The State of Peer-to-Peer Network Simulators

ANIRBAN BASU, Tokai University and University of Sussex
SIMON FLEMING, JAMES STANIER, STEPHEN NAICKEN, and IAN WAKEMAN, University of Sussex
VIJAY K. GURBANI, Bell Laboratories, Alcatel-Lucent and Illinois Institute of Technology

Networking research often relies on simulation in order to test and evaluate new ideas. An important requirement of this process is that results must be reproducible so that other researchers can replicate, validate, and extend existing work. We look at the landscape of simulators for research in peer-to-peer (P2P) networks by conducting a survey of a combined total of over 280 papers from before and after 2007 (the year of the last survey in this area), and comment on the large quantity of research using bespoke, closed-source simulators. We propose a set of criteria that P2P simulators should meet, and poll the P2P research community for their agreement. We aim to drive the community towards performing their experiments on simulators that allow for others to validate their results.

Categories and Subject Descriptors: I.6.7 [**Simulation and Modelling**]: Simulation Support Systems

General Terms: Design, Experimentation, Measurement, Performance, Standardization

Additional Key Words and Phrases: Peer-to-peer, P2P, simulator evaluation, simulator usage

ACM Reference Format:
Basu, A., Fleming, S., Stanier, J., Naicken, S., Wakeman, I., and Gurbani, V. K. 2013. The state of peer-to-peer network simulators. ACM Comput. Surv. 45, 4, Article 46 (August 2013), 25 pages.
DOI: http://dx.doi.org/10.1145/2501654.2501660

1. INTRODUCTION

Peer-to-peer (P2P) is a distributed computer architecture that facilitates the direct exchange of information and services between individual nodes (called *peers*) rather than relying on a centralized server. In P2P networks, users (i.e., human beings) participate only through their representative agents, that is, software agents running on networked devices.

P2P forms the basis of many distributed computer systems, permitting each peer node to act as both a client and a server, consuming services from other available peers while providing its own service to the rest of the network. Peers within a P2P network engage in direct exchanges with their known neighbors, in order to submit requests and serve responses.

The definition of what specifically constitutes a P2P system is broad. For example, in theory, a P2P system is thought of as having no centralized authority, when in reality many existing P2P applications rely on one. For example, early versions of the

Authors' addresses: A. Basu, Rm. 1603, Tokai University, 2-3-23 Takanawa, Minato-ku, Tokyo 108-8619, Japan; S. Fleming (corresponding author), J. Stanier, S. Naicken, and I. Wakeman, School of Informatics, University of Sussex, Chichester Building, Falmer, Brighton BN1 9QJ, UK; email: Sf58@sussex.ac.uk; V. K. Gurbani, Bell Laboratories and Alcatel-Lucent, 1960 Lucent Lane, Rm. 9C-533 Naperville, IL 60566.

DOI: http://dx.doi.org/10.1145/2501654.2501660

BitTorrent[1] protocol required a “tracker” in order to rendezvous the peers and perform membership management of the swarm (a *swarm* is a collection of peers that are interested in distributing the same content). We find that the following definition by Risson and Moors [2006] is well-suited to classifying P2P systems:

> Peer-to-peer systems are distributed systems consisting of interconnected nodes able to self-organize into network topologies with the purpose of sharing resources such as content, CPU cycles, storage, and bandwidth, capable of adapting to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance, without requiring the intermediation or support of a global centralized server or authority.

P2P offers many advantages. These include scalability, high resource availability, no need for a centralized authority (eliminating a single point of failure) and robustness. However, the consequence of using a P2P architecture is that the quality and usefulness of the services on offer rely entirely on the participating members of the group.

The power of P2P is apparent when considering Metcalfe’s Law [Hendler and Golbeck 2008], which states that the *value* of a network is proportional to the square of the number of connected users. The number of possible connections within a P2P network can be exponential in relation to the number of network nodes, n. All nodes can potentially connect to all other nodes, giving a theoretical maximum number of connections of $n(n-1)/2$; the same number as in a fully connected mesh network. Peers may end up communicating directly, however, multihop routes should exist between any pair of nodes connected to a P2P network through emergent pathways via other peers.

With the enormity of the modern Internet and the efforts to expand it further with the introduction of the Internet Protocol Version 6 (IPv6) [Deering and Hinden 1998], the number of devices that can be explicitly addressed is always increasing. Therefore, P2P will continue to offer the potential for highly scalable and efficient networking technologies in the future on many more devices than just traditional computers, switches, and routers.

Researchers and developers continue to develop new technology and protocols for networks. In order to propose, design, implement, and evaluate new P2P technologies, there is a key problem: how can experiments be performed and monitored in a “real” environment? Due to the complexity and scale of real P2P networks, a popular option is to use computer simulations. A P2P network can be simulated on one or more computers, allowing for the behavior of peers to be monitored precisely, giving the possibility for iterative design, and perhaps most importantly, detailed evaluation. However, as no single simulation platform or set of standards has been agreed upon, many different simulators have been created and used for these purposes in the literature.

In this survey we are interested in looking at the landscape of current P2P simulators. We present the following.

—We give an overview of P2P networking, highlighting key features of such systems. We also look at some of the most well-known P2P applications that are being used currently, and describe the benefits and drawbacks of P2P simulation.
—We present our evaluation criteria for categorizing and evaluating the features of P2P simulators.
—We poll the P2P networking community for their agreement on our proposed criteria, and discuss our findings.

[1]`http://www.bittorrent.com`.

—We perform a survey of over 100 post-2007 papers in order to discover which P2P simulators are being used in the recent literature, and combine these findings with 180 papers from our previous pre-2007 survey.
—Using our findings, we motivate the need for unity and repeatability in the P2P simulation community. We propose that a set of standards should be adhered to which could be based upon our evaluation criteria.

We are also socializing our findings with standards communities, in particular the Internet Engineering Task Force (IETF) and the Internet Research Task Force (IRTF), for a broader dissemination of this information in the form of IETF Request for Comments (RFC) standard documents [Gurbani et al. 2011].

1.1. P2P Networking

In this section we will define our terminology, identifying the key features of P2P systems and the terms with which we shall work.

1.1.1. Neighbors and Connectivity. Each peer typically knows about a small number of other peers in the network. A peer's neighbors are the possible routes for forwarding messages, and they also act as gateways to receive responses from the network. The neighbors of a peer are directly reachable. Routes beyond these neighbors provide connectivity to the entire P2P network as a whole. Peers need to keep track of their neighbors to ensure that they still remain connected to the P2P network. Without neighbors, no routes to or from any single peer exist. As such, peers may need to drop existing neighbors, or request new neighbors, throughout the duration of their P2P session to ensure that they remain connected to the network.

1.1.2. Network Churn. Peers arrive and leave the network at various rates. Current peers are the active population within the network. The action of peers arriving and leaving is called network churn. Churn is induced by "willing" actions, for example, node departures and mobility, and also "unwilling" actions, such as inadvertent failure of a node. Peers will arrive at some point in time and begin to participate within a P2P network, remain for some period of time (called the session duration) while possibly making requests and serving other users, and then eventually leave. They may return at some point in the future. High levels of churn indicate many peers coming and going frequently, while low levels exhibit longer peer session durations with less frequent arrivals and departures.

By nature, different P2P applications will have different levels of churn. "Seeders", or peers that have the entire content, may stay connected to the swarm for a longer period of time, while "leechers", or peers that have portions of the content, may connect with the swarm until the content is downloaded and then leave the swarm. For a swarm to continue serving content, the leechers may have to act altruistically by becoming seeders after acquiring the complete content.

