Application Layer Traffic Analysis of a Peer-to-Peer System

Dietmar Tutsch
Technical University Berlin
Institute of Computer Engineering
and Microelectronics
Berlin, Germany
DietmarT@cs.tu-berlin.de

Gilbert Babin
HEC Montreal
Information Technologies
Montreal, Quebec, Canada
Gilbert.Babin@hec.ca

Peter Kropf
University of Neuchâtel
Institute of Computer Science
Neuchâtel, Switzerland
Peter.Kropf@unine.ch

***Abstract*— Self-similar or multifractal behavior has been observed for LAN and Internet WAN (backbone) traffic. Investigations about this kind of behavior for application level protocols are rarely found because sessions or even applications are usually too short to be characterized in this direction. Only Telnet and FTP were examined so far. This paper analyzes the traffic shape of Peer-to-Peer (P2P) networks using the Gnutella protocol. Data was collected using a modified LimeWire servent. Self-similarity was estimated using a variance-time plot. The results show that Gnutella messages exhibit a self-similar shape, regardless of the message type.**

I. INTRODUCTION

Many of today's computer applications depend on a sufficient amount of available network capacity to perform the required communication. Thus, an important issue in network design is dimensioning the network such that an adequate bandwidth is provided.

In previous years, it was assumed that the network traffic could be described by a Poisson process: Interarrival times of discrete data are exponentially distributed. But during the last decade, it turned out that particularly background traffic in networks is multifractal or, in an extreme case scenario, self-similar. Self-similar traffic is defined as a traffic pattern that is invariant against changes in scale or size. If a part of the self-similar traffic is cut out and magnified, it will show the same structure and behavior as the non-magnified original traffic. Multifractal traffic can be viewed as "stepwise" self-similar traffic.

Leland, Willinger et al. discovered this phenomena of self-similarity concerning background traffic patterns first [1], [2], [3]. They investigated Ethernet LAN traffic to characterize the traffic shape at the data link layer (Layer 2) of the OSI Reference Model. These results motivated similar investigations at the network layer (Layer 3). Lucas et al. observed wide-area IP traffic of the Internet [4], [5]. Again, a self-similar traffic shape was detected.

A more detailed research on the kind of traffic shape refined previous results. It turned out that the traffic shape exhibits a more complex scaling behavior: it must be characterized as multifractal instead of self-similar. Dealing with a multifractal shape means that there is a kind of "stepwise" self-similarity with different degrees. For instance, Feldmann et al. documented the multifractal nature of Internet WAN traffic [6]. Carlsson and Fiedler [7] also worked out the differences between both traffic shapes. An example that models how the network performs under these traffic shapes is given by Tutsch and Hommel [8].

Investigations about self-similar or multifractal behavior of protocols of higher OSI Reference Model layers are found very rarely because sessions or even applications are usually too short to be characterized in this direction. Only Telnet and FTP were examined by Paxson and Floyd [9].

This paper concentrates on a popular protocol type at the application layer: the Gnutella protocol. The Gnutella networks are unstructured P2P networks using a "flooding" approach for searching contents and maintaining network structures. They are inherently scalable and self-organized without any central control. It is well known that P2P network applications generate a large amount of traffic on the Internet. Several investigations have also shown that content and request distributions have power-law like characteristics where few very popular objects are being requested most of the time. For dimensioning efficient networks (e.g. determining buffer sizes) it appears thus interesting to know the traffic shape generated by such applications.

The frequencies of the different Gnutella messages have been measured by Ripeanu [10]. But his measurement only gives mean values. Similarly, mean values have also been studied by Sen and Wang [11]. Besides the Gnutella network, they also investigated FastTrack and DirectConnect networks. Transient measures have also been performed but only examined concerning the hourly change at a day. The traffic distribution is not characterized.

Markatos [12] already discovered the self-similarity of the Gnutella traffic. But he did not investigate the self-similarity in detail and he drew no conclusions. Further on, he did not distinguish the different message types of the protocol. Our approach considers these issues.

