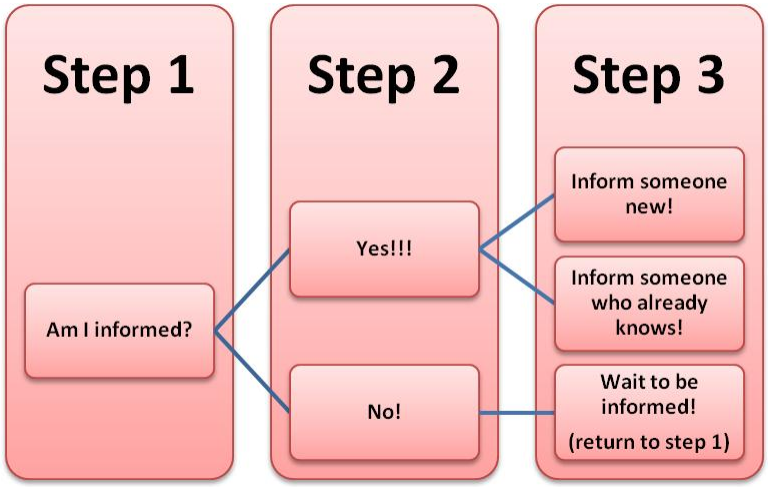
Rumor Spreading on Several Graph Topologies

Matthew Huberman
Charles Drake Poole II

May 19, 2010

1 Introduction

Randomized rumor spreading protocols are classically used to spread information across a network. We started with the model proposed by Doerr-Huber-Levavi, the basic assumption, where each node must follow certain rules to maintain the robustness of a random protocol to insure against transmission failure. Here is the flow of rumor spreading for the preexisting model:



Here is a quick rundown of our thought process, or what we'd want to add. No included is what we discussed involving efficiency and optimization.

1.1 Goal Notes

Chattiness

Based on where is the total amount of neighbors a node has and is the net weight of all the edge weights that the node has.

New edges

Apart from a nodes own neighbor set, a given timestep, there should exist the possibility for that node to make a new connection with a previously unconnected node and, on top of that, spread the rumor to that node in the same step. Maybe it should be less likely and based on the previously existing network, but should not be completely overlooked.

Quietness

A randomly assigned coefficient greater than 0 which calls to it the final say in whether or not the rumor signal is sent

Receiving

A randomly assigned coefficient greater than 0 which calls to it the final say in whether or not the rumor signal is sent

Credibility of Sender

A randomly assigned coefficient greater than 0 which calls to it the receiving nodes opinion of the senders honesty

Credibility of rumor

A randomly assigned coefficient greater than 0 which calls to it the believability of the rumors content

Receivers sheep factor/ gullibility

How likely they are to believe something with no facts, just being told. Like, ”The moon is made of cheese, the astronauts who visited brought wine to introduce to the natives. The Selenites now enjoy wine and cheese parties thanks to American Astronauts.” This would effect their wanting/eagerness in spreading a rumor.

The main idea here is to expand on the preexisting model. We extend the protocol to define more efficient methods for robust rumor spreading by including human behavior in the model. This includes new variables like interconnected sender/receiver credibility, rumor credibility,and a nodes preset quietness factor. We will also considera model on an incompletely connected graph that takes the time, in between rumor spreading time steps, to add innew edges as the rumor spreads. We will compare our model against the original method of rumor spreading to show increased efficiency and more realistic effects.

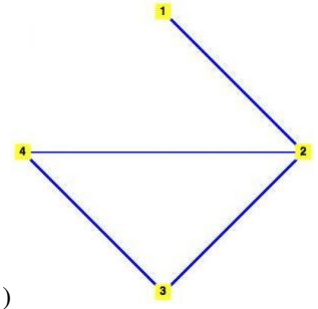
2 Programming Methods

2.1 Linear Algebra Idea

Given that A is the adjacency matrix of e connected graph and [R1] is the initial state vector where some node has been informed. Is it possible to calculate a new state vector [R2], then [R3], and so on to [Rn]using the previous state vector, the, adjacency matrix, and some interesting transition matrix T?