The quality of service in P2P networks may vary under different levels of churn. Therefore, protocol designers need to take into account the anticipated levels of churn to provide an appropriate service for users. For example, in a file sharing service with very high churn, replication of critical files across peers would give a greater chance of them being available.

1.1.3. Bootstrapping. Peers need a means of gaining access, and becoming a participant of, a given P2P network. This process is known as bootstrapping. Often, peers will bootstrap using some kind of centralized resource, commonly called a bootstrapping node or rendezvous host, which gives the unconnected peer an entrance point into the network in the form of a set of active network member addresses, or a list of other centralized repositories to query. Once these addresses have been obtained, the peer can start communicating with the network.

1.1.4. Routing Tables. Peers are typically aware of some or all of the other members of the network. To keep track of these nodes and possible routes, a local collection of peer addresses is managed via a routing table. A routing table may indicate a neighbor's address directly, or alternatively via a next hop in a multihop route, and also associates a cost with each potential path.

1.1.5. Bandwidth and Resources. Bandwidth is a measure of the capacity of a peer has for uploading and downloading data and is often asymmetrical. The amount of bandwidth that each peer has can vary greatly depending on the type of device. A peer could be a dedicated server with a fast broadband connection, or conversely it could be a mobile phone with only a GPRS connection. Resources such as memory, CPU speed, number of cores, and number of sockets or connection blocks that can be maintained are also important contributors. Peers hosted on a well-provisioned host will have more resources at their disposal than peers running on resource-constrained devices such as Personal Digital Assistants (PDAs) or mobile phones.

1.1.6. Network Topologies. The connections between nodes in a network form a graph $G = (V, E)$, such that G consists of the set of vertices V (peers) and the collection of edges in E (links to neighbors). The topology of a P2P network is the structure created by the active connections from peers to other peers. The topology makes use of an underlying physical network topology consisting of the networking devices and physical communication channels and links.

1.2. Example P2P applications and algorithms

There are a number of P2P systems and algorithms that have become popular, with many receiving mainstream attention for both good and bad reasons. Academic systems tend to focus on routing algorithms, while real-world systems facilitate a specific need such as file sharing.

1.2.1. P2P Algorithms. There are two types of P2P networks: structured and unstructured. The primary aim of these networks is to host resources at one or more peers in a network (a storage function) and to allow other peers to find these resources (a routing function) in a distributed space. Generally speaking, the resouces can be viewed as "values" and the search characteristics used to locate the resources can be viewed as "keys". In this light, the function of a P2P network can be specified as a pair of methods: $put(key, value)$ and $value = get(key)$, that is, store a value with an associated key (or set of keys) in the network and retrieve the value given a key at some later time. The value stored in the network represents the resource of interest. The resource could be an image, a media file (movies, songs), books, or the latest Linux kernel. P2P networks typically do not care what is stored in the network as long as there is a some search criteria by which the peers in the network can find the resource.

The difference between structured and unstructured P2P networks is how they approach the task of finding resources. This is where most of the algorithmic research in P2P networks has been focused [Risson and Moors 2006]. Unstructured P2P networks did not impose a strict formation on the peers themselves; a joining peer could simply connect to any existing peer to become part of the P2P network. In structured P2P networks (or overlays), the joining peer is admitted based on an identifier that places the joining peer in relation to a set of existing peers. Thus structured P2P overlays are characterized as forming a Distributed Hash Table (DHT). DHTs are distributed data structures that map a key to a value; each peer is responsible for a certain portion of the namespace where the keys are hashed. The result of this deterministic hashing is that there exists a strict upper bound on the search time for a resource. Comparitively, unstructured P2P networks used blind flooding and random walks to search for the resource. Because the flooding was unbounded, routing costs accrued rapidly. The flooding was costly when the resource was not present in the P2P overlay, but determining the nonexistence of the resource expended considerable computational and network complexity. To remedy this, some structure in the form of supepeers or peer clustering was developed with a centralized index to rapidly find a resource.

Gnutella [Klingberg and Manfredi 2002] was an early protocol that used flooding while Napster evolved to support decentralized content and a centralized index. Early structured algorithms included Pastry [Rowstron and Druschel 2001], Tapestry [Zhao et al. 2001], Chord [Stoica et al. 2001], Content Addressable Network (CAN) [Ratnasamy et al. 2001], and Kademlia [Maymounkov and Mazieres 2002]. While the topological arrangement of these overlays varied—Pastry and Tapestry could be visualized as spanning trees, Chord arranged its peers in a ring, CAN created zones in a torus and assigned peers to these zones, and Kademlia effectively treats nodes as leaves in a binary tree—their biggest benefit was that resources could be located using a fixed number of hops. Furthermore, the state information maintained by peers in order to traverse the network was minimal as well. Each peer maintained a small fixed-length routing table where the number of entries in the routing table was in logarithmic proportion to the number of peers in the P2P network.

1.2.2. P2P Applications. Some of the most well-known P2P systems are given next.

—*Distributed event management.* Astrolabe [Van Renesse et al. 2003] is a DNS-like distributed information management system that monitors the dynamically changing state of a collection of distributed resources, reporting summaries of this information to its users. The updates disseminate on the order of seconds or tens of seconds. Astrolabe achieves robustness through the use of a P2P protocol to communicate between the distributed hosts.

—*P2P Multimedia.* With the nexus of communication moving steadily towards the Internet, decentralized, and sometimes free, voice and video communications have proliferated. In many cases, these systems are composed of P2P networks. Skype[2] is the most visible of such P2P communication networks. RELOAD [Jennings et al. 2012] is another P2P protocol for multimedia communications based on the Session Initiation Protocol (SIP [Rosenberg et al. 2002]). More recently P2P networks have been used to deliver streaming content, both static recorded content as well as live content. Early versions of Joost[3] (2007–2008) used P2P techniques to deliver static content. PPS.tv (PPStream)[4] is P2P video streaming service popular in China. QQlive[5] is another Chinese video P2P streaming technology.

—*Distributed data stores.* The database functions related to large scalable systems such as Facebook and Twitter place more emphasis on search performance, real-time nature, and eventual consistency when compared to enterprise back office functions that use traditional databases where the ACID (Atomicity, Consistency, Isolation, and Durability) properties dominate. Dynamo [DeCandia et al. 2007] and Cassandra [Lakshman and Malik 2010] are the prime examples of such distributed storage systems. Such systems manage very large amounts of loosely structured data spread out across commodity server farms while providing robust service with no single point of failure. P2P techniques such as consistent hashing [Karger et al. 1997]

[2]URL: `http://www.skype.com`.
[3]URL: `http://www.joost.com`.
[4]URL: `http://www.pps.tv`.
[5]URL: `http://live.qq.com`.

are used to partition data across a cluster and replication is used to achieve high availability. The Tahoe Least-Authority Filesystem (Tahoe-LAFS [Wilcox-O'Hearn and Warner 2008]) is an open-source, decentralized, fault-tolerant, and secure P2P distributed data store.

—*Anonymity through P2P.* While we make no judgment on the use of anonymity to access illegal content, we do note that the act of being anonymous has high value in societies. Anonymous sources have a rich tradition in journalism while anonymous dissidents can hide from repercussions while providing insight to a society. There are a number of P2P systems built for anonymity. They typically hide the identity of the participants by special routing techniques. Freenet [Clarke et al. 2000] is a censorsip-resistant distributed file system for anonymous publishing. I2P[6] [Zantout and Haraty 2011] is a decentralized overlay network for strong anonymity and end-to-end encryption.