In order to analyze the traffic shape of the Gnutella protocol, data was collected over long periods using a modified LimeWire servent. Self-similarity was estimated using variance-time plots. The results indeed show that Gnutella messages exhibit a self-similar shape, regardless of the mes-

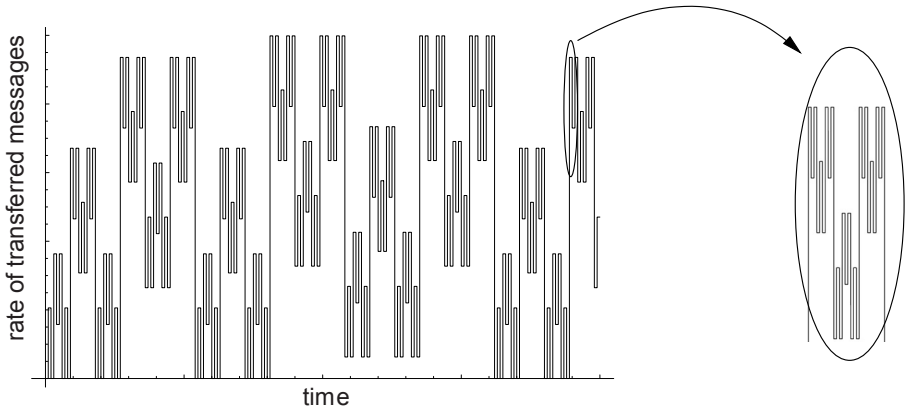

Fig. 1. Approximation of self-similar traffic

sage type.

In the following, we first introduce self-similar and multifractal behavior of network traffic followed by a brief description of the Gnutella protocol. We then present and discuss the measured results and the self-similarity observed in our study.

II. SELF-SIMILAR AND MULTIFRACTAL BEHAVIOR

Many approaches exist to characterize network traffic. One of them is to describe the message distribution in time. Particularly, the variation over time of the traffic density is addressed. The self-similar and multifractal traffic shapes provide examples of such traffic shapes.

A. Definition

Self-similar traffic is defined as traffic pattern that is invariant against changes in scale or size [2], [13]. If a part of the self-similar traffic is isolated and then magnified, it will show the same structure and behavior as the non-magnified original traffic again.

An approximation of a self-similar traffic is depicted in Figure 1. It shows some kind of rectangular function. If a part of this function is cut out and magnified, it will again show the same structure (right-hand side of the figure). This cutting and magnification (scaling) can be performed for several steps (time scales). Here, self-similarity arises because the function of Figure 1 is built by superposing periodical rectangle functions of different frequencies. The different frequencies represent different time scale factors.

Figure 1 does not exactly represent a self-similar traffic because cutting out and magnifying results for a few time scales yields a similar structure, but the similarity stops if scaling is continued further on.

Self-similarity can easily be defined if discrete time traffic is assumed. This is the case, for instance, for clocked systems. A time discrete signal $x(t)$ is said to be self-similar with parameter β ($0 < \beta < 1$) if for all $m \in \mathbb{N}\backslash\{0\}$

$$\text{Var}(x^{(m)}) = \frac{\text{Var}(x)}{m^{\beta}} \quad (1)$$

$$\rho_{x^{(m)}}(t) = \rho_x(t) \quad (2)$$

holds where Var denotes the variance, ρ the autocorrelation, and

$$x^{(m)}(\tau) = \frac{1}{m} \sum_{t=\tau m-(m-1)}^{\tau m} x(t) \quad (3)$$

describes the average value over m values of the original signal.

The parameter β characterizes the self-similarity. It is related to the Hurst parameter H which gives the degree of long-range dependence:

$$H = 1 - \frac{\beta}{2} \quad (4)$$

A value of $H \leq 0.5$ ($\beta \geq 1$) expresses the lack of self-similarity. The closer H is to 1, the greater the self-similarity. $H = 1$ means $\beta = 0$ and results in

$$\text{Var}(x^{(m)}) = \text{Var}(x) \quad (5)$$

If β is not constant for all m but constant with value $\hat{\beta}$ for a small m and with a slight change for a larger m, such traffic is called multifractal. In other words, self-similar traffic is an extreme case of multifractal traffic where β remains constant for all m.

