p-sim: A Simulator for Peer-to-Peer Networks *

Shashidhar Merugu Sridhar Srinivasan Ellen Zegura
College of Computing,
Georgia Institute of Technology,
Atlanta, GA 30332
{merugu, sridhar, ewz}@cc.gatech.edu

Abstract

In the past few years, there has been intense interest in designing and studying peer-to-peer networks. Many initial measurement studies on current deployed peer-to-peer networks attempted to understand their performance [10, 11, 12]. However, the large size and complex nature of these networks make it difficult to analyze their properties. Simulation of these peer-to-peer networks enables a methodical evaluation and analysis of their performance. However, to our knowledge, there is no tool for simulating different peer-to-peer network protocols for a comparative study. We present p-sim: a tool that can simulate peer-to-peer networks on top of representative Internet topologies. p-sim has several capabilities to provide an accurate model of real-world peer-to-peer networks. p-sim can scale to thousands of nodes and is extensible to simulate new peer-to-peer network protocols. In this paper, we discuss the capabilities of p-sim, its user-interface and two case-studies.

1. Introduction

In the past few years, there has been intense interest in designing and studying *peer-to-peer* networks. The peer-to-peer networks differ from the conventional client-server approach in several ways. The most distinguishing characteristic is that a *peer* acts both as a client and a server of the system. For example, in file sharing, the most popular peer-to-peer application to date, a peer both requests files from its peers, and serves files to its peers. A second important difference is the transient lifetime of peers and their asynchronous arrivals and departures. While active, each peer maintains neighbor relationship with a small set of peers and participates in the application protocol on the peer-to-peer network. This neighbor relationship, a logical link between peers, defines the topology of the peer-to-peer network. The peer-to-peer topology forms an "overlay" on the IP-level connectivity of the Internet. Messages of the application protocol (e.g. search queries and replies of a file sharing application) are exchanged between peers on this overlay topology. Many initial measurement studies on deployed peer-to-peer networks attempted to understand the performance of peer-to-peer networks [10, 11, 12]. However, the dynamics in peer lifetimes and complexity of these networks make it difficult to obtain a precise comprehensive snapshot. Simulation of peer-to-peer networks provides a methodical evaluation and analysis of their performance.

To our knowledge, there is no tool for simulating different peer-to-peer network protocols, existing and proposed, so that the results of different proposals are directly comparable. In addition, we believe current network simulators such as *ns* [7] are not well suited for the task of simulating peer-to-peer networks. In peer-to-peer networks, we are interested in studying the macroscopic properties of the network as events happen, rather than the fine-grain packet-level dynamics. The events of interest in peer-to-peer network studies occur at medium time-scale and often include node arrivals/departures, search queries for locating files, and robustness/recovery from failures. Most traditional network studies, on the other hand, concern issues at the network and transport layer, for example, routing protocols, queuing disciplines, congestion avoidance and control, and throughput improvements. Another property of peer-to-peer networks that distinguishes them from regular network studies is the variability in topology. The overlay graph formed by the peers is constantly changing and this behavior cannot be easily modeled by current network simulators that assume a static topology. A further characteristic of peer-to-peer networks is their large size (on the order of tens of thousands of nodes [5]) and packet-level simulators such as *ns* are known to have concerns about scalability [9]. Thus these differences present a new set of criteria for simulating peer-to-peer networks.

∗ Please see http://www.cc.gatech.edu/people/home/merugu/p-sim/ for source code and documentation.

