Analysis of P2P mobile networking using Bluetooth

Rajwinder Singh 2008CS50223
Narinder Pal Singh 2008CS50218
Gyan Sagar Sinha 2008CS10170
Ravee Malla 2008CS50224

Assignment 4, Computer Networks

16 November 2011

Abstract

We present below a few interesting observations and experiments done on the data logs of bluetooth interactions between mobile devices. The application was written on a Blackberry phone and the data was analysed using Python and NetworkX.

1 Introduction

In this assignment, we designed a Blackberry application which uses the Bluetooth adapter to periodically (every 5 minutes) search in the viscinity for other mobile devices propogating a Bluetooth signal. The list of such devices' addresses are logged onto a central database timestamped by the server time. This database acts as a crowd-sourced data base which can be used to analyze and mine interesting patterns in the data and predict the usage patters and design efficient Bluetooth routing policies for P2P file transfer. We would like to extend this project into the winter break to be able to test our predictions and analysis on real networks.

2 Nature of Database

This database is created by the application developed by the students. Hence due to heterogeneity in the application implemented, there is bound to be a lot of noise in the dataset. We first try to analyze certain basic properties of the data, and note whether there are some things we can do to clean the dataset.

2.1 Database Description

The structure of the database is depicted below. Field 1 & 2 are 12 byte bluetooth adapter device addresses while Field 3 is the server timestamp when the entry was made.

Device 1 ID	Device 2 ID	Timestamp
002556D38946	14741134D023	237489237

Table 1: Database Schema

2.2 Database Cleaning

Various kinds of cleaning operations were tried out with variable success. We report the issues in the dataset and how we can tackle them.

2.2.1 Removing malformed records

Since we know that all device addresses are 12 character long, we remove all dummy entries like `(test,test,22748234)`.

2.2.2 Entry density

According to the specification, for some device pair d_1 & d_2, the entry (d_1, d_2) is to be logged once every 5 mintues. Hence, there can be atmost 12 entries for some device pair an hour. However, we find that there are sometimes as large as 220 entries for some device pair. This can give us a biased result in favour of those implementations that take more frequent readings. Hence we need to filter out all such entries.

2.3 Logging Pattern

An interesting (though expected) observation is drawn by studying the number of entries logged as the time passed from the day assignment was given. We find that as days progress, more entries are being logged per day. However, a general rise is during the *8 AM - 2 PM* period. On November 15, it is possible that some of the groups started running their application for the first time with the policy of logging entries very frequently which explains the sudden jump.

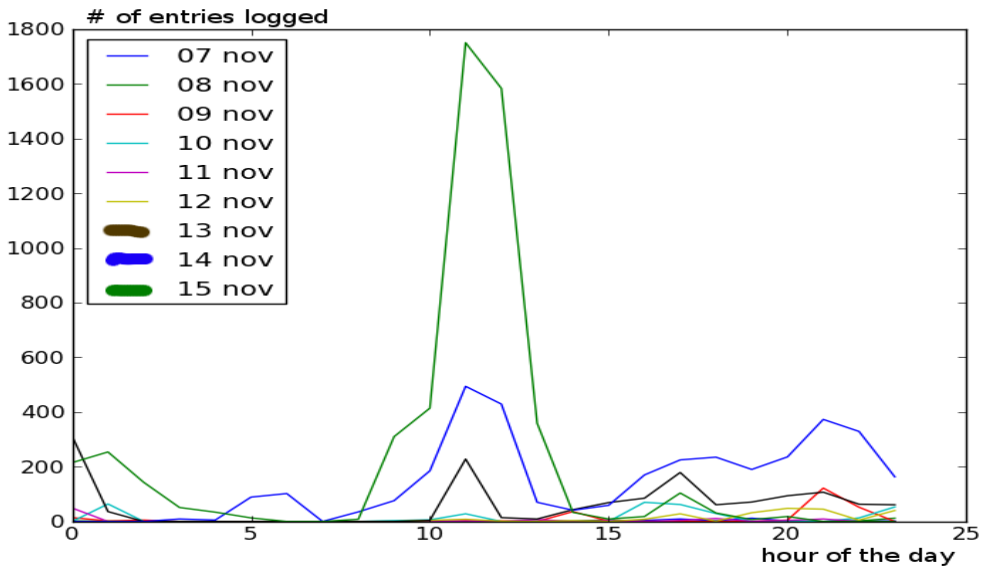

Figure 1: Number of entries logged per hour for various days starting from November 7

3 Database Analysis

3.1 Node Popularity

Owing to the problems described above, we have considered node popularity as defined in Q2 of analysis as the number of *unique devices* that come in contact with a given device. These are the top 5 most *popular* nodes in that sense. Here we have also tried out various other metrics like Total Database entries and Activity Period[1]. These hint towards the node `14741134D023` being a strong node, as in a smaller duration of activity and by logging fewer entries, it has actually communicated with a larger set of nodes.