The definition of self-similarity for continuous time traffic emerges from an extension of the theory above and can be found in [13].

B. Estimation

When observing network traffic, its self-similarity can be characterized by two main methods: the Whittle's estimator and the variance-time plot. The Whittle's estimator assumes the traffic shape to be self-similar and estimates the Hurst parameter via the spectral density of the underlying stochastic process. But usually, it is unknown whether self-similarity exists and thus, this method cannot be applied. In contrast, the variance-time plot provides both, the existence of self-similarity and an estimation of the Hurst parameter. This method will thus be used in the following.

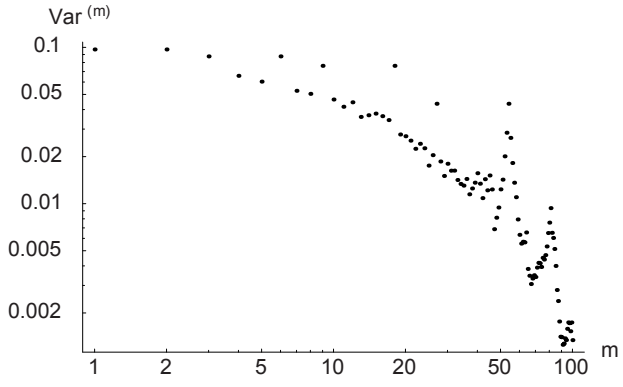

Fig. 2. Log-log plot of the given traffic

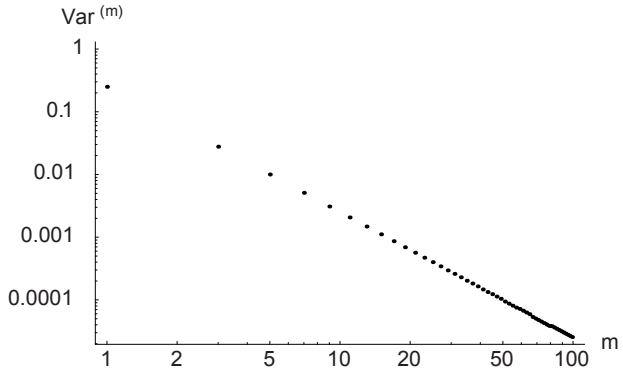

Fig. 3. Log-log plot of a single rectangle function

The variance-time plot follows from Equation (1) which can be rewritten as

$$\log[\text{Var}(x^{(m)})] = \log[\text{Var}(x)] - \beta \cdot \log m \qquad (6)$$

with $\log[\text{Var}(x)]$ independent of m. Thus, plotting $\log[\text{Var}(x^{(m)})]$ versus $\log m$ must result in a straight line with slope $-\beta$ where $-1 \leq -\beta \leq 0$ if $x(t)$ is self-similar. In case of multifractal traffic, the log-log plot shows a stepwise straight line with slopes $-\beta_i$ indicating the different degrees of self-similarity.

Figure 1 showed a given traffic with a simple shape. We determine its self-similarity as an example. Its multifractal properties are pointed out by the variance-time plot in Figure 2. The slope of a straight line representing the graph points (for lower m) equals $-\beta$ (-0.2)). That means β results in approx 0.2 and thus, the Hurst parameter of $H = 0.9$ exhibits very high self-similarity. For larger m, β changes: the traffic shape represents in fact multifractal traffic. For large m, β exceeds 1 and self-similarity vanishes.

In contrast to the previous example, a single rectangle function of period 2 results in the variance-time plot of Figure 3. In this case, $\beta = 2$ which indicates, as expected, that the single rectangle function exhibits no self-similarity and no multifractal behavior at all.