1.3. Advantages and Disadvantages

P2P networks allow for a distributed system to exist that benefits from the combined resources of the networked group of clients. Connecting together a collection of nodes/peers provides robust services that operate without significant setup costs to one particular organization or individual. As each node in the network operates as both a client and a server, it promotes incentives to participate rather than spectate within the system.

Unlike centralized systems, as the number of P2P clients grows the total resources available to the peer-to-peer network (such as CPU, memory, files, and storage capabilities) available also increases. The perceived growth of total resources along with the growth of nodes echoes the ideas of the grid and cloud computing communities. This flexible growth of resources has often been called "elasticity".

P2P does not suffer a single point of failure as seen in centralized services, as the core functionalities are distributed across the entire network. There can be points of failure, such as BitTorrent network trackers going down, but there is no single failure that disrupts the entire network. There is no centralized authority on P2P networks which gives greater freedom of operation for the network participants, but as ever, such freedom comes at a cost: without administration, malicious code and files may exploit vulnerabilities of nodes in the network.

Since there is no way to authenticate the data being transferred around P2P networks, illegal material may be shared. Also, a resource may not actually be what it claims to be: for example, a fake file server sharing viruses instead of the advertised files.

1.4. P2P Simulation

When developing new networking technology, such as protocols or routing algorithms, researchers and developers need to evaluate the technology with regard to several properties such as its quality, overhead, and robustness. However, real P2P networks may consist of thousands of unique entities and experience unpredictable levels of churn. This makes evaluation of new systems with real users and devices a complicated task. How can this new technology be deployed inexpensively for testing, and on a large enough scale to show useful results? What if something harmful were to happen to one of these devices? Generally, the expense of real user testing is too high to be considered. Furthermore, the difficulty of attracting real users to what is an engineering improvement without any visible user benefits makes P2P simulation an attractive option.

[6]URL: `http://www.i2p2.de`.

A P2P protocol could be implemented on top of a real-world testbed environment such as the global research network PlanetLab [Chun et al. 2003]. Such a testbed consists of over 1000 computers situated at participating sites across the globe[7]. While such a setup provides valuable real-world data in terms of network latency, the disadvantage is that a researcher has to depend on the machines on which an experiment was conducted being constantly available and appropriately configured on PlanetLab in order to reproduce results. This itself is difficult task since the machines are owned by different administrative domains, each with its own support staff and hardware/software upgrade schedule. Therefore, tracking issues and problems across such a large number of real computers is a challenging and daunting task.

Simulation is therefore a valuable tool to develop and validate technology before deployment and exposure to the vagaries of the real world. At their essence, P2P simulators allow for the representation of nodes (which simulate peers) with incoming and outgoing message queues. Over time steps these message queues are processed, passing messages between peer outgoing and incoming queues, simulating the communication in a P2P network. All nodes are aware of their neighbors, and therefore send and receive messages to their neighbors via their message queues.

General-purpose network simulators could and have been used to evaluate P2P systems, even though they are generally tailored to the emulation of low-level networking mechanics. The Network Simulator (ns2 and its successor ns3) [McCanne and Floyd 1999; Henderson 2008], for example, provides a vast range of networking devices and channels to allow accurate and realistic representation of the underlying physical network.

If P2P simulation is able to accurately simulate real P2P networks, it can give the ability to design, implement, and evaluate new technologies in a controlled environment, while allowing for the collection of statistics with greater ease compared to deployment with real users. The importance of P2P simulation is apparent in the literature, with a majority of new P2P developments being validated by implementation in a simulator.

There exist many different P2P simulation environments that all aim to simulate a P2P network by providing an abstracted networking model on which users can build their specific simulation models and behaviors. A timeline of major simulator releases is shown in Figure 1. We will look at these later in the survey. However, as shown previously by Naicken et al. [2007], many P2P developments are evaluated by the use of bespoke P2P simulator technologies, where the simulator is written by the authors of the paper. This decision is interesting considering that many open-source P2P simulators already exist. Three reasons are identified for this: the steep learning curve of P2P simulators; the excess simulator functionality that is not required for many experiments, which may hinder the implementation of experiments and the analysis of results; and where required functionality is absent, modification of the simulator code may prove complex and time consuming.

However, this raises a concern. Since a range of open-source P2P simulators exist, and additionally, since many bespoke simulators are used, verifying and repeating results in the literature is difficult. While we are positive that no researchers would wish to present false or misleading results, the community would benefit from being able to repeat, verify, and build upon new findings [Zhang et al. 2009]. It is perhaps tempting to conjecture whether simulator certification should be a requirement for publication. Certainly, one could run a simulator in parallel with deployment and observe the behavior for small- and medium-scale experiments. Simulation results that match empirical results will provide higher confidence in larger-scale simulations. Unfortunately, as

[7]`http://www.planet-lab.org/`.

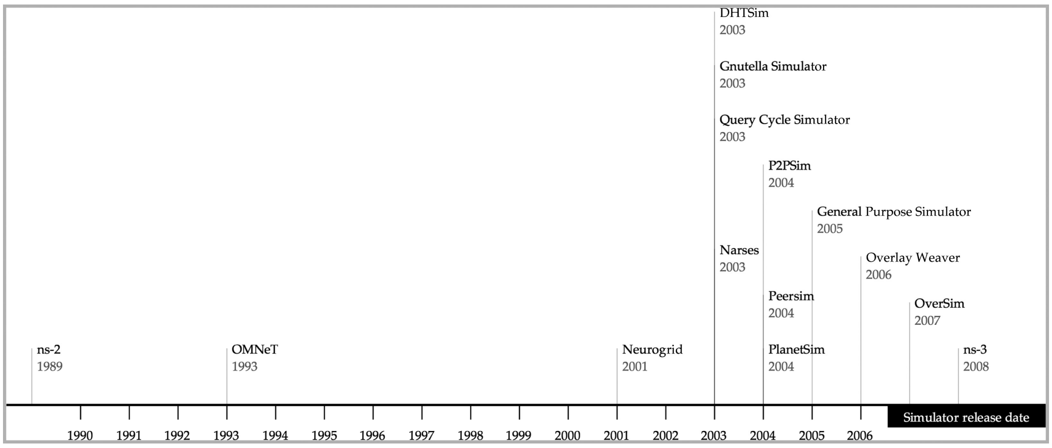


Fig. 1. A timeline of network simulator releases.

attractive as certification sounds, there are drawbacks. The biggest question would be who will be willing to act as an impartial certifying authority? Certifying a simulator's results will require detailed knowledge of the P2P protocol being simulated and the expected behavior of the P2P system. Furthermore, simulations are typically run in order to confirm the behavior of a P2P system if certain changes to the protocol were suggested. Ergo, there will exist a chicken-and-egg problem where certifying the results of a simulation will require the empirical system to support the change that is being simulated. In the end, perhaps the best that can be done is to ensure that the simulation results are repeatable such that a future generation of researchers can rapidly repeat past experiments to have a greater confidence in the simulation that will, in turn, allow them to continue the research by building on the previous work.