In this paper, we present *p-sim*: a tool that can simulate peer-to-peer networks on top of representative Internet topologies. *p-sim* allows comparative study of various peer-to-peer network protocols under common network conditions. *p-sim* attempts to provide an accurate representation of real-world peer-to-peer networks. The following is a list of capabilities included in *p-sim*:

- **Peer Dynamics:** The population of nodes in a peer-to-peer network is extremely dynamic with nodes joining and leaving the network. The time a particular node is connected to the network also varies widely. *p-sim* can simulate these variations in node lifetimes and arrival and departure patterns.
- **File Sharing and Searching:** Most peer-to-peer networks are used for content location and employ several different mechanisms for performing searches. *p-sim* includes implementations of multiple search protocols.
- **Evaluation:** Metrics on the peer-to-peer network are crucial to study the performance of its protocols and *p-sim* has mechanisms to evaluate metrics on both the overlay network as well as the underlying network.
- **Scalability:** Recent studies on work load of peer-to-peer networks [11, 12] attest to their immense popularity. *p-sim* has performance enhancements to handle the dynamics of thousands of nodes.
- **Extensibility:** *p-sim* is designed to be modular with clean interfaces so that it can be easily extended to implement new peer-to-peer protocols and search mechanisms.

The remainder of the paper is organized as follows. In Section 2, we give an overview of *p-sim* and how it works. The User Interface to the simulator is described in Section 3. We discuss enhancements to *p-sim* to increase its performance in Section 4. Section 5 provides two example usage scenarios of our simulator. Finally, Section 6 has concluding remarks and future work.

2. Overview

p-sim is designed to be generic and captures most of the key components of a peer-to-peer network. In this section, we highlight the sequence of operations in a typical simulation and describe the different components of *p-sim* that fit together to model a peer-to-peer network.

2.1. Simulation Framework

p-sim is based on the popular event-driven simulation technique [1]. In this approach, each event is associated with a *time-stamp*. Events have a description of an *action* and an *identifier* of the corresponding peer where the action is to be performed. Some of the events that are used in our simulator are:

- **Peer Arrival/Departure:** The arrival and departure events denote when a peer has become active (on-line) or inactive (off-line). These events are associated with individual peers and trigger re-organization of the overlay topology.
- **Search Query:** The file search query events are initiated at a peer and try to locate the specified file following the search protocol.
- **Evaluation:** The evaluation events serve two purposes: to monitor the simulation progress as well as to measure properties of the peer-to-peer network.

The list of events is ordered according to increasing time-stamps. We implement a priority min-heap to store the event list. The event list is initialized with certain known events and their associated times. The simulation engine takes the top event (with least time-stamp) and executes the action associated with the event. The virtual clock of the simulation "jumps" to the time-stamp of this event. This execution might add more future events to the event list. The simulation engine repeats this process until the event list is exhausted. The following subsections describe the events corresponding to different components in a peer-to-peer network.

2.2. Typical Peer-to-Peer Network Simulation

The specific details of simulation of a peer-to-peer network depend on the input configuration parameters, but we can broadly think of a simulation to proceed along the following sequence of operations:

1. Construction of the underlying physical topology at the router-level using an Internet Topology model (e.g., GT-ITM [15]). This operation also involves attaching end-hosts to the leaf routers to realize a model of the Internet and the computation of a routing table (both distance and next hop information) between every pair of end-hosts on this underlying topology[1].
2. Identification of hosts on the underlying topology that would participate in the peer-to-peer network and form an overlay topology. Initialization of the data structures (e.g., neighbor information, file storage) required for each peer.

1 We assume shortest-path routing, but it could be extended to other routing algorithms (e.g., policy-based) as well.

3. Distribution of the files among all the peers in the system according the input parameters for file replication. This step includes distribution of the origins of queries for these files as per the query placement parameters.
4. Estimation of the time of arrivals and departures of peers based on the input configuration parameters. We explain in Section 2.4 how the dynamics of peers are realized by arrival and departure events.
5. Initialization of the list of events with arrival/departure events, query search events, metric evaluation events, etc. and invocation of the simulation engine that executes these events in the increasing order of their timestamps.
6. Analysis of output log of the simulation. This output log records information about execution of events and also includes the values of metrics as determined by evaluation events.