III. THE GNUTELLA PROTOCOL

The Gnutella protocol is a very simple protocol that supports file exchange without any central control [14]. A node joins the network by contacting a well-known host, which in turn supplies a list of potential neighbors in the current network. A node, or *servent* (for *serv*er and cli*ent*) will only connect to a limited number of neighbors. This number is configured by the system or the user and often set to a number between 4 and 8. All communications occur through these connections. If a connection is terminated, the servent attempts to open a new connection to another neighbor.

The network cohesion is preserved using a combination of Ping/Pong messages. Ping messages are flooded through the network while Pong messages are routed back to the servent from which the corresponding Pong originated. Each servent will periodically send Ping messages to all its neighbors (with a given time-to-live or TTL), which will forward it to their neighbors, etc., reducing the TTL at each node. The visited servents will keep track of all Ping messages received and from which connection it originated. They may also reply with a Pong message to the neighboring servent from which came the Ping, until it reaches the emitter of the Ping. The protocol helps preserve the integrity of the network, in spite of its very unstable nature [14]. This, however, is achieved at the cost of high bandwidth usage by the servents.

Searching for files uses a similar process using a combination of Query/Reply messages. Query messages are flooded through the network while Reply messages are routed back to the servent from which the corresponding Query originated. File transfer itself takes place out of band with the usual Web file transfer protocol directly between source and destination. In case a local area network does not allow its host to answer a file transfer request, but permits to initiate a file transfer, the Push message type provides a means to arrange for it.

Newer versions of the protocol support two types of servents : leaf nodes and ultrapeers. A leaf node will only try to establish connections with ultrapeers and will not participate in message routing. An ultrapeer will accept connections both from leaf nodes and ultrapeers, the number of connections being a parameter set by the user.

A. Experimentation Hardware/System set-up

The data analyzed in this paper was collected using a modified LimeWire servent [15]. The modifications did not change the basic operations of the servent. They provided the mechanisms to collect statistics about the servent, the network usage, and the messages received/sent.

Multiple experimental runs were performed in both leaf node and ultrapeer modes. We used the experimental methodology proposed in [14]. In each run, we execute two servents in the same mode (leaf or ultrapeer). One of the servents always uses the same parameters for all runs. This peer is called the *benchmark peer*. The other peer, called *test peer* uses different parameters for every run. For each run, whether in leaf or ultrapeer mode, the benchmark and test peers are run for 45 minutes, every hour for 24 hours. In ultrapeer mode,

we also keep track of incoming connections for an extra 4.25 hours (255 minutes) after the peer has stopped. A session corresponds to the execution of a single servent (benchmark or test peer), in either leaf or ultrapeer mode, for 45 minutes.

In leaf mode, a peer may only connect to ultrapeers. For each run, we changed the target number of connections to ultrapeers. The benchmark peer will try to stay connected to 4 ultrapeers.

In ultrapeer mode, a peer may connect to other ultrapeers and accept connection requests from leaf peers. For each run, we changed the target number of connections to ultrapeers and the maximum number of connections from leafs. The benchmark peer tried to connect to 32 ultrapeers and accepted at most 30 leaf peers.

The data collected includes :

- Statistics on all connections established during a session : duration, termination code.
- Statistics on all messages received/sent during a session : type, size, hopcount, TTL, date received.
- A list of all queries performed on the network while the servent was active.
- Statistics on the servent : bandwidth, horizon (number of reachable servents, files), connections attempts, received messages, routed messages, sent messages.

In this paper, we have limited our analysis to the study of the message flows. Specifically, we have only considered statistics pertaining to transient messages received/sent by the servent.

IV. MEASURED RESULTS

As already mentioned in the previous section, each run lasts for 45 minutes every hour. Thus, we separately determined the multifractal behavior for each of the 45 minute runs. This is quite a short time interval for such a type of long range investigation. Nevertheless, due to the high resolution of the time variable that we have available, this investigation still provides reasonable and useful results. On the other hand, very long range dependencies cannot be determined. Therefore, we only calculated the first (lower-m) slope in the variance-time plot estimating the self-similarity in the area of lower m. We refrained from determining multifractal behavior.