$[A] * [T] * [R1] = [R2][A] * [T] * [Rn-1] = [Rn]$

Example Case: (1)



(2) $\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$

Here, we look at this example case of a simple connected graph and its respective adjacency matrix. If we suppose that node 1 is informed than the appropriate setup we would like to see to find a transition matrix is as follows:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} * \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} * \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

or

$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} * \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} * \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Since we cannot say for sure which order the matrices should be in, we are forced to take into consideration two possible transition matrices; one where the adjacency matrix goes first and one where it goes second.

$[A * B] * [R1] = [R2]$ or $[C * A] * [R1] = [R2]$

In the example case, we get these two options for a transition matrix:

$$B = \begin{pmatrix} 3 & 1 & -1 & -1 \\ 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 \\ -1 & 0 & 1 & 0 \end{pmatrix}$$

$$C = \begin{pmatrix} 2 & 1 & -1 & -1 \\ 3 & 1 & -1 & -1 \\ -1 & 0 & 0 & 1 \\ -1 & 0 & 1 & 0 \end{pmatrix}$$

Unfortunately, this does not yield anything interesting. The two possible transition matrices do not amount to much

in terms of showing how the rumor spreads in terms of the adjacency matrix, as hoped, but this is certainly an area to be looked into.

2.2 Programming

For the programming we used a mix of python and matlab to produce numerical data to analyze. Both had their strengths and weaknesses, and we learned a lot about programming different visual effects in both languages.

2.2.1 Matlab

Quick Intro

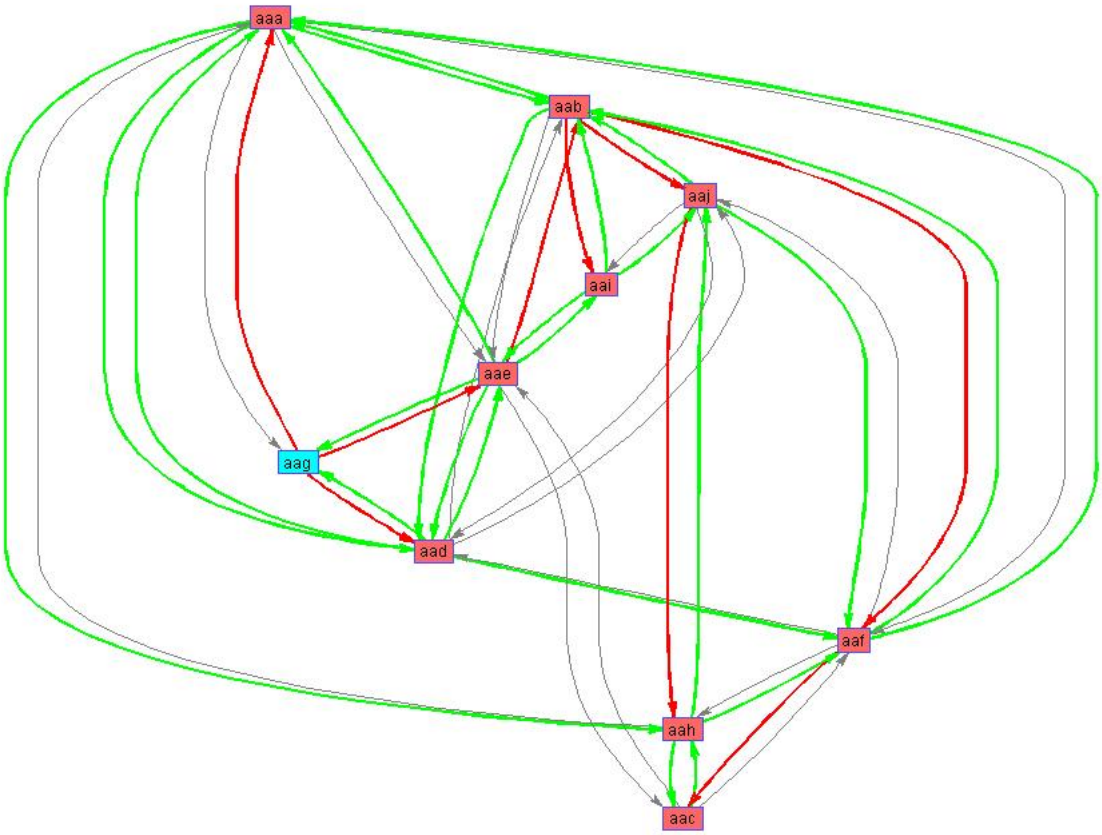
We really wanted to emphasize how computationally wasteful the original algorithm was. The idea was to show robustness while not throwing away so many cycles of informing. With the original algorithm, even when a node had informed all it's neighbors, it would continue trying to inform the same neighbors. Other concerns would be that, if this were actually implemented as a way of spreading information, there would have to be some meta communication for when the entire graph is informed. If there was this "meta structure" in place, then the robust quasi random spreading would be pointless, so back communication of being informed would be necessary to have everyone stop attempting to spread the rumor. For this we'll include a rule that reads,