1.4.1. Alternatives to Simulation. Even though the main focus of this article is P2P simulators, it is worth noting that there are alternative approaches, such as deployment and development frameworks, as well as networking libraries that allow running P2P protocols both on simulation and on a real network. These include PlanetLab [Chun et al. 2003], SPLAY [Leonini et al. 2009], NEST [Dupuy et al. 1990], and NEKO [Urban et al. 2001]. The system proposed in Wang et al. [2008] allows for trace replay, which can replay a deployed application from trace gathered from simulations. The WIDS toolkit [Lin et al. 2005] achieves higher scalability at the cost of accuracy, by relaxing the synchronization model between nodes.

1.5. Impact of Previous Survey of P2P Simulators

In our previous survey of P2P simulators [Naicken et al. 2007], conducted in 2007, we postulated that the significant use of bespoke simulators in academic research was motivated by poor or missing functionality in existing P2P simulations. It was hoped that the findings of the survey would lead to closer collaboration between the designers of P2P simulators and academics conducting P2P research. A survey of papers that cite this work shows that the opposite has occurred and that there has been further fragmentation in the domain of P2P simulators.

Much of the research that cited the 2007 paper did so in order to justify the use of a P2P simulator, be it custom or otherwise, however, there were also contributions to address some of the issues that we raised. Efforts were concentrated on both the development of new simulators and the addressing of issues, such as the transition of simulation code to deployment and APIs.

Rather than collaborate to work on a single simulators, a number of genera-purpose simulators were created, such as DEUS [Amoretti et al. 2009], OverSim [Baumgart et al. 2007], and PeerfactSim.KOM [Stingl et al. 2011]. The ProtoPeer framework allows for the transition of simulator code to the network, an issue we previously highlighted, while the authors of the P2PAM framework [Agosti et al. 2008] adopt an approach that we expected to be more common, which is the modification of existing simulators to improve functionality.

Juxtaposed to the development of general-purpose simulators are application-specific frameworks that have been developed post-2007, in particular for BitTorrent [Barcellos et al. 2008; De Vogeleer et al. 2008]. This shows that there may be two contradictory schools of though: either P2P simulators can address different use cases or that they can not. If the latter is true it would go some way to explaining the popularity of custom simulators.

The main concern of previous and the current work is the repeatability of experiments and mechanisms to facilitate their reproducibility. A common API for simulators to allow for the repeatability of experiments across simulators was proposed by Gross et al. [2010].

In hindsight, it was somewhat naive of us to assume that the P2P academic community would pool their efforts into the design and use of a single P2P simulator, as we did not work alongside a body that would undertake the responsibility for leading such efforts. To ensure that efforts continue beyond the publication of this survey, we have worked closely alongside the IRTF P2P group. This will ensure that the research community will be able to coordinate a collaborative effort to ensure the principal goal, which is the repeatability of academic P2P experiments, is met.

2. EVALUATION CRITERIA

The abundance of P2P algorithms and related applications requires a wide range of features from a simulator in order to perform realistic simulations. The purpose of a peer-to-peer simulator is to provide the building blocks for P2P researchers to implement and test new protocols. In order to evaluate the existing P2P simulators, we construct a set of criteria which we would expect to be met by all simulators in order for them to be accepted and used widely. Further to that, we also present a set of proposals, most of which we envision should be considered when designing and developing new P2P simulation tools. The criteria and the set of proposals are developed and extended from the evaluation criteria presented in our previous works [Naicken et al. 2007, 2006b; 2006a] as well as opinions gathered by conducting user surveys.

2.1. Evaluation Criteria

Simulation architecture and features. (1) There are two main classes of simulators: query-cycle and discrete event. A query-cycle simulation loops through each node in the overlay network, and carries out queries for networks as and when required. In contrast, a discrete event simulator maintains a thread scheduler that synchronizes a queue of messages to be transferred between the simulated nodes of the overlay network. A P2P simulator should be either discrete event or query-cycle, or enable both modes of operation. The classical trade-off in simulation involves sacrificing the level of detail of a simulation in order to gain speed and scalability (in terms of number of hosts being simulated). Session-level simulators also exist, however, they do not simulate packet-level events, but rather consider a flow, or a session as the important attribute of a simulation. This allows for simulation of larger networks, albeit at a lower level of detail. (2) The simulator should enable specification of event input files, either as a complete list of predetermined events or events that are dynamically generated may generate other causal events; (3) random number generation seeding should allow for repeatability in the simulator; (4) it should support modeling of realistic transport protocols, for example, message queueing; (5) the simulator should not present any bootstrapping problems[8] to simulation experiments; (6) it should be written in a well-known programming language.

Underlying network simulation. While P2P networks are overlay networks, many a time P2P researchers are interested in testing their protocols with different implementations of the underlying network, for example: (1) topology generation, such as random, circular; (2) network latency and bandwidth including any variation between any pair of nodes and congestion between nodes; (3) different types of access links, for example, Ethernet, WiFi, 3G, LTE; (4) the ability to model a node's computational capabilities in terms of processing power and storage specified in compute units, as is done in Amazon EC2[9].

Behavior. P2P overlay networks often need to model the behavior of individual nodes in terms of features, for example: (1) misbehavior (intentional or unintentional); (2) message loss; (3) mobility models, such as the mobile waypoint model; (4) churn with parameterized distribution models; and (5) node failure.

Statistics. The ability to generate statistical data about a simulator run is an essential feature for a P2P simulator. Certain properties of such statistics generation are to be expected, for example: (1) support for output formats, such as SQL, XML, text; (2) mechanisms for logging experiments; (3) network graph properties, such as HITTS, PageRank, power law, centrality, betweenness; and (4) support for GraphML (a versatile graph file format) or similar.

Usability. One of the most important criterion that P2P simulators should meet is usability. Despite having a host of features, many simulators do not get used simply because of their learning curves or unavailability of community support. The simulators should: (1) adhere to standard programming interfaces, such as the Common API [Dabek et al. 2003] when simulating DHTs; (2) provide clear documentation; (3) have user community support through forums or a mailing list, amongst others; (4) have a reference implementation of wel-known P2P protocols, for example, Chord, Kad network (which is a Kademlia-based DHT and implemented by eMule [Kulbak and Bickson 2005], among other applications), CAN, Tapestry, and so on; and (5) have a GUI for visualization of various aspects of the simulation.

2.2. Recommendations

We make the following generic recommendations in view of improving existing and new simulation platforms.

Efficiency and scalability. In order to evaluate how P2P algorithms will perform in large-scale networks, it is essential to be able to simulate large networks. Often, this is restricted by simulator bottlenecks arising from the computational power of single machines. Ideally, simulators should make use of multicore, multiprocessor machines through the use of multithreading. To grow out of the boundary of a group of CPU cores connected through a high-speed memory bus, further concurrency can be achieved if simulators scale to distributed setups, such as a cluster. Network latencies affecting delays in such distributed simulation will not affect discrete event simulations because of the discrete time scale. However, on a networked setup, it is also harder to maintain a unified clock. We envision that distributed P2P simulation is a research area in itself

[8]To simulate a dynamic network requires the ability to acquire bootstrap nodes, such as uniformly at random (random graph), from a distinct set of nodes (supernodes) or possibly a single node (central control). Without such facilities, these will need to be manually designed and implemented.