2.3. Topology

A representation of the Internet topology provides a structural framework for simulating networks. Several models of the Internet topology and their synthetic topology generators exist (e.g., [6, 14, 15]). *p-sim* uses the Transit-Stub graph model[2] of the GT-ITM topology generator [15]. The Transit-Stub graph model is a representation of the hierarchical router-level topology of the Internet. We extend the router-level topology generated by the Transit-Stub model by "attaching" end-hosts to the leaf routers. The extended topology, thus serves as an underlying physical network. The number of hosts attached per leaf router is determined by a random number distribution as specified by user. Latency values can be assigned to the links of the extended topology according to the type of the link. There are four types of links depending on the endpoints of a link: transit-transit, transit-stub, stub-stub and stub-host. The latency values for each link type can be sampled from a random number distribution, also an input configuration parameter. Of all the end-hosts in the underlying physical network, a fraction of them are selected to participate in the peer-to-peer network. The number of peers in the peer-to-peer network and the choice of their arrangement on the underlying physical network are input configuration parameters.

2.4. Peer Dynamics

The population of peers is dynamic with random arrivals and departures. The activity of each peer is modeled as a two state Markov chain. The active state refers to the time when a peer is participating in the peer-to-peer network, while the inactive state refers to the time when a peer is "offline". A peer arrival event corresponds to a transition from the inactive state to the active state. Similarly, a peer departure event corresponds to a transition from the active state to the inactive state. A *session* comprises an arrival event and its corresponding departure event. The number of sessions per peer during the entire simulation is an input configuration parameter. Users can also specify a random number distribution to sample the lengths of both the active and inactive periods. The peer arrival and departure events are pre-computed and added to the event list during initialization. Any operations by a peer are performed only when it is "active".

2.5. File Search Protocol

p-sim implements two file searching protocols: *(1)* a *scoped-flooding* search protocol that is used by Gnutella and its family of protocols [4] and *(2)* a *random-walk* search protocol proposed recently for unstructured peer-to-peer networks [2]. Users can specify either search protocol for simulation. The input configuration parameters to simulator include the number of files in the system and a distribution that models the file replication (e.g. Zipf). These files are distributed among all the peers during initialization. We also create queries for these files and "spread" their origins over the peer-to-peer network according to user-specified input parameters. The search query process is simulated according to the search protocol by examining the file storage at each of the peers within search radius from query source.

2.6. Evaluation Metrics

We define some metrics below that are used to evaluate various aspects of a peer-to-peer network. These definitions are translated into actions that produce output log of the state of the network. The evaluation events are scheduled periodically at a frequency specified by the user. The following metrics are implemented in *p-sim*:

1. **File Success Rate:** A query for a file is *successful* when it finds at least one match for the requested file. *Success rate* of a file is defined as the ratio of number of successful queries to the total number of queries issued for that file throughout the system.
2. **Nearest Search Result:** Of the multiple file locations returned by a search protocol, the *nearest* one in the underlying network may be able to better serve the file than other locations. This distance to the nearest search result is an important metric when evaluating performance of a search protocol.

2 We have extended our simulator to work with Power-law random graph (PLRG) model [6, 14] as well. However these extensions are preliminary and need further support to scale to large topologies.

3. **Connectivity:** Since the peer-to-peer network topology is a variable graph, we check whether it is connected or not at every measurement instance. As part of this test for connectivity, we compute the diameter of the overlay topology among active peers.

4. **Stress:** *Stress* [3] of a link in underlying topology is defined as the number of logical links (between peers) whose mapped paths include the underlying link. Intuitively, stress measures global effect of the overlay topology of the peer-to-peer network on the underlying network.

3. Simulator User Interface

We designed a simple user interface language to write the input configuration parameters to *p-sim*. The input configuration has five parts: peers, files, search, topology and evaluation. Within each part, the parameters are specified as: (**keyword** `value`)[3]. Depending on the context, the `value` could either be a constant (e.g., (**count** 5000)) or description of a random number distribution to sample values from (e.g., (**activity** `EXPONENTIAL` 100) to extract samples from an exponential distribution with a mean of 100).