If everyone you know, knows, Stop spreading, and tell them you're not spreading anymore

The idea is that this "counter rumor" will eventually stop everyone from spewing the same information over and over.

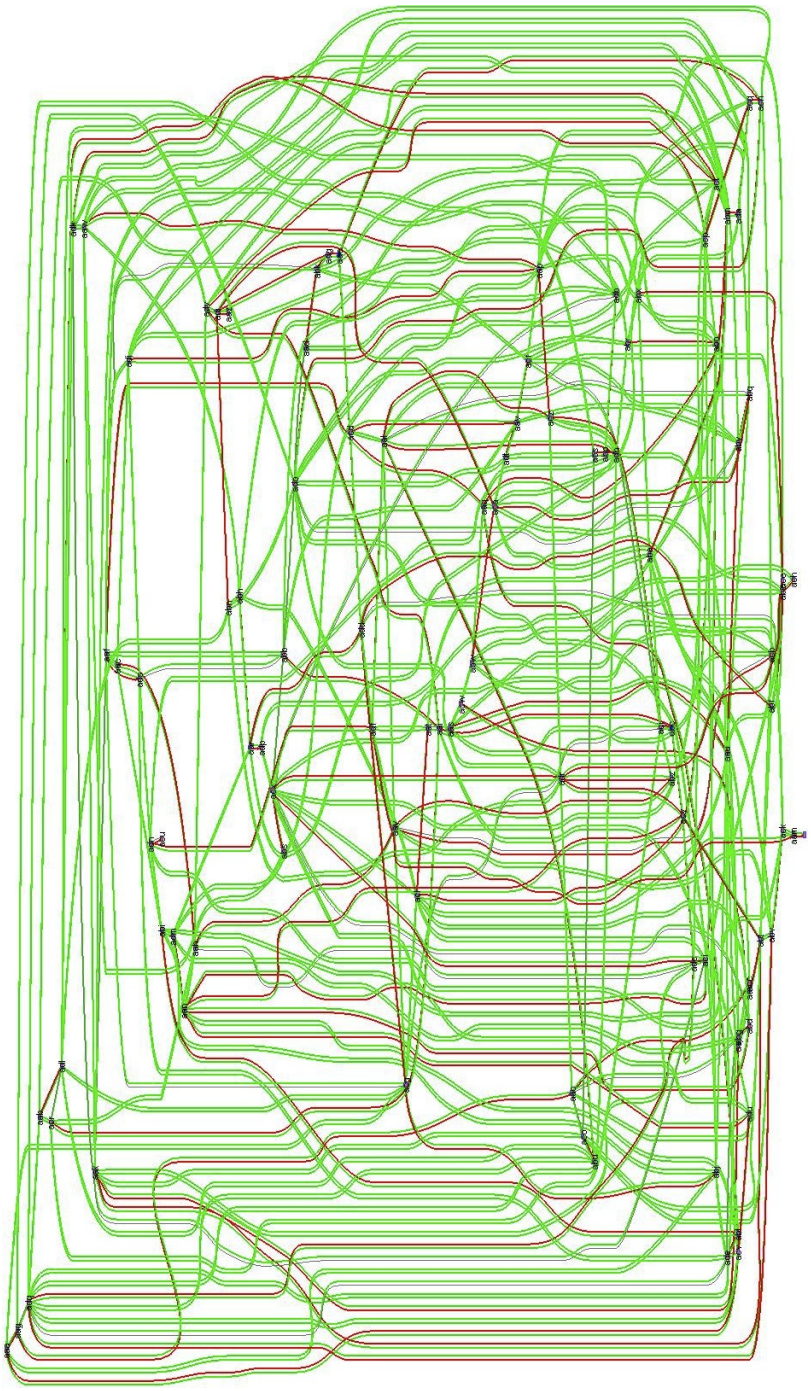
Graphs

Here are some graphical examples of the wastefulness of the algorithm employed.