Device Address	Unique Device Count	Total Entries	Activity Period (s)
14741134D023	292	1831	122000
1474113AAAAA	123	3956	390000
A86A6FE1E3D6	100	1641	135000
2CD2E7FFA7D5	89	806	180000
1814569239A7	78	1491	460000

Table 2: Popular Nodes

3.2 Node Rank Analysis

We can consider multiple metrics to measure node ranks.

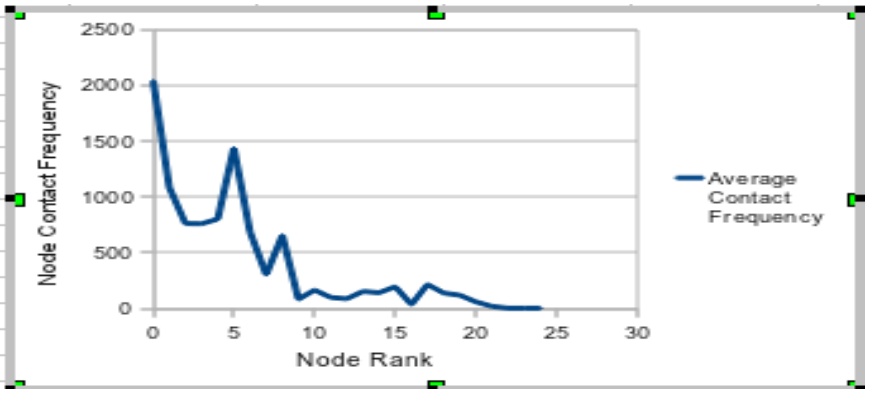

Figure 2: Plot of node rank against the average device contact frequency over all times

3.2.1 Ranking by a popularity metric

As we talk about in 3.1, *node rank* can be defined in a lot of ways. We can simply consider the number of unique nodes a given node can *directly* peer with. If we first consider a time independent static case, we can determine the rank from the number of unique devices a node pairs with. Contact frequency can be the number of log entries for this node over all times and devices. Before doing this, it is essential to clean the data of the noise discussed in 2.2.2 to prevent any biased readings. We have plotted the node rank and contact frequency in Fig 2.

[1]defined as the total duration this node has been logging entries

3.2.2 Ranking by a collaborative scoring mechanism

We tried out various scoring mechanisms which rely on the structure and connectivity of the graph itself along with it's temporal dependence to come up with a more dynamic ranking strategy. We have drawn inspiration from PageRank Algorithm of Google and similar ideas. The basic ideas are presented below.

- Temporal Coherence: Node u is of more *utility* if it has more peers temporally localized rather than far apart. Hence, we give higher weight to a node which has a larger neighbourhood in a time window τ as that implies a faster file propogation.
- Neighbourhood Quality: Just as in PageRank, we define $\forall u \in V, S_u(t+1) = \sum_{v \in N(u)} \frac{S_v(t)}{d_v}$. Let there be N nodes in the network. Assign each node a base rating of $\frac{1}{N}$. At each iteration, assign the new score according to the formula above. Repeat till the values converge.
- Hub Index: For a node u, define the hub index h_u as the ratio of the expected time required for a file to reach all parts of the network in the case u is removed from the network as against the case u is included in the network.

We tried to combine these heuristics together to get interesting insights into how the network is behaving over time. We find that PageRank procedure strongly correlates with our notion of popularity via unique devices peered in a time unit. If we can combine these metrics more carefully, we can simulate smarter heuristics and achieve better performance.