The **peers** part defines the characteristics of individual peers in the simulation. A total of **count** peers are created for the simulation. Currently, the number of **neighbors** maintained by a peer is a `RANGE` with specified minimum and maximum, but could easily be extended to define classes of peers each with a different range of neighbors. Similarly, the storage of **files** at a peer could be a `CONSTANT` or variable to incorporate heterogeneity among peers. The dynamics of peer population are controlled by the **sessions** parameters. Each peer is scheduled to have **numSessions** where the periods of **activity** and **inActivity** are sampled from the specified random number distributions.

The **files** and **search** parts configure the file storage and discovery of these files by search queries. A total of **numFiles** are distributed among all peers of the network. We have a provision to allow multiple copies of these files according to a random number distribution specified by **replication** parameter. This "spread" of files is based on a random number distribution specified according to **placement**. A `UNIFORM` placement picks locations at random uniformly among all peers to assign files. We could easily extend the file storage scheme to include other random number distributions to reflect "locality" in storage of files. The search queries for files are propagated according to the **protocol** up to a maximum of **radius** hops on the overlay topology. For example, (**protocol** SCOPED_FLOOD) and (**radius** 7) uses the scoped-flooding search protocol with a search radius of 7 hops. We can specify whether we want to simulate queries for `ALL` files or only a fraction of the files in the system. Duplicate queries can be generated according to the random number distribution specified by **duplicateQueries** parameter. The source of these queries can also be decided by sampling a random number distribution given by **queryOriginPlacement**.

The **topology** part describes the configuration of the underlying physical network topology. The topology is constructed by the transit-stub model of the **GT-ITM** topology generator using the configuration found at **location**. The router-level topology thus constructed is extended by adding hosts to the leaf routers. The numbers of hosts added to these leaf routers are sampled from **hosts** random number distribution. Of these end-hosts on the extended topology, we pick **count** number of hosts as participants in the peer-to-peer network. The arrangement of these participant hosts is dictated by the random number distribution given by **peerArrangement**. For example, (**peerArrangement** UNIFORM) picks end-hosts uniformly at random to participate in the peer-to-peer network. Latencies are also assigned to links in the underlying network according to random number distributions in the **linkLatency** parameters.

The final part of the input configuration has **evaluation** metrics. These parameters describe the evaluation metric and the frequency at which is it evaluated. For example, (**stress** 100) refers to a measurement of stress on the links in the underlying network every 100 ticks of simulation. This frequency of measurement is used in scheduling the evaluate events that are executed periodically.

4. Performance Improvements

In this section, we describe some of our efforts to decrease the run-time of execution of the simulator, especially when scaled to a large number of peers.

4.1. Hierarchical Routing in GT-ITM

We have implemented a piecewise route computation that greatly improves the efficiency of all-to-all route computation in GT-ITM topologies. The scheme computes routes independently within each stub domain and in a graph consisting of the border router for each stub and all the transit nodes. A complete end-to-end route is then constructed by piecing together routes in each stub domain with the route from origin domain border router to destination domain border router. The efficiency gains are considerable; for example, an all-pairs computation which took hours to run now takes tens of minutes.

3 Pre-defined keywords of our user interface language are shown in boldface in this section. Similarly, pre-defined values are in typewriter typeface.

