail to the data; for a network of 50,000 nodes I observed an average of about 100 nodes with degrees ranging between 200 and 35,000. In a typical partial hub immunization scenario, at least one of these very highly connected hubs will have its immunity removed. Once this happens, eliminating a virus becomes difficult because an infected hub can easily spread infection to hundreds of other viruses.

4.3 Dyanmic Model

The results appear in Figure 3; each line represents results for a different value of λ . Virus durations were always infinite for small values of g , but grew shorter as more hubs were immunized. The overall result is similar to what was observed for the static total hub immunization model, namely that once we've reached $g = .17$, virus spread is severely reduced. In this case the viruses are not eliminated in a single timestep when we have large g , but they are eliminated in less than twenty timesteps.

5 Analysis

5.1 Conclusions from the Models

The first question asked was whether the results reported in [3] can be duplicated, specifically whether immunizing the 16 percent of nodes with highest degree stops all virus spreading. The simulation that I ran confirms this result. At some point between $g = .16$ and $g = .17$, we reach a scenario where all viruses are eliminated within one timestep. In practice, those nodes with highest degree in an e-mail

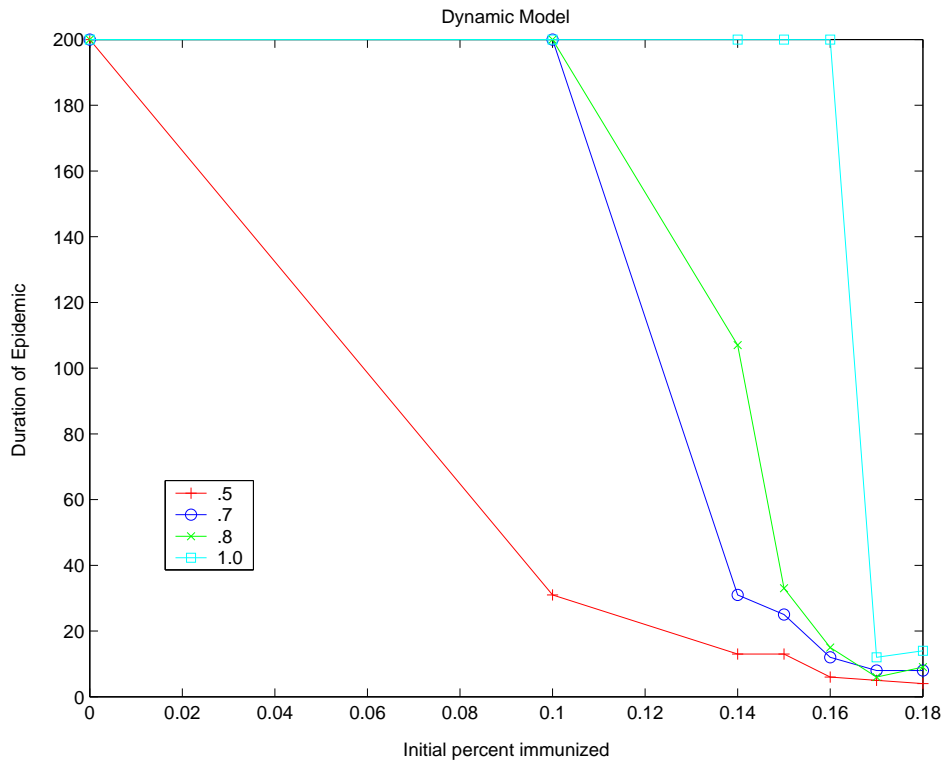


Figure 3: Virus lifetime as a function of the percentage of nodes classified as hubs, for four viruses with different values of λ . Again infinite or very long durations are represented at the top.

network would probably be the central e-mail servers for businesses, government offices, and schools. In addition, some individual computers with very high e-mail load might also be among the seventeen percent of nodes with highest degree. The test results suggest that if we could implement a completely effective virus solution on all of these hubs, all virus spreading would be stopped.

The second question was whether the results still held even when some percentage of these hubs are not immunized. The second model clearly shows that they do not. In fact, even when immunizing the 30 percent of nodes with highest degree and only removing immunity from 1 percent of those, viruses still persisted for a long time. Unfortunately, this model is probably more realistic with regards to human behavior. All the hubs on the global e-mail network are not controlled by any central authority; different people, businesses and governments all over the world control them. Thus, it would be hard to guarantee that an anti-virus solution was used in a correct and timely way on every single hub in existence. Furthermore, some anti-virus programs may not work on every operating system. In order to make the hub immunization strategy work, it would be necessary both to devise robust anti-virus solutions and to promote them aggressively to ensure that all hub controllers used them.