Device Address	PageRank Score
14741134D023	0.13387
1474113AAAAA	0.04271
A86A6FE1E3D6	0.035924
2CD2E7FFA7D5	0.0301357
2CD2E7FFA7D5	0.0301357

Table 3: Highly scored nodes via PageRank

4 Devices Network Graph

We drew the graph of the bluetooth network by considering all devices as nodes and drawing edges between devices if they come in contact, labeled by the timestamp at which their entry was logged. We have tried to vary a lot of things to draw many conclusions. Nodes are labeled by the device address and edges are drawn between node d_1 & d_2 if there is an entry (d_1, d_2) in the database.

4.1 Degree Distribution & Connected Components

The degree sequence of such a graph was analysed and found to be very linear. Most (about 500/650) nodes have degree 1, implying that they either form small linear chains. The **number of connected components is 1**. Hence this is a highly connected graph *without many central points*. Hence, we don't have many options for a time independent hub of some sort. This is because, on averaging over all times, we find that there are few nodes of high degree.

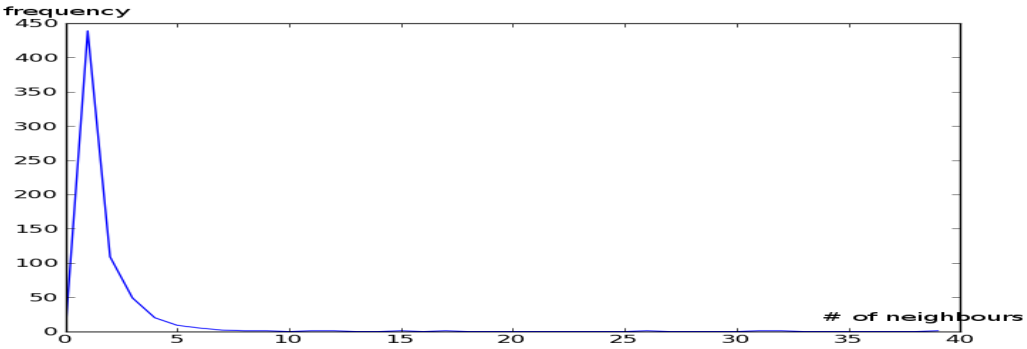

Figure 3: Degree distribution of the graph

4.2 Graph Visualization

We have tried to vary parameters and study the affect on the resultant graph. We have used this to draw conclusions on how such a P2P network gets formed and gets shaped based on the peering between nodes. The analysis is divided into the following cases. In this analysis, we have ignored the effect of time by averaging over all values of it.

4.2.1 Complete Graph

The complete graph is drawn and we clearly see 2 subnets appearing where the bridge nodes are the critical nodes in this network.

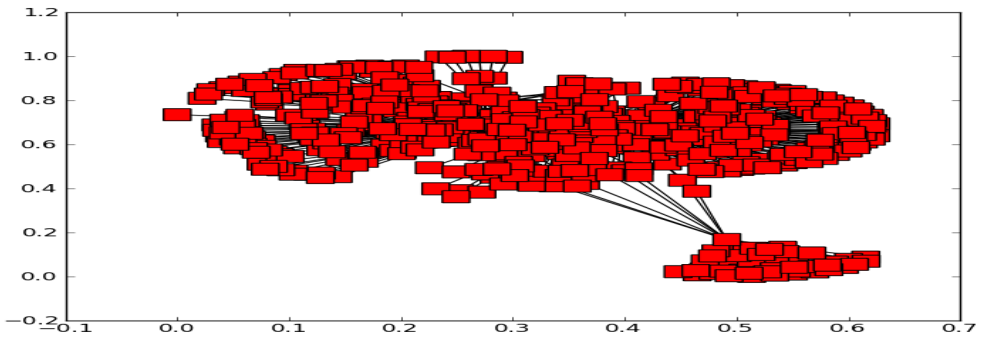

Figure 4: Graph over all times consisting of all nodes

4.2.2 Graph of Unpopular Nodes

We show here a few subgraphs of the above graph considering only nodes that are unpopular according to the metric defined above. We can get a sense of how such a P2P network *builds up* by iteratively considering a larger and larger set of nodes. Interestingly, the graphs at each step seem to grow as fractals and once a node of sufficient popularity comes in, 2 or more clusters get clubbed together.