[9]`http://aws.amazon.com/ec2/`.

with a number of unsolved challenges. For example, there is research on concurrency using distributed Java Virtual Machines [Zhu et al. 2002] that spawn over multiple nodes in a network allowing Java programs to see a globally addressable memory space and computing cores larger than one node. On the other hand, Dean and Ghemawat [2008] proposed the generalized MapReduce concurrent computational paradigm, which we envision can also be used to run discrete event simulations now that robust MapReduce libraries such as Apache Hadoop are freely available. In addition, cloud computing also provides substantial nonvolatile storage space and reasonable computational power, which may be useful in some simulation scenarios. Massive parallelism can also be obtained by delegating parallelizable tasks to a Graphics Processing Unit (GPU). For example, the nVidia Tesla "Fermi" series "desktop supercomputer"[10] platform allows virtual addressable memory space spreading across more than one GPU chipset in a distributed fashion.

Interoperability with an emulator. While complicated scenarios can be simulated, many researchers also opt for emulation on testbeds such as PlanetLab and EmuLab. Emulators allow capturing unforeseen network behavior that cannot always be modeled in a simulator. The ability to port simulation scenario code from a simulator to an emulator will help researchers obtain faster results from their evaluation prototypes. New simulators should consider interoperability of simulation code with emulators, such that researchers can deploy emulation scenarios from their existing simulation scenarios with minimal or no additional coding.

Realistic simulation. One of the fundamental objectives of simulation is to model realistic scenarios as closely as possible. This means that simulators should consider realistic network conditions. They should also simulate different capabilities of network nodes in terms of their computational power, device types, and connectivity.

Flexibility. One aspect of flexibility in a simulator platform is node (or peer) scalability as discussed earlier. Beyond scalability, a simulator framework should also exhibit flexibility in allowing transient state of a simulation to be saved and subsequently resurrected so that long-running simulations can be stopped and restarted at a later point in time, or stopped and migrated to a completely different host. A simulation framework should also allow trace-driven simulation for easy replay of events captured in a log file. This allows real-world dynamics to be replayed in the simulator.

3. PAPER REVIEW

Continuing in the same vein as our previous survey [Naicken et al. 2007] (Figure 2), we aimed to evaluate 200 P2P-related published papers. We took the first 100 search results from Google Scholar[11] for the search terms "peer-to-peer" and "P2P", sorted by number of citations, in order to locate research using P2P simulation. Our previous paper used the same methodology to survey papers at the time of writing (2007) and before. To continue from that work, and to provide a snapshot of the current state of P2P simulators, we only considered papers that were published since 2007. From the search results, we only considered papers we were able to access in an electronic format through our institution's subscriptions on the day of evaluation. Out of the 200 papers we found, 125 were available to download. Papers which used no implementations (e.g., theoretical papers), emulators, and real test platforms (such as PlanetLab [Chun et al. 2003]), or where no simulator was mentioned, are omitted from our results (Figure 3).

[10]`http://www.nvidia.com/object/tesla_computing_solutions.html`.
[11]`http://scholar.google.co.uk`.

Simulator	Papers
Custom	43
ns-2	8
Chord (SFS)	7
Javasim	2
PeerSim	2
Aurora	1
CSIM 19	1
ModelNet	1
Nab	1
Narses	1
NeuroGrid	1
P2PSim	1
SOSS	1

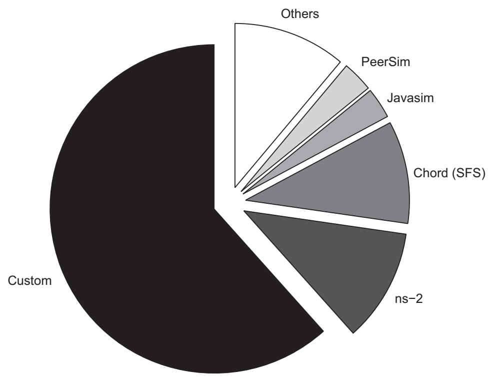


Fig. 2. Simulator usage pre-2007.

Simulator	Papers
Custom	47 (Fig 4)
ns-2	7 (Fig 5)
PeerSim	4 (Fig 6)
SALSA	2 (Fig 7)
P2PSim	1 [Ghinita et al. 2007]
OMNeT++	1 [Awad et al. 2008]
NGS	1 [Webb and Soh 2007]
NEW	1 [Chen et al. 2008]
GnuSim	1 [Karakaya et al. 2008]

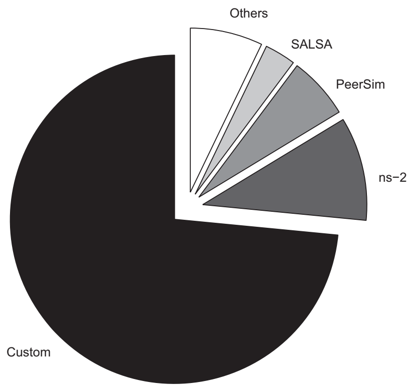


Fig. 3. Simulator usage post-2007.

The most notable result from both time periods is that custom simulators are predominantly used; ns-2 is the second most used simulator in both sets of results. It is worth mentioning that ns-2 is not a P2P simulator; it is a general network simulator. Therefore, a P2P simulation overlay needs to be written for it, and in most, if not all, cases this is a custom implementation (we found no specific mentions of openly available ns-2 P2P overlays). The usage of other open-source simulators is more fragmented. Clearly, it can be seen that the research community prefers to write their own simulator software rather than using existing simulators. The consequence of this fact is that the repeatability of results, and the ability to build upon existing findings, is poor.

4. FEATURES QUESTIONNAIRE

We polled the P2P community to see how useful our criteria would be in a potential P2P simulator. We gathered information from two groups: the authors of the papers that we surveyed earlier, and also the users and developers of existing simulators. We prepared a questionnaire using the Likert scale [Likert 1932] to judge the importance of our criteria, and also gave free text areas to gain insight into individual users' requirements and concerns. The Likert scale is used to measure positiveness or negativeness of response to a question. The scale's response categories are as follows: (1) strongly

[Annapureddy et al. 2007b; Zhou et al. 2007; Yunhao et al. 2007; Xie et al. 2008; Mol et al. 2008; Silva et al. 2008; Iliofotou et al. 2009; Pianese et al. 2007; Annapureddy et al. 2007a; Leonardi et al. 2008; Doulkeridis et al. 2007a; Cheng et al. 2007; Chi et al. 2007; Rowaihy et al. 2007; Ahmed and Boutaba 2007; Belenkiy et al. 2007; Kantere et al. 2009; Huang et al. 2007b; Garbacki et al. 2007; Buchegger et al. 2009; Crainiceanu et al. 2007; Zhou and Hwang 2007; Bharambe et al. 2008; Terpstra et al. 2007; Zaharia and Keshav 2008; Wanget al. 2007; Guo et al. 2007; Hefeeda and Saleh 2008; Ren et al. 2008; Zhou et al. 2008; Zhuge and Li 2007; Haridasan and van Renesse 2008; Feng and Li 2008; Magharei and Rejaie 2009a; Liu 2007; Yang and Chen 2008; Do et al. 2008; Do et al. 2009; Boufkhad et al. 2008; Duminuco and Biersack 2008; Yiu et al. 2007; Doulkeridis et al. 2007b; Skobeltsyn et al. 2009;Johansson et al. 2007; Huebsch 2008; Liu et al. 2008; Mol et al. 2007]

Fig. 4. Papers using custom P2P simulators post-2007.

[Mavlankar et al. 2008; Magharei and Rejaie 2009b; Taninet al. 2007; Eger et al. 2007; Huang et al. 2007a; Setton et al. 2008]

Fig. 5. Papers using ns-2 P2P simulators post-2007.