The final question was how viruses would spread on a network where the immunity status of hubs was constantly changing. The third model used provides results fairly similar to the first. In this case the duration of viruses isn't cut to zero after we immunize the 17 percent of nodes with highest degree, but it is reduced substantially. This tells us that in a real-world setting where new forms of viruses emerge and new anti-virus software is released on a continual basis, a strategy that focuses on immunizing hubs can still be effective. However, it would require certain activities that might be difficult. In the model, hubs that lose their immunity gained it back after just six time steps on average. To justify this, in real life it would be necessary for new versions of anti-virus software to be created and distributed to customers very quickly.

5.2 Parameters

In creating the model, several parameters were employed based on somewhat arbitrary choices. Before drawing conclusions about how to combat e-mail viruses in the real world, we should consider the

meaning of some factors that went into the models.

First, all time measurements were only recorded and presented in time steps, without defining what that meant in real time. Realistically, people hope to eliminate viruses within a week, while viruses that persist for several months cause the most damage. A time step in the model is the amount of time it takes for an infection to spread from one node to its neighbor. On computers, viruses typically spread once a user opens an e-mail or executes an attachment. One time step would then correspond to the amount of time between when an a virus first executes on one computer and when it gets sent to and downloaded on another computer. Some study of real world data would be needed to figure out how long, on average, this is. More sophisticated models might also account for the fact that some users check their e-mail more often than others.

Each run of a simulation includes an associated parameter λ that determines the probability of a virus spreading from one node to its neighbors. In general, larger λ makes a virus tougher to eliminate. In real life, a virus' chances of spreading depends on many factors, including how it operates, how careful users are with their e-mail, and what operating systems and e-mail clients the virus attacks. Again, a study of real data would be needed to determine what values of λ are realistic.

5.3 Future Work

The first extension of these models should be an expanded study of a wider range of parameters, particularly for the dynamic model. Due to time constraints, I only studied one set of parameters for changes in hub immunization, namely that where a hub has a ten percent chance of changing status after any timestep. Obviously there are many other possible scenarios worthy of investigation. One might look not only at other values of stable immunization, but also unstable cases where the number of immune vertices increases or decreases as time goes on.

As mentioned before, due to limited resources I could only run simulations of networks with 50,000 or fewer nodes. In real life, the internet has hundreds of millions of e-mail users, with more joining every day. Researchers studying scale-free networks often use much larger networks for their simulations. Any result found to hold on a small network should be verified on larger ones before definite conclusions are made about the internet.

In addition to size, there is considerable variety in the topology of scale-free networks. Researchers have shown that minor variations in starting conditions when creating a network can lead to major differences in the number of hubs and other qualities of the final network. [5] Other research suggests that using slightly different algorithms can create graphs that are more accurate representations of real-world networks. [1] Epidemic simulations should be run on these networks as well.

6 Acknowledgements

- Professor de Pillis, Department of Mathematics, Harvey Mudd College
- Computer Science Staff, Harvey Mudd College

References

- [1] Bianconi, G. and Barabasi, A. “Competition and Multi-Scaling in Evolving Networks,” *Europhysics Letters* **54** (2001) p.436-42
- [2] Barabasi, Albert-Laszlo and Reka, Albert. “Emergence of Scaling in Random Networks,” *Science* **286** (Oct. 1999) p. 509-12
- [3] Pastor-Satorras, Romualdo and Vespignani, Alessandro. “Epidemics and Immunization in Scale-free Networks,” in *Handbook of Graphs and Networks: from the Genome to the Internet*. eds. S. Bornholdt and H.G. Schuster, Wiley-VCH, Berlin, 2002
- [4] Pastor-Satorras, Romualdo and Vespignani, Alessandro. “Epidemic Spreading in Scale-free Networks,” *Physics Review Letters* **86** (2001) p. 3200-3
- [5] Barabasi, Albert-Laszlo, Reka, Albert, and Jeong Hawoong. “Mean-field Theory for Scale-free Random Networks,” *Physica a* **272** (1999) p. 173-87
- [6] “What does the Internet Look Like?” *The Economist* (Oct. 3, 2002)