A. Considering All Messages

Figure 4 depicts the changing degree of self-similarity by the Hurst parameter H for the entire measurement period.

The x-axes represents the start time of the 45 minute interval for which the corresponding Hurst parameter is determined based on the number of bytes received/sent. The time (x-axes) of received, sent or dropped messages is given in milliseconds since January 1st, 1970. In the figure, the interval start time is divided by 10^8 to keep it readable. Measurements were started on July 21, 2003 at 2pm and ended on August 25, 2003 at 11am.

The Hurst parameter is displayed with narrow bars side by side. The areas where no bars are present are time intervals where no measurement took place, for instance, due to a server failure.

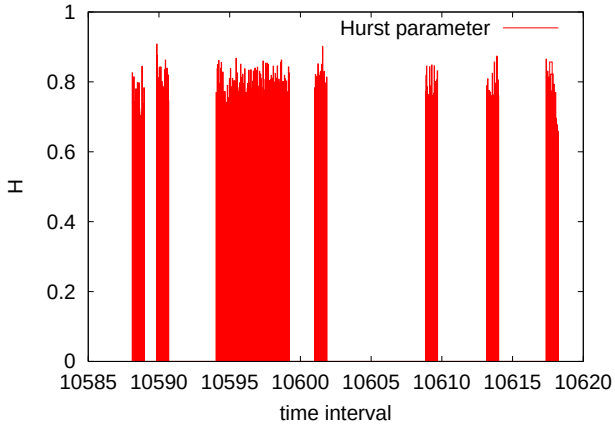

Fig. 4. Self-similarity of the measured traffic

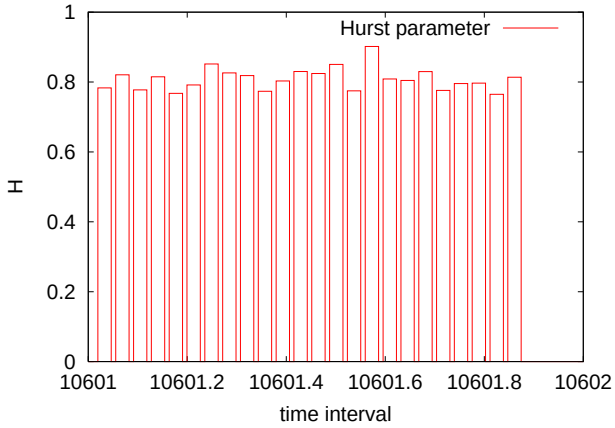

Fig. 5. Scaled part of the measured traffic

As can be seen from the figure, the Hurst parameter always exceeds 0.5, which is the lower limit of self-similarity. Moreover, the average Hurst parameter is around 0.7 to 0.8 indicating a high self-similarity. A small part of Figure 4 is scaled up and shown in Figure 5. It gives a more detailed view on how the Hurst parameter changes in adjacent intervals (hours) of measurement.

B. Differences between Message Types

In the following, the different message types of the Gnutella protocol are separately investigated : *Ping, Pong, Query, Reply, Push, Route Table, Vendor Message*.

The Route Table message is used by a leaf node to make its routing table available to an ultrapeer, and by neighboring ultrapeers to share their routing tables. The Vendor Message is used by user agent software to send proprietary information or for Gnutella extensions.

As explained in the previous section, Ping and Pong messages are strongly related and thus analyzed together. Figure 6 and 7 depict the self-similarity for the combination of both message types, in the entire time interval and in the scaled part, respectively. The average self-similarity resembles to that

of all message types together. However, the variance increases as a simple comparison of Figure 5 and 7 shows.

Query and Reply messages are also investigated together. The results are again similar to those of all message types and are thus omitted here.