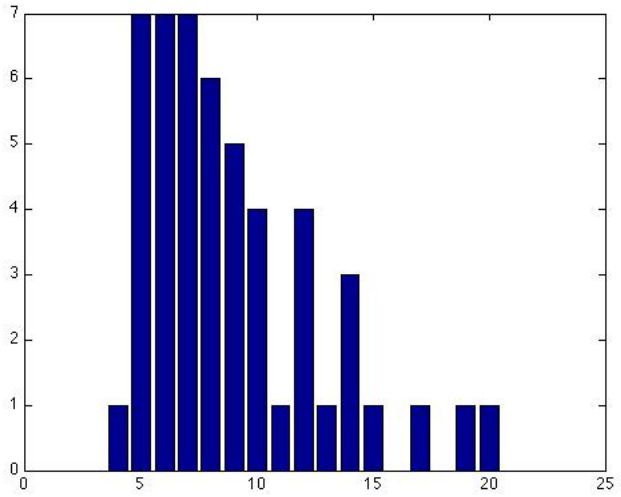
The red is a ”first spread”. The green is a ”respread”. Grey would be the unused vertices, by direction. Red and Green are what were interested in, the grey can be safely ignored.

From all the green on that graph it’s clear to see that nodes that are informed keep telling the same other nodes over and over and over. The effect is more evident on a larger more connected graph.



With this large graph there are very few gray paths. So, both in the sense there are more total nodes, so more total computation, but there is an increase in the pure wastefulness in that a larger percentage of the paths are reinforms instead of first informs.

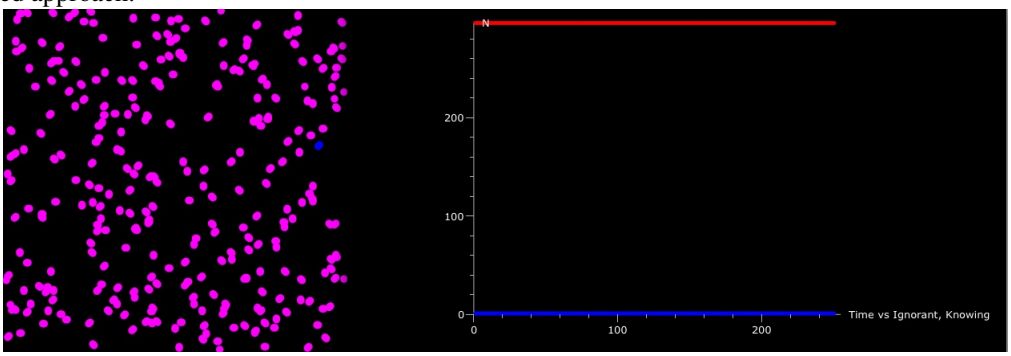
Also we did some investigation of the number of times steps taken to fill any given graph.



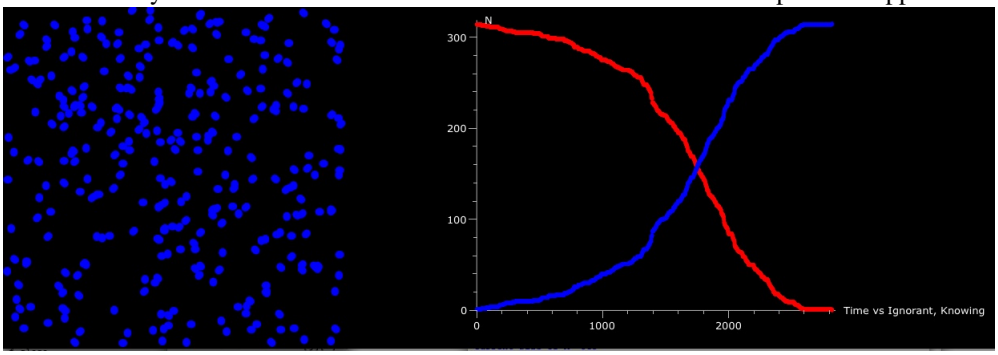
The runs end up making a curve with a strong indicator that there is a relation to the amount of nodes. With a fat body and very few tails. Not quite a bell curve.

