IRTF P2PRG
Internet Draft
Expires: July 26, 2006
A. Brown
Stirling
M. Kolberg
Stirling
January 27, 2006

Tools for Peer-to-Peer Network Simulation
draft-irtf-p2prg-core-simulators-00.txt

Abstract

Due to the extremely large scale of P2P overlay networks, complexity has become a major issue. With the number of connected nodes reaching into the millions, a platform which can accurately simulate an overlay network is important. Many freely available network

simulators exist, ranging from modelling networks at the packet level to concentrating purely on the overlay network. The following document provides an overview of these simulators, documenting the maximum number of simulated nodes achieved in experiments, the P2P overlay architectures modelled and the chosen implementation language.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [1].

Table of Contents

1. Introduction

To support the performance work of service overlays to be undertaken by the P2PRG CORE subgroup, we are examining existing simulation tools for suitability. This document summarizes information we have collected to date, covering only freely available simulation tools.

Simulating Peer-to-Peer (P2P) overlay networks is a common problem for researchers/developers. Firstly, P2P overlay networks need to be scalable (Kazaa has more than 4 million simultaneous users) and

creating a simulation for such a large network is difficult due to memory constraints using even the most powerful machines. However, some tools allow a simulation to be distributed over a set of machines.

Additionally, it may be desirable that a simulation behaves in accordance to a physical network (packet delay, traffic and network congestion, bandwidth limitations etc). These considerations also increase the overhead on the host machine. Subsequently, when choosing a network simulation tool for a given network, a distinction must be made between simulation tools which are suitable for low level networks and those which are suitable for P2P overlay networks.

In respect to this decision, this document is split into two further sections. The first section describes simulation tools which are suitable for overlay networks, boasting scalability and small memory footprints. The second section describes some of the heavier network simulators, which boast an array of Internet models and are deemed more suitable for lower level network simulation.

2. Overlay Network Simulators

2.1. Narses

The Narses simulator is a flow-based network simulator designed to avoid the overhead of packet level simulators [2] [3]. Narses is written in Java and achieves high levels of scalability by simulating aggregate flows rather than individual packets. By simulating a chunk of bytes and not individual packets, Narses reduces the number of events and subsequently the amount of memory used in the simulation. To further reduce memory requirements, Narses assumes that transfers between two end users are limited to the bandwidth allocated on the first link to the network. This ensures there is no need for intermediate routers and reassessment of link bandwidth, hence further reducing the number of events in the simulation.

By taking such measures, Narses trades detail of the underlying network for decreased runtimes and a lower memory footprint. Of the models of Narses, the largest is the bandwidth-share model. This TCP like model tries to replicate the sharing of link bandwidth between independent flows. This model was tested in a network consisting of 600 nodes transmitting 200KB flows of data. Narses was reported to be 45 times faster and consumed 28% of the memory of NS2. There is no evidence of existing simulations of overlay networks for Narses.

Download Simulator: http://sourceforge.net/projects/narses/

2.2. 3LS

3LS (3-Level-Simulator) is a Java based simulator for overlay networks that uses a central step clock [3]. The system is separated into three models:

1. Networked

2. Protocol

3. User

The network model is described in terms of distance between nodes in a 2D space. The protocol model is simply the P2P protocol being used. The user model generates input from users using a GUI or file. Communication can only be achieved between directly connected layers.

When running the simulator it is possible to either create the models for each of the three described levels or to select existing models from a library. As the simulation runs, each event is displayed on the screen. Once completed, the simulation data is stored to a file and this data is accessed by a visualisation tool named AiSee [15]. Currently, an implementation of Gnutella 0.4 exists and has been tested on a small network of fewer than 20 nodes.

Download Simulator: Please e-mail the creators Nyik San Ting and Ralph Deters (nyt431@mail.usask.ca, deters@cs.usask.ca) at the University of Saskatchewan for a copy.

Documentation: [3]

2.3. NeuroGrid

NeuroGrid [5][6] is a Java based network simulator designed for comparisons between file sharing P2P systems such as Gnutella, Freenet and other NeuroGrid systems. The NeuroGrid simulator is a single-threaded event simulator and uses a configuration file to define the protocol to simulate and the network properties. These properties include the number of nodes and queries to execute. Statistics including the number of successful queries and messages parsed can be saved into files for later analysis. Currently there are simulations available for Gnutella, NeuroGrid, Freenet and Pastry.

NeruoGrid has been used to simulate networks up to 300,000 nodes on a machine with 4GB of RAM.