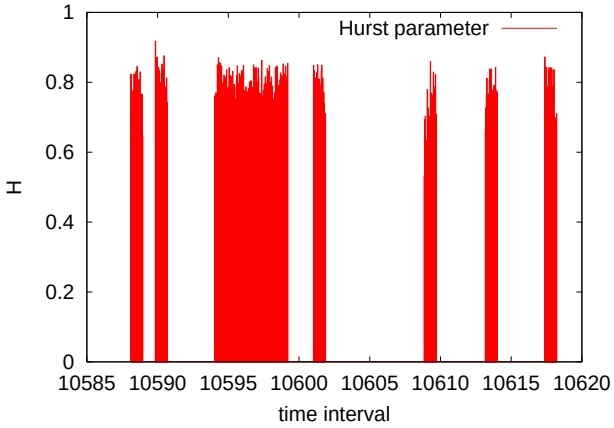

Fig. 6. Self-similarity of Ping/Pong

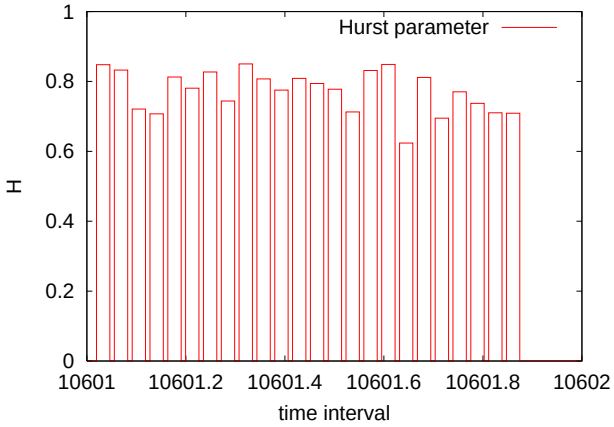

Fig. 7. Scaled part of Ping/Pong

Push messages reveal a different behavior (see Figures 8 and 9). This message type comes with a Hurst parameter of about 0.5 to 0.6 representing a very slight self-similarity. In some of the 45 minute time intervals, it drops below 0.5 which means there is no self-similarity at all in these cases.

Finally, the Hurst parameter of the Route Table messages is depicted in Figures 10 and 11. The self-similarity is clearly higher than for the Push messages but lower than for the message types 1 to 4. The figures for the Vendor Messages are again omitted because they show the same characteristics as Figures 4 and 5.

C. Interpretation

As stated above, self-similarity is observed for all message types of the protocol. To prove this point, we determined $\tilde{H}$ such that $P(H \leq \tilde{H}) < 0.0001$. Values obtained for all messages types are presented in Table I.