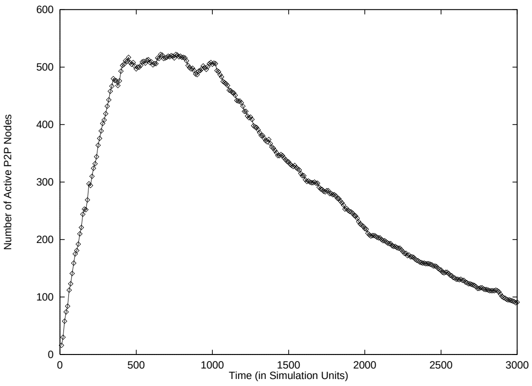

Figure 1. Population Activity with dynamic arrivals and departures

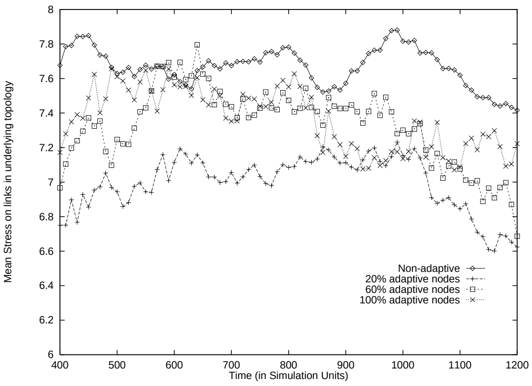

Figure 2. Effect of adaptation on mean stress of underlying links

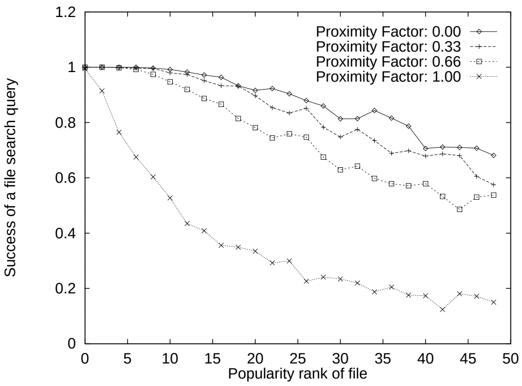

Figure 3. Scoped-Flooding Search Query Success Rate

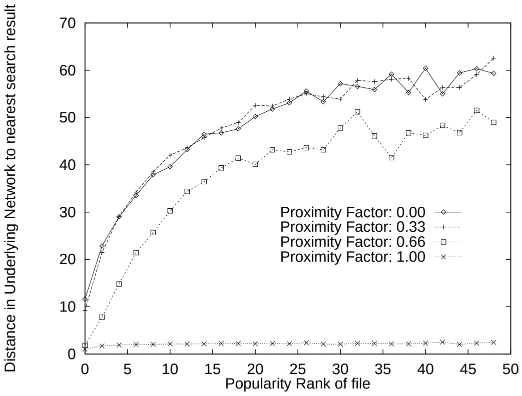

Figure 4. Underlying network distance to nearest search result

4.2. Delta Metric Evaluation

The evaluate events that monitor the state of the peer-to-peer network are scheduled periodically. Computing these evaluations, such as stress on the links of underlying network, as defined in Section 2.6 requires significant amount of time. However, we can reduce this computation time by observing that the number of changes that take place since the last evaluation may be far less compared to the total size of the system. Specifically, the number of logical links that are added or deleted since the last call to stress computation event may be less than the total number of links. Under these circumstances, it is more efficient to remember the data structures from the last invocation and apply the changes that happen in most recent period to obtain a new value of the metric. In addition to stress, this "delta metric evaluation" technique is applicable for other metrics as well. The savings in computation time using this technique are significant. On a peer-to-peer network with 512 peers, periodic stress computation reduces from roughly 700 seconds to about 10 seconds.

5. Example studies using *p-sim*

In this section, we demonstrate the capabilities of *p-sim* with two studies: one examining the dynamics of peers and its effect on the topology and the other exploring the implications of overlay topology on file search mechanisms. The purpose of these examples is not to present the merits of the proposed protocols, but rather to exhibit the range of features available in *p-sim*.

5.1. Peer Dynamics

We have used *p-sim* to study the effects of peer dynamics, i.e., nodes joining and leaving a peer-to-peer network. In our study, a peer-to-peer node joins the network for a period of time drawn from an exponential distribution. While the node is part of the peer-to-peer network, it "adapts" its position by replacing some of its peers in a sequence of rounds. In each round, the node measures the network latency to its peers. It uses this measure to replace peers with large latency by other closer ones. The node finds these other nodes by examining neighbors of its current peers in each round. Using *p-sim*, we simulated the peer-to-peer network for different numbers of peers and underlying topologies and measured metrics such as connectivity, stress and relative stretch penalty. In Figure 1, we plot the total number of nodes active in the peer-to-peer network as a function of simulation time. The network has a maximum population of 2048 nodes and each peer is active for only one session. The active peer population shows an increasing trend from the start of the simulation to about 400 ms with more arrivals than departures. Between $[400, 1200]$ ms, the population is relatively constant at about 500 peers with equal number of arrivals and departures, while later part of the simulation (> 1200 ms), the overall population decreases slowly. In the next Figure 2, we show the effect of adaptation on the stress of the links in the underlying network during the time frame: $[400, 1200]$ when the population of the network is relatively constant. The four curves correspond to different fractions of the population that employed our proposed adaptation protocol.