[Laredo et al. 2008; Jelasity et al. 2007; Baldoni et al. 2007; Silverston et al. 2008]

Fig. 6. Papers using PeerSim P2P simulators post-2007.

disagree; (2) disagree; (3) neither agree nor disagree; (4) agree; and (5) strongly agree. The questionnaire was sent by email to the authors of our surveyed papers, and also to several active P2P simulator-related mailing lists[12]. Although the inclusion of the P2P simulator-related mailing lists could have introduced bias in the sample, we argue that this is countered by the need to ensure that end-users of P2P simulators were included in the sample and to maximize response to the questionnaire We received a total of 81 responses, 45 of which were from the paper authors, and 36 from mailing list users. We now present the results of the questionnaire, organized into the categories in which they were presented. In each section we provide a list of the statements we provided, along with a reference to our results. In each histogram that we present, the x-axis is labeled according to the response given. The y-axis represents the number of responses we received. Typically users were asked to rate questions according to how important they believe the feature to be. On the scale, 1 means not at all important, 3 is neutral, and 5 means essential.

4.1. Simulation Architecture and Features

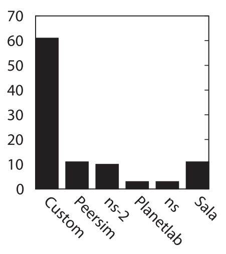
This section encompasses the design of the simulator and the features it should simulate. These questions represent what the high-level features of a P2P simulator should be. The questions were the following.

—How important is the ability to simulate in both query-cycle and discrete event manners? (Figure 8(a))
—What language should the simulator be written in? (Figure 8(b))
—How many P2P nodes would you expect to simulate? (Figure 8(c))
—How important is the option of distributing the load of the simulator over multiple computers? (Figure 8(d))

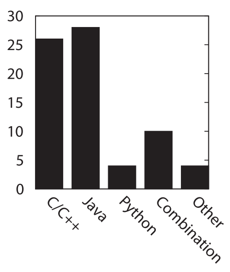
[12]These lists were: p2p-hackers, OverSim, PeerSim, ns, PlanetSim and PlanetLab.

[Mittal and Borisov 2008; 2009]

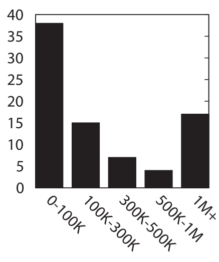
Fig. 7. Papers using SALSA P2P simulators post-2007.



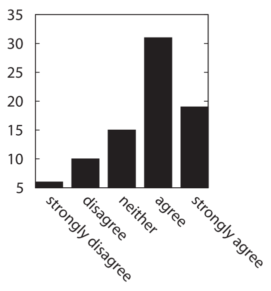
(a) query-cycle and discrete event



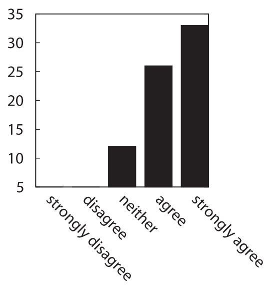
(b) preferred implementation language



(c) required number of nodes



(d) support distributed computation



(e) realistic underlying network emulation

Fig. 8. Simulation architecture and feature responses.

—How important is it that the simulator models the low-level details of the underlying network? (Figure 8(e))

There was a trend towards having the option of simulating using either query-cycle or discrete event simulators (Figure 8(a)). The most popular languages for the implementation of the simulator were C/C++ and Java (Figure 8(b)), possibly due to the ubiquity of the languages in the community and also in teaching. We also allowed for free text responses, and we received two additional responses: one for Erlang and one for Lua. Generally speaking, our respondents preferred high-level imperative programming languages for implementation. Our respondents strongly preferred the smallest range of nodes for simulations (Figure 8(c)). This suggests that 0–100,000 nodes is sufficient for most P2P simulations. It is worth noting that 15 respondents are interested in P2P networks consisting of nodes in excess of 1M. Simulation is a computationally expensive activity. Thus, the need to distribute computation was also agreed upon (Figure 8(d)). This is especially important to those respondents interested in networks in excess of 1M nodes. There is also a strong trend towards wanting to realistically simulate the underlying network layer (Figure 8(e)). We hypothesize that this would give more credibility to any simulator results, as they would be similar to those that would be experienced on real networks.

4.2. Statistics and Simulation Run

This section considers the generation and collection of statistics from simulation runs. Current P2P simulators offer varying degrees of data logging in a variety of formats. We wanted to find out more detail about this element of P2P simulators. The questions were as follows.

—How important is it to generate statistics for a given run? (Figure 9(a))
—Should different output formats be supported, such as xgraph, MATLAB, etc.? (Figure 9(b))
—Should detailed log information be generated for every simulator run? (Figure 9(c))
—Should it be possible to pause the simulation and inspect the current state at any time? (Figure 9(d))
—How important is a specific simulator debugger that allows breakpoints and step-based simulation? (Figure 9(e))

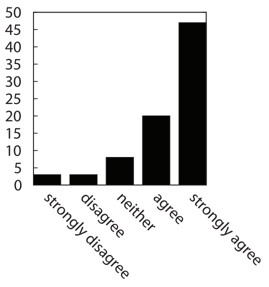
Generally speaking, any simulation aims to generate results in some form. Therefore, it is not surprising that Figure 9 shows strong positive responses for all questions. The questions related to pausing the simulation (Figure 9(d)) and debugging (Figure 9(e)) show a clear positive response, albeit not as strongly correlated as the others. Clearly, any tools that can assist the developer in locating and solving problems are welcome additions to any simulator. It is worth noting that a vast range of statistical functions may be desirable in order to analyze results, and a simulator may facilitate this by incorporating a statistical package such as SSPS[13] or open-source libraries that implement the same functionality.

4.3. Usability

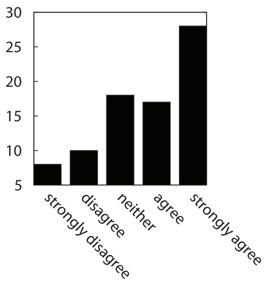
The easier it is to implement an algorithm or scenario on a P2P simulator, the better. This leads to faster implementation and dissemination of results, and crucially, the minimization of errors. Therefore we were interested in what features determine good usability. The questions were the following.

—How important is community support in the form of mailing lists, newsgroups, and forums, for example? (Figure 10(a))
—How important is a well-documented simulator API? (Figure 10(b))
—How important is a specific declarative scripting language to generate overlays and events? (Figure 10(c))

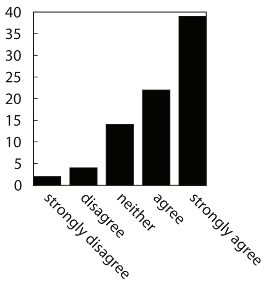
[13]`http://www.spss.com`.



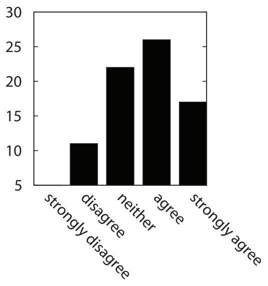
(a) ability to generate statistics



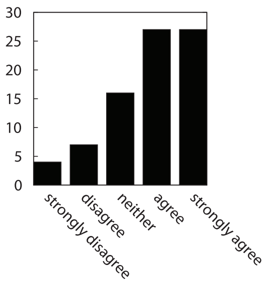
(b) various output formats



(c) simulation log information



(d) ability to pause simulation

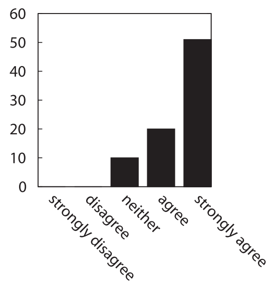


(e) debugging tools and aids

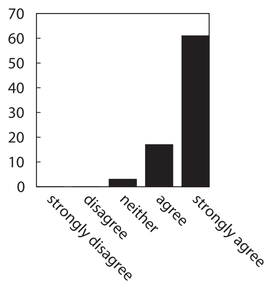
Fig. 9. Statistics and simulation run responses.