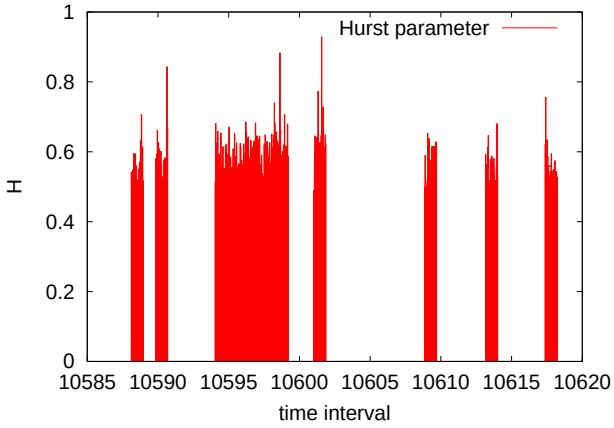

Fig. 8. Self-similarity of Push

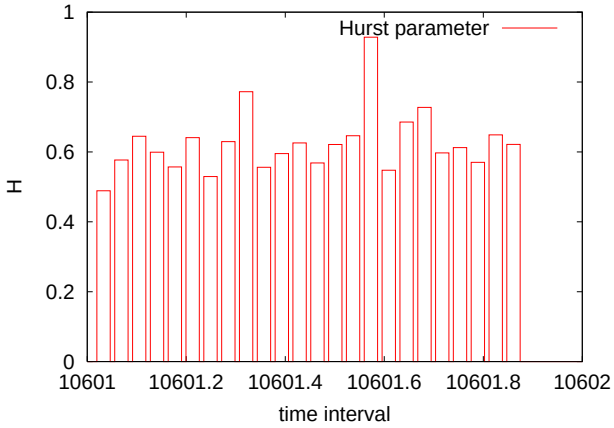

Fig. 9. Scaled part of Push

Message Type	$\tilde{H}$
Ping + Pong	0.752
Query + Reply	0.780
Push	0.563
Route Table	0.706
Vendor Message	0.766
All messages	0.769

TABLE I
VALUE OF $\tilde{H}$ SUCH THAT $P(H \leq \tilde{H}) < 0.0001$

Although Gnutella is an application level protocol, its role is similar to a layer 3 protocol in that it serves as a mechanism to forward queries and requests within the network. Therefore, it is not surprising to find that it had a self-similar shape for a short time span.

Furthermore, previous studies [16], [17] have shown that content popularity in Gnutella networks follow mostly a power-law like distribution. A quick look at the log-log plot of the power-law, which yields a straight line, clearly shows that content distribution is also self-similar. Therefore, the results we obtain are consistent with these other findings.

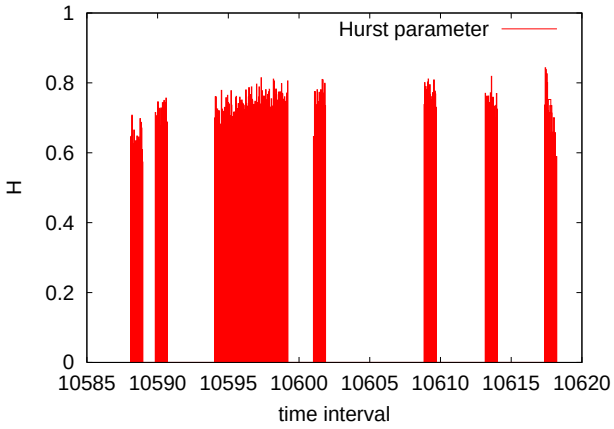

Fig. 10. Self-similarity of Route Table

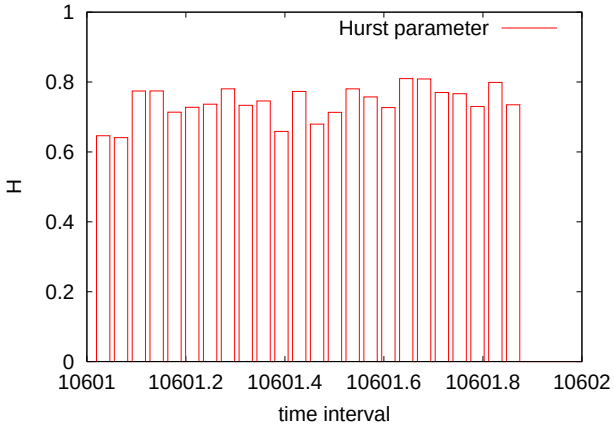

Fig. 11. Scaled part of Route Table

V. CONCLUSION

Based on the results obtained, we can conclude that the traffic shape for the Gnutella protocol is self-similar in the area of lower m for all message types. Indeed, the Hurst parameter for all message types is greater than 0.5 with a confidence level of 0.0001. We cannot conclude, however, that the traffic exhibits a self-similar behavior for larger time spans. This may only be determined with further data collection that would collect traffic data over a very long time span. Considering previous results on content distribution, we anticipate that the long range traffic shape will also exhibit self-similarity.

The self-similar shape of Gnutella traffic as observed means that the interarrival times of messages show a heavy tailed distribution. Messages thus produce bursty traffic resulting in high buffer utilization, increased bandwidth requirements and more processing power for routing. With the objective of providing and exploiting the most efficiently network infrastructures, we therefore suggest that a heavy tailed distribution for interarrival times of messages would be more accurate in modeling the P2P protocols than the commonly used exponential interarrival time assumption.