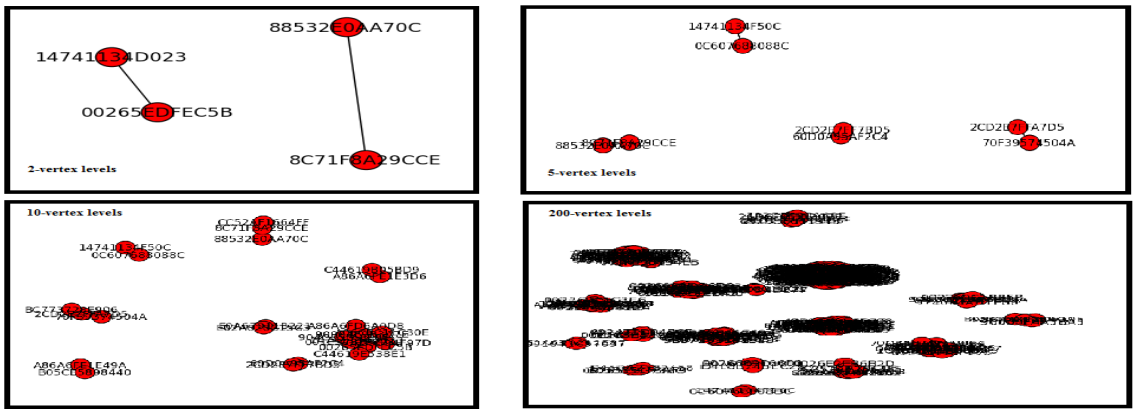

Figure 5: Graphing unpopular nodes and their neighbours in the network. a) 2 least popular nodes. b) 5 least popular nodes. c) 20 least popular nodes. d) 200 least popular nodes.

4.2.3 Graph of Popular Nodes

We find that with only a few popular nodes, we can recreate the whole graph as can be seen in this sequence. Hence, each of these nodes is a candidate for distributing a file and being sure that the file will propogate to all parts of the network.

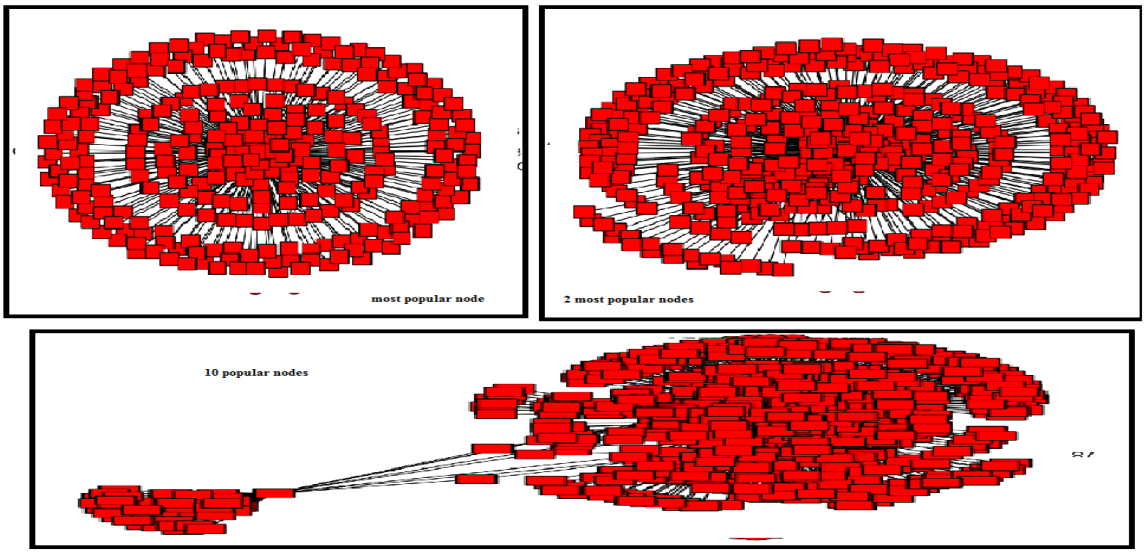

Figure 6: Graphing most popular nodes and their neighbours in the network. a) Most popular node. b) 2 most popular nodes. c) 10 most popular nodes which recreate the entire network by their neighbours.

5 File Propogation

We studied the file propogation patterns in the network, and find very interesting results. Although, there 650 nodes in the network, no node requires more than 35 hops to connect to every other node in the network.