Download Simulator: http://sourceforge.net/projects/neurogrid/

Documentation: http://neurogrid.net/php/simulation.php

2.4. PeerSim

PeerSim [7] is written in Java and offers an array of predefined * protocols for P2P simulation. These include:

. OverStat: A collection of aggregation protocols. Aggregation is a collective term for a set of functions aimed at providing statistical information over distributed systems, such as the average load of the nodes or the size of the network.

. SG-1: A protocol for self-organizing and maintaining a superpeer based topology.

. T-Man: This protocol can be used to build a wide range of different topologies using only a ranking function that defines the preference of each node for neighbours.

PeerSim is composed of two simulation engines, one is cycle-based and the other is event driven. This allows it to be both dynamic and scalable. The engines consist of components which may be 'plugged in' and use a simple ASCII file based configuration mechanism which helps reduce the overhead. The cycle-based engine does not model the transport layer and subsequently is more scalable. The event-based engine is less efficient but more realistic. PeerSim can achieve a network consisting of 10^6 nodes using the cycle-based engine.

Download Simulator: http://sourceforge.net/projects/peersim/

Documentation: http://peersim.sourceforge.net/#docs,

http://peersim.sourceforge.net/tutorial1/tutorial1.html

2.5. P2PSim

P2PSim [10] is a written in C++ and designed for the evaluation of P2P protocols. This multi-threaded, discrete event simulator runs in several UNIX-like operating systems. P2PSim already supports Chord, Accordion, Koorde, Kelips, Tapestry, and Kademlia. These implementations are specific to P2PSim and do not model all features

* Description of P2P protocols taken from [7]

of the protocols. P2PSim has been tested with up to 3000 nodes using the Chord implementation.

Download Simulator: http://pdos.csail.mit.edu/p2psim/

3. Packet level Network Simulators

3.1. Omnet++

OMNeT++ [8] is an open-source, component-based simulation environment with a strong focus on supporting the user with a Graphical User Interface (GUI). Although Omnet++ is not strictly a network simulator, it is gaining popularity in that domain. Components (modules) are programmed in C++ and assembled into larger components using a high level language (NED). It is primarily used for simulating communication networks, but can additionally be used for protocol modelling and modelling queuing networks and multiprocessors. Available models include but are not limited to IP, TCP, UDP, PPP, Ethernet, MPLS with LDP and RSVP-TE signalling and 802.11. Currently there exists a P2P swarming simulation for Omnet++ [16], which has been tested on a network of up to 1000 nodes. The overhead of modelling the transport layer in this simulation environment will likely hinder scalability. However, Omnet++ can also run distributed simulations over a number of machines.

Download simulator: http://www.omnetpp.org/filemgmt/

Documentation: http://community.omnest.com/doc/tictoc-tutorial/

http://www.omnetpp.org/doc/manual/usman.html,

http://www.omnetpp.org/doc/api/index.html

3.2. NS2

NS2 [9], developed at UC Berkeley, is a discrete event simulator targeted at networking research and provides substantial support for simulation of many Internet, routing, and multicast protocols. It can be installed on both Windows (using Cygwin) and Linux but evidently is more suited to Linux. NS2 is comprised of many various packages to provide its features (e.g. compatibility with numerous protocols and providing graphical representation of simulations etc) thus is notoriously difficult to use.

NS2 implements a vast array of protocols including TCP and UDP, traffic source behavior such as FTP, Telnet, VBR and CBR. Additionally, there is router queue management mechanisms such as

Drop Tail, RED and CBQ implemented. NS2 also implements multicasting and some of the MAC layer protocols for LAN simulations. Currently, there is only one P2P simulation available for NS2 (Gnutella). Simulations in NS2 are constructed using a mixture of C++ and OTcl (Object Orientated version of Tcl created by MIT).

Simulating in NS2 is a four step model.

1. Firstly, add a combination of C++ and OTcl code to the NS2 standard set of libraries to implement your protocol.

2. Create the simulated network using an OTcl script (define the nodes, links etc).

3. Run the generated scripts.

4. Analyze the trace files generated by NS2. A graphical model (NAM) can use the trace files to show a graphical representation of the simulation. Other tools available include graph plotting.

Due to the realistic nature of the packet level NS2 simulator, scalability is a major issue. However, like most packet level simulators NS2 can run in parallel with a number of other machines. This can increase the maximum number of nodes for a given simulation but can become difficult to manage. Subsequently, NS2 is commonly used for simulating small networks and is generally unsuitable for modelling overlay networks.