5.2. File Search

In another study, we used *p-sim* to simulate file searching on a class of overlay topologies characterized by *proximity-factor*: the fraction of peers that are "close" to a node. We investigated the performance of file searching on different instances of these overlay topologies. For this, we assumed 50 unique files in the network, each with multiple copies in proportion to its popularity. The popularity was assumed to follow a Zipf distribution with 2000 copies of the most popular file. The file copies were stored uniformly at random among all peers. The queries were also assumed to be proportional to the popularity of the files and they were initiated from random nodes. In Figure 3, we plot the success rate of queries for files with different popularity for each instance of overlay topologies. We note that queries for popular files (lower rank) have very high success rate and the success rate drops as the popularity decreases. The variation of distance to the nearest search result node in the underlying network for the four kinds of topologies is plotted in Figure 4. We observe that the popular files are found closer to the source of the query due to the availability of multiple copies. Unpopular files, on the other hand, are rare and found farther away from the query source.

6. Conclusions and Future Work

We have presented *p-sim*, a tool that can simulate the behavior of a large-scale peer-to-peer network. Some of the capabilities of *p-sim* include peer dynamics and file sharing/searching. *p-sim* includes several metrics for evaluating performance of peer-to-peer network protocols. It has a modular design to support the inclusion of future peer-to-peer network protocols as well.

In future, we plan to extend the capabilities of *p-sim* by adding implementations of structured peer-to-peer networks (e.g., Chord [13], CAN [8]). We also plan to distinguish nodes into different types with varying capacities of storage, communication and computation. This heterogeneity of nodes reflects the observations of deployed peer-to-peer networks [11].

References

[1] J. Banks, J. S. Carson, and B. L. Nelson. *Discrete Event System Simulation*. Prentice-Hall, Englewood Cliffs, NJ, 1996.

[2] Y. Chawathe, S. Ratnaswamy, L. Breslau, N. Lanham, and S. Shenker. Making Gnutella-like P2P Systems Scalable. In *ACM SIGCOMM*, 2003.

[3] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang. A Case for End System Multicast. *IEEE Journal on Selected Areas in Communication, Special Issue on Networking Support for Multicast*, 2002.

[4] http://rfc-gnutella.sourceforge.net.

[5] Gnutella network hosts: www.limewire.com/current_size.html.

[6] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An Approach to Universal Topology Generation. In *International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems- MASCOTS*, 2001.

[7] The Network Simulator (ns). http://www.isi.edu/nsnam/ns/.

[8] S. Ratnaswamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content Addressable Network. In *ACM SIGCOMM*, 2001.

[9] G. F. Riley, R. M. Fujimoto, and M. H. Ammar. A Generic Framework for Parallelization of Network Simulations. In *International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems, (MASCOTS)*, 1999.

[10] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the Gnutella Network: Properties of Large Scale Peer-to-Peer Systems and Implications for System Design. *IEEE Journal on Internet Computing, Special Issue on Peer-to-peer Networking*, 2002.

[11] S. Saroiu, K. Gummadi, and S. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Multimedia Conferencing and Networking*, Jan 2002.

[12] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. In *Internet Measurement Workshop*, 2002.

[13] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *ACM SIGCOMM*, 2001.

[14] J. Winich and S. Jamin. Inet-3.0: Internet Topology Generator. Technical Report CSE-TR-456-02, University of Michigan, 2002.

[15] E. Zegura, K. Calvert, and M. J. Donahoo. A Quantitative Comparison of Graph-based Models for Internet Topology. *IEEE/ACM Transactions on Networking*, 5(6), Dec 1997.