—How important are graphical visualizations during or after simulator runs? (Figure 10(d))

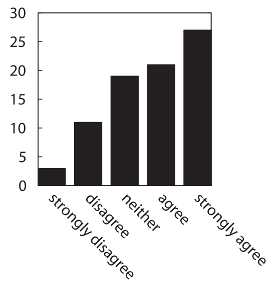
Generally speaking, the more features available to increase usability, the better the reception of the simulator. Access to a well-documented API (Figure 10(b)) and also the use of a specific language (Figure 10(c)) are paramount. Community support is also favored (Figure 10(a)) as an active community can help answer questions and also fix bugs in the simulator. Interestingly, the ability to visualize simulations shows uncertainty (Figure 10(d)). This is possibly because visualizing very large networks is unfeasible due to the vast quantity of nodes and edges. In summary, visualization



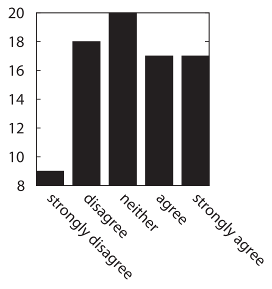
(a) community features and support



(b) API access



(c) a declarative scripting language



(d) simulation visualizations

Fig. 10. Usability responses.

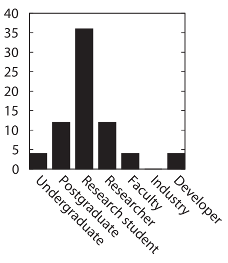


Fig. 11. Profession of questionnaire respondents.

may be a good debugging and demonstration tool and not necessarily a tool to view the entire simulated network.

4.4. Miscellaneous

In addition to the questions cited before, we asked each respondent his/her profession. As seen in Figure 11, the overwhelming majority of respondents were research students. This may suggest that research students are the main workforce behind P2P simulator development. No industry researchers responded to the poll, however, this could be because they are not authoring papers or subscribed to the mailing lists we chose.

Table I. Criteria Present in Simulators

	Simulator					
Criteria	P2PSim	PeerSim	PlanetSim	OverSim	ns	SimGrid
Simulation architecture and features.						
1. Query-cycle and discrete event	DE	DE, QC	DE	DE	DE	DE
2. Scripted Events	–	–	–	●	–	–
3. Random number seeding	●	●	–	●	●	●
4. Realistic transport layer	–	●	–	●	●	–
5. Bootstrapping features	–	–	●	●	–	–
6. Implementation language	C++	Java	Java	C++	C++, Python	C, C++ Ruby, Lua
Underlying network simulation.						
7. Topology generator	●	●	●	●	–	–
8. Latency and bandwidth	●	●	–	●	●	●
9. Various access links	–	–	–	●	●	●
11. Computation ability modelling	–	–	–	–	–	–
Behaviour.						
12. Misbehaviour	–	–	●	●	●	●
13. Message loss	●	●	●	●	●	●
14. Mobility models	–	–	–	●	●	●
15. Churn	●	●	–	●	●	●
16. Node failure	●	●	●	●	●	●
Statistics.						
17. Output formats	T	T, G, O	T, G, O	T	T, P	T
18. Logging	●	●	●	●	●	●
19. Graph properties	–	●	–	–	–	–
20. GraphML	–	●	●	–	–	–
Usability.						
21. Programming interfaces	–	–	●	●	●	●
22. Documentation	–	●	●	●	●	●
23. Community	●	●	●	●	●	●
24. Reference implementations	●	●	●	●	●	●
25. GUI	–	–	–	●	–	–
– information not known or not readily available DE = discrete-event, QC = query-cycle T = text, G = gml, O = other, X = XML, P = pcap						

5. SIMULATOR FEATURE COMPARISON

Using the simulator feature criteria discussed in Section 2.1 and compiled in Figures 8, 9, and 10 and augmented with the data from our paper reviews, we present a comparison table of six simulators—P2PSim, PeerSim, PlanetSim, OverSim, ns (ns-2 and ns-3), and SimGrid. We note that the list of these simulators in not authoritative and neither is it driven by popularity counts as measured by the number of papers referring to the said simulator. Rather, this list is driven by information gathered from our questionnaire and the aggregate personal experiences from using some of the simulators in a P2P context. Thus, for instance, while Figure 8(a) indicates PlanetLab as a simulator, we do not include it in Table I because it is not a simulator as much as it is a reasonable substitute for observing program and code behavior on a real network. Similarly, while

Figure 8(a) did not include any mention of SimGrid [Casanova et al. 2008] (see also http://simgrid.gforge.inria.fr/), we include it in our list since it is actively supported and in use.

The six simulators along with an evaluation criteria discussed in Section 2.1 are shown in Table I. Some inferences can easily be drawn from the data in Table I, although we stress that these are only inferences and not authoritative results. Most simulators appear to favor discrete event simulation; Java and C/C++ appear to be widely supported. SimGrid supports both of these languages in addition to Ruby and Lua bindings. All simulators support logging as well as some manner of textual output of statistics. None of the simulators listed in the table allows creating topologies using drag- and-drop palettes, though some of the simulators allow for a visualization component once the topology is set up and the simulation is running (although this visuzalization is rather basic).

Furthermore, while the simulation frameworks specifically designed for P2P networks (such as PeerSim and OverSim) inherently contain behavior specific to overlay networks—churn, node misbehavior, node failure, etc.—general-purpose simulators such as NS-3 and SimGrid also allow such behavior to be specified by the application programmer in the specific P2P behavior model. And finally, although no P2P simulator contains all of the features in our proposed criteria, it can be seen than OverSim contains a majority of them. Older simulators such as P2PSim do not match the increased requirements that the community expects presently. It is also instructive to note that general-purpose simulation frameworks such as NS-3 and SimGrid satisfy many of our evaluation criteria features. However, the implicit understanding here is that programmers using these simulation frameworks may have to spend extra time fine-tuning the application model to allow it to behave in a certain fashion instead of having a parametrized application model whose behavior can be changed through the parameters.

6. CONCLUSION

We have presented an in-depth review of P2P papers investigating the use of simulators and simulations in this research.

By examining our new set of papers, since 2007 we have found a greater proportion of research being tested with real systems and real trace data. The P2P community appears to be focusing less on algorithms and more on network software optimizations such as streaming video (perhaps motivated by the inevitability of Internet Television).

An overwhelming body of work relies heavily on simulations to produce results and comparisons. The most alarming fact about our findings is that a large majority of highly cited P2P research papers produce their results on custom written simulators. These simulators are not publicly available. Most importantly this does not allow the research community to verify the results of these papers by repeating experiments. It also hinders the overall development of simulators and tools because others are not able to build upon existing work. This has a profound effect on the time taken for new research to emerge, as many academics and developers alike have to reproduce existing code that is not available, despite the fact that a wide variety of open-source P2P simulators already exist. Prior to 2007, this issue was identified. The problem still persists today in similar proportions. Additionally, even though new simulators continue to be released, the research community tends towards only a handful of open-source simulators. The demand for features in simulators, as shown by our criteria and survey, is high. Therefore, the community should work together to get these features in open-source software. This would reduce the need for custom simulators, and hence increase repeatability and reputability of experiments.