5.1 Expected File Transfer Time

In the graph below, we report the frequency distribution of the number of hops required by the nodes to reach about 50%, 80% & 100% of the network. Since the time to reach all the nodes is about double that to reach 50% of the nodes, we can conclude that there are a few centers of content distribution in this simulation.

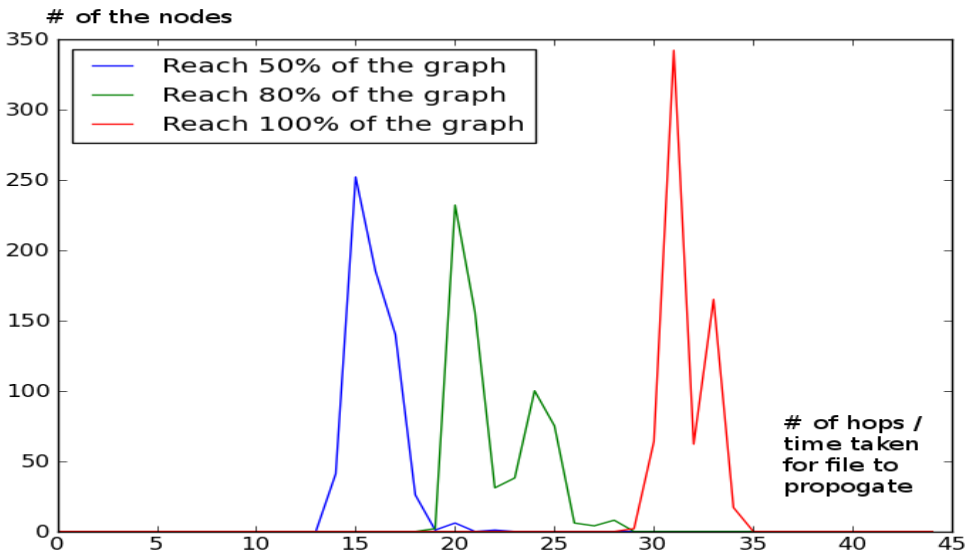

Figure 7: Histogram plot of the number of nodes which take time t to reach x % of the network

5.2 Correlation between Node Popularity and Transfer Time

Yes, we found a strong correlation between the popularity of a node and the transfer times. For eg. the node `14741134D023`, which we earlier determined to be the most popular, has $t_{\frac{1}{2}} = 15$, $t_{\frac{4}{5}} = 20$ & $t_{full} = 30$ which is the lowest for all cases.

6 Designing Routing Principles

Finally, we come to designing a few heuristics that can be used for efficiently propogating a file all through the P2P network. There is a key trade-off between the time required to completely propogate the file and the network resources utilized[2] in the process.

Once we have defined a score as defined above, we can almost route a file along a path of high scores and minimally copy files only at nodes which have high scores. This ensures that the files copied are copied into disjoint parts of the network and we can immediately reach farthest parts of the network once we reach these nodes of high scores. As we see from Fig 7, it takes atmost 40 steps to reach 100% of the nodes starting from any node. We can also do some time dependent routing schemes based on the past record of how different nodes come in and out at different times of the day.

[2]in replicating and forwarding files

We can use the scores computed by the PageRank algorithm to rank the neighbours of each node, and then propogate the file to only those nodes. We can define 2 kinds of nodes in the network.

- Sink Nodes: Nodes with low PageRank which don't replicate the file to their neighbours
- Forwarding Nodes: Nodes with high PageRank value which will replicate the received file to all of their neighbours

Using the above classification, we only need to replicate files at some of the nodes and still achieve the lower bound on the time required for a file to propogate to 100% of the files on the network. Empirically, we need only 5% of the nodes to replicate files to it's neighbours. This is an effective strategy to attain the trade-off between file propogation overload and propogation time.

7 Conclusion

In this report, we looked into many issues of bluetooth P2P networks, and wireless networks in general. We considered varied possibilities and played with control parameters to test our network. There are still some more analysis that can be done by looking into time dependent dynamic graphs. We have currently assumed a static model of the graph. The dataset can be made richer by including the geolocation tag with each entry, and physically mapping all the nodes for a realistic setting. Actual tests can be run to verify the simulation predictions. We would like to extend this project into the winter break to formally test some of the ideas that we had on nature of such adhoc Bluetooth networks and effectiveness of a routing scheme.