2.3 Python

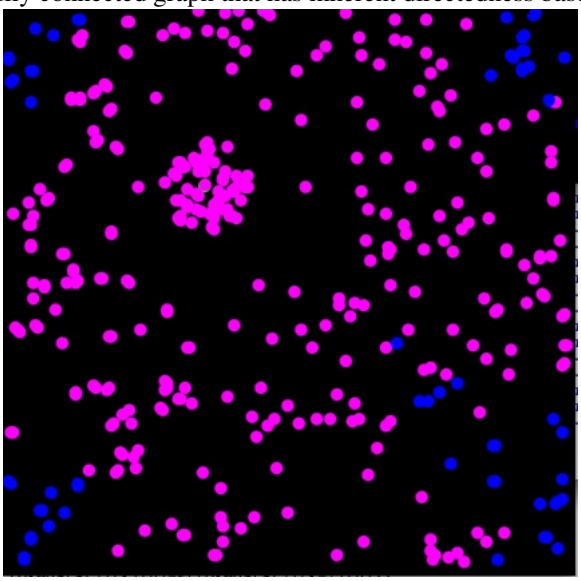
Another area of interest was programming an agent based model of this. Originally we wanted to avoide an agent based approach, giving each node the same rules that are carried out linearly. Python made it really easy for an agent based approach.



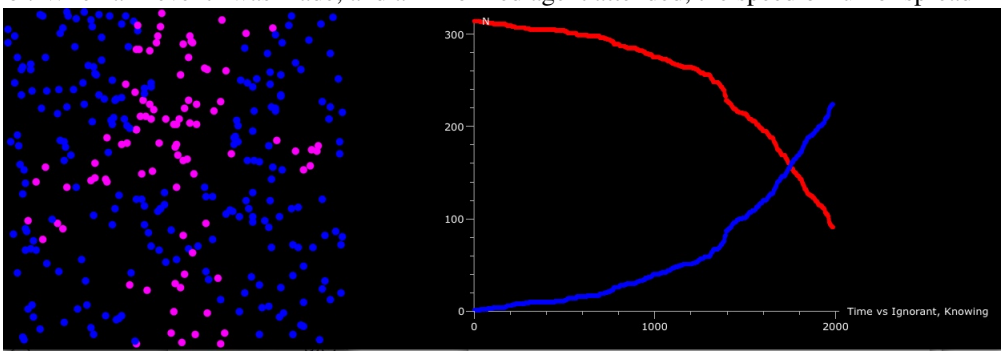
The agent based approach had agents moving on a torus. Every agent did a random walk on the torus and were informed when they were in close proximity to an informed node. The informed then informed others, and activity ceased when everyone knew the rumor. This had the same drawbacks as the previous approach.



Other drawbacks included no actual grids for who does and doesn't know each other. Each node could be considered connecting to each other node, with the only real information being a grid formed as they inform each other. Or, a fully connected graph that has inherent directedness based on agent interaction.



What was a neat feature was showing how attractors, like school, work, concerts, etc, could effect the speed of a rumor. When an "event" was made, and an informed agent attended, the speed of rumor spread increased greatly.



But, the clustering can hurt the spread at times. But, even when it's no in effect, the spread continues at a reasonable pace.

3 Conclusions

We spent most of the time programming and exploring numerical results. Charles is going to continue this research over the summer. Our results basically confirm the original paper's conclusion, and shows how wasteful such an algorithm is.

4 Future Work

- Random walks in higher dimensions
- Rumor spreading in higher dimentions
- Hybridized graph/agent model for rumor spreading

- Implement our original goals, the different modes.

5 Biblo

Sources/References: Leon, Steven J. (2006),Linear Algebra With Applications(7th ed.), Pearson Prentice Hall B. Doerr, A. Huber, A. Levavi, (2010). ” Strong Robustness of Randomized Rumor Spreading Protocols”. arXiv:1001.3056