We have investigated the overall traffic as seen by a single node in the Gnutella network. The data collected would also allow us to study the traffic shape of a single connection, which still remains to be done.

ACKNOWLEDGMENTS

We wish to thank Frédéric Bastien who was instrumental in the modification of the LimeWire servent and for performing the experimental runs. We also would like to thank Carl St-Pierre for his insight on statistical analysis.

REFERENCES

[1] A. Erramilli, M. Roughan, D. Veitch, and W. Willinger, "Self-similar traffic and network dynamics," *Proceedings of the IEEE*, vol. 90, no. 5, pp. 800–819, May 2002.

[2] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–14, Feb. 1994.

[3] W. Willinger, M. S. Taqqu, R. Sherrman, and D. V. Wilson, "Self-similarity through high-variablility: Statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 71–86, 1997.

[4] M. T. Lucas, B. J. Dempsey, D. E. Wrege, and A. C. Weaver, "(M,P,S) – an efficient background traffic model for wide-area network simulation," in *Proceedings IEEE Global Telecommunications Conference*, vol. 3. IEEE Computer Society Press, Nov. 1997, pp. 1572–1576.

[5] M. T. Lucas, D. E. Wrege, B. J. Dempsey, and A. C. Weaver, "Statistical characterization of wide-area IP traffic," in *Proceedings Sixth International Conference on Computer Communications and Networks*. IEEE Computer Society Press, Sept. 1997, pp. 442–447.

[6] A. Feldmann, A. Gilbert, and W. Willinger, "Data networks as cascades: Investigating the multifractal nature of Internet WAN traffic," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, pp. 42–55, Oct. 1998.

[7] P. Carlsson and M. Fiedler, "Multifractal products of stochastic processes," in *Proceedings of the 15th Nordic Teletraffic Seminar (NTS-15)*, 1999.

[8] D. Tutsch and G. Hommel, "Multifractal multicast traffic in multistage interconnection networks," in *Proceedings of the High Performance Computing Symposium 2001; Seattle*. SCS, Apr. 2001, pp. 257–262.

[9] V. Paxson and S. Floyd, "Wide area traffic: the failure of Poisson modeling," *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226–244, June 1995.

[10] M. Ripeanu, "Peer-to-peer architecture case study: Gnutella network," in *Proceedings of the First International Conference on Peer-to-Peer Computing 2001*. IEEE, 2001, pp. 99–100.

[11] S. Sen and J. Wang, "Analyzing peer-to-peer traffic across large networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 2, pp. 219–232, Apr. 2004.

[12] E. Markatos, "Tracing a large-scale peer to peer system: an hour in the life of Gnutella," in *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*. IEEE, 2002, pp. 56–65.

[13] W. Stallings, *High-Speed Networks*. New Jersey: Prentice Hall, 1998.

[14] J. Vaucher, G. Babin, P. Kropf, and T. Jouve, "Experimenting with gnutella communities," in *Conference on Distributed Communities on the Web (DCW 2002)*, ser. Lecture Notes in Computer Science, no. 2468. Sydney, Australia: Springer Verlag, Apr. 2002, pp. 85–99.

[15] C. Rohrs, "Limewire design," LimeWire LLC, 2001, available from http://www.limewire.org/techdocs/design.html.

[16] M. Andreolini, R. Lancellotti, and P. Yu, "Analysis of peer-to-peer systems: workload characterization and effects on traffic cacheability," in *Mascots 2004*, Volendam, Oct. 2004.

[17] K. Gummadi, R. Dunn, S. Saroui, S. Gribble, H. Levy, and J. Zahorjan, "Measurement, modeling, and analysis of a peer-to-peer file sharing workload," in *SOSP-19*, Bolton Landing, NY, Oct. 2003.