Download Simulator: http://www.isi.edu/nsnam/ns/ns-build.html

Documentation: http://www.isi.edu/nsnam/ns/, http://nile.wpi.edu/NS/,

http://www.isi.edu/nsnam/ns/ns-tutorial/index.html

3.3. SSFNet

SSFNet [11] is a collection of Java and C++ based components and uses the Domain Modelling Language (DML) to configure networks as a series of configuration files. The principal classes in the SSF.OS Framework (for modelling of the host and OS components) and the SSF.Net Framework (for modelling the network connectivity) can be used to construct virtually any Internet model. The frameworks SSF.OS and SSF.Net hide all details of the discrete event simulator SSF API, which allows implementation of the protocols similar to a real operating system. This was the chosen simulator for Epichord [14].

SSFNet simulations can be executed independently or in parallel with other simulations running on Linux, Windows or SUN Solaris alike. Regarding maximum network size, a network of 33,000 nodes is achievable (on a powerful machine with 2GB of RAM). SSFNet also has the ability to run a distributed simulation.

Download Simulator: http://www.ssfnet.org/ssfImplementations.html
Documentation: http://www.ssfnet.org/internetPage.html,
http://www.ssfnet.org/SSFdocs/ssfapiManual.pdf

4. Summary

The following tables summarizes the tools.

SIMULATOR	P2P PROTOCOLS	MAX NODES	DISTRIBUTED SIMULATION
Narses	None	600	No
3LS	Gnutella	< 1,000	No
NeuroGrid	Gnutella, NeuroGrid, Pastry, FreeNet	300,000	No
PeerSim	Collection of internally developed P2P models	> 10^6	No
P2PSim	Chord, Accordion, Koorde, Kelips, Tapestry, Kademlia.	3,000	No
Omnet++	None	1,000*	Yes
NS2	Gnutella	N/A	Yes
SSFNet	None	33,000	Yes

Table 1. Simulator Characteristics

Max Nodes is the reported maximum based on information from the simulation developers, and refers to number of simulated nodes per host. This number may be hardware dependent. See the website for the tool for more details.
P2P Protocols are those known to have modeled with the given tool and are models available with the downloaded tool.
For Omnet++, this data was obtained from the P2P Swarming protocol [15]. This does not mean more nodes cannot be achieved using Omnet++.
The following table shows the implementation language.

SIMULATOR	LANGUAGE
Narses	Java
3LS	Java
NeuroGrid	Java
PeerSim	Java
P2PSim	C++
Omnet++	C++/NED
NS2	C++/OTcl
SSFNet	Java/C++/DML

Table 2. Implementation language(s) for each simulator

5. Security Considerations

There are no new security considerations in this document.

6. References

6.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

6.2. Informative References

[2] TJ Guili and Mary Barker, Narses: A scalable flow based network simulator, arXiv:cs.PF/0211024 v1 20 Nov 2002

[3] Narses Simulator, http://sourceforge.net/projects/narses/, viewed January 23, 2006.

[4] Nyik San Ting, Ralph Deters, 3LS - A Peer-to-Peer Network Simulator, Third International Conference on Peer-to-Peer Computing (P2P 2003).

[5] NeuroGrid, http://www.neurogrid.net, viewed January 23rd, 2006

[6] Sam Joseph and T. Hoshiai. Decentralized meta-data strategies: Effective peer-to-peer search. IEICE Transactions on Communications, E86-B(6):1740-1753, 2003.

[7] PeerSim, http://peersim.sourceforge.net, viewed January 23, 2006.

[8] Omnet++, http://www.omnetpp.org, viewed January 23, 2006.

[9] Network Simulator 2, http://www.isi.edu/nsnam/ns/, viewed January 23, 2006.

[10] P2PSim, http://pdos.csail.mit.edu/p2psim/, viewed January 23, 2006.

[11] Scalable Simulation Framework Net, http://www.ssfnet.org, viewed January 23, 2006

[12] Kant, K., Iyer, R.: Modeling and simulation of ad-hoc/p2p file-sharing networks. In proceedings of Tools 2003. (2003)

[13] Alberto Montresor, Gianni Di Caro, Poul E. Heegaard, BISON: Architecture of the Simulation Environment, IST-2001-38923

[14] B. Leong, B. Liskov and E. D. Demaine: EpiChord: Parallelizing the Chord Lookup Algorithm with Reactive Routing State Management, MIT Technical Report, Aug 2004.

[15] AbsInt. AiSee homepage. http://www.aisee.com/, viewed January 24th, 2006

[16] Omnet ++ P2P Swarming Protocol, http://me55enger.net/swarm/, viewed January 24th, 2006.

Author's Addresses

Alan Brown
Department of Computing Science and Mathematics
University of Stirling
Stirling FK9 4LA
Scotland
Email: abr@cs.stir.ac.uk

Mario Kolberg
Room 4B60, Cottrell Building
Department of Computing Science and Mathematics
University of Stirling
Stirling FK9 4LA
Scotland
Email: mkolberg@ieee.org

Intellectual Property Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.