One outlet to socialize the need for common simulation frameworks is the Internet Research Task Force (IRTF), a sister organization to the Internet Engineering Task Force (IETF). IRTF meetings are colocated with IETF meetings, which are held three times a year. The locations of IETF meetings vary, but to facilitate the reach of Internet as widely as possible, IETF typically arranges meetings in Europe, North America, and Asia during the calendar year. The attendees to IETF meetings include both academia and industry participants. Given the composition of the IETF attendees and the reach of the organization to all countries, we believe that socializing the need for a common simulation framework in IETF will have the most impact. To that extent, we have started working with the IETF community to distribute the results from this survey to the IRTF Peer-to-Peer Research Group (P2PRG). We will continue working with the IRTF P2PRG to socialize, disseminate, and publish the results.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their time, feedback, and a detailed reading of this manuscript. Their reviews have contributed greatly to the quality of the manuscript. The authors would like to thank Barney Livingston, Sethalat Rodhetbhai and Dan Chalmers for their inputs on an earlier related work.

REFERENCES

Agosti, M., Zanichelli, F., Amoretti, M., and Conte, G. 2008. P2pam: A framework for peer-to-peer architectural modeling based on peersim. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems and Workshops (Simutools'08).* Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, 22:1–22:10.

Ahmed, R. and Boutaba, R. 2007. Distributed pattern matching: A key to flexible and efficient P2P search. *IEEE J. Selected Areas Comm. 25,* 1, 73–83.

Amoretti, M., Agosti, M., and Zanichelli, F. 2009. Deus: A discrete event universal simulator. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques (Simutools'09).* Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, 58:1–58:9.

Annapureddy, S., Guha, S., Gkantsidis, C., Gunawardena, D., and Rodriguez, P. 2007a. Exploring vod in p2p swarming systems. In *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM'07).* IEEE, 2571–2575.

Annapureddy, S., Guha, S., Gkantsidis, C., Gunawardena, D., and Rodriguez, P. R. 2007b. Is high quality vod feasible using p2p swarming? In *Proceedings of the 16th International Conference on World Wide Web (WWW'07).* ACM Press, New York, 903–912.

Awad, A., German, R., and Dressler, F. 2008. P2P-based routing and data management using the virtual cord protocol (vcp). In *Proceedings of the 9th ACM International Symposium on Mobile Ad-Hoc Networking and Computing (MobiHoc'08).* ACM Press, New York, 443–444.

Baldoni, R., Beraldi, R., Quema, V., Querzoni, L., and Tucci-Piergiovanni, S. 2007. TERA: Topicbased event routing for peer-to-peer architectures. In *Proceedings of the Inaugural International Conference on Distributed Event-Based Systems (DEBS'07).* ACM Press, 2–13.

Barcellos, M., Mansilha, R., and Brasileiro, F. 2008. Torrentlab: Investigating bittorrent through simulation and live experiments. In *Proceedings of the IEEE Symposium on Computers and Communications (ISCC'08).* IEEE, 507–512.

Baumgart, I., Heep, B., and Krause, S. 2007. OverSim: A flexible overlay network simulation framework. In *Proceedings of the IEEE Global Internet Symposium.* IEEE, 79–84.

Belenkiy, M., Chase, M., Erway, C. C., Jannotti, J., Kupc U. A., Lysyanskaya, A., and Rachlin, E. 2007. Making p2pP accountable without losing privacy. In *Proceedings of the ACM Workshop on Privacy in Electronic Society (WPES'07).* ACM Press, New York, 31–40.

Bharambe, A., Douceur, J. R., Lorch, J. R., Moscibroda, T., Pang, J., Seshan, S., and Zhuang, X. 2008. Donnybrook: Enabling large-scale, high-speed, peer-to-peer games. In *Proceedings of the ACM SIGCOMM Conference on Data Communication (SIGCOMM'08).* ACM Press, New York, 389–400.

Boufkhad, Y., Mathieu, F., De Montgolfier, F., Perino, D., and Viennot, L. 2008. Achievable catalog size in peer-to-peer video-on-demand systems. In *Proceedings of the 7th International Conference on Peer-to-Peer Systems (IPTPS'08).* USENIX Association, 4–10.

BUCHEGGER, S., SCHIOBERG, D., VU, L.-H., AND DATTA, A. 2009. PeerSoN: P2p social networking: Early experiences and insights. In *Proceedings of the 2nd ACM EuroSys Workshop on Social Network Systems (SNS'09).* ACM Press, New York, 46–52.

CASANOVA, H., LEGRAND, A., AND QUINSON, M. 2008. SimGrid: A generic framework for large-scale distributed experiments. In *Proceedings of the 10th IEEE International Conference on Computer Modeling and Simulation*.

CHEN, M., PONEC, M., SENGUPTA, S., LI, J., AND CHOU, P. A. 2008. Utility maximization in peer-to-peer systems. In *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'08).* ACM Press, New York, 169–180.

CHENG, B., JIN, H., AND LIAO, X. 2007. Supporting vcr functions in p2p vod services using ring-assisted overlays. In *Proceedings of the IEEE International Conference on Communications (ICC'07).* IEEE, 1698–1703.

CHI, H., ZHANG, Q., JIA, J., AND SHEN, X. 2007. Efficient search and scheduling in p2pP-based media-ondemand streaming service. *IEEE J. Selected Areas Comm. 25,* 1, 119–130.

CHUN, B., CULLER, D., ROSCOE, T., BAVIER, A., PETERSON, L., WAWRZONIAK, M., AND BOWMAN, M. 2003. PlanetLab: An overlay testbed for broad-coverage services. *ACM SIGCOMM Comput. Comm. Rev. 33,* 3, 3–12.

CLARKE, I., SANDBERG, O., WILEY, B., AND HONG, T. W. 2000. Freenet: A distributed anonymous information storage and retrieval system. In *Proceedings of the International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability.* Springer, 46–66.

CRAINICEANU, A., LINGA, P., MACHANAVAJJHALA, A., GEHRKE, J., AND SHANMUGASUNDARAM, J. 2007. P-ring: An efficient and robust p2p range index structure. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'07).* ACM Press, New York, 223–234.

DABEK, F., ZHAO, B., DRUSCHEL, P., KUBIATOWICZ, J., AND STOICA, I. 2003. Towards a common api for structured peer-to-peer overlays. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03).* 33–44.

DE VOGELEER, K., ERMAN, D., AND POPESCU, A. 2008. Simulating bittorrent. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems and Workshops (Simutools'08).* Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, 2:1–2:7.

DEAN, J. AND GHEMAWAT, S. 2008. MapReduce: Simplified data processing on large clusters. *Comm. ACM 51,* 107–113.

DECANDIA, G., HASTORUN, D., JAMPANI, M., KAKULAPATI, G., LAKSHMAN, A., PILCHIN, A., SIVASUBRAMANIAN, S., VOSSHALL, P., AND VOGELS, W. 2007. Dynamo: Amazon's highly available key-value store. *ACM SIGOPS Oper. Syst. Rev. 41,* 6, 205–220.

DEERING, S. AND HINDEN, R. 1998. Internet protocol, version 6 (IPv6) specification. http://www.ietf.org/rfc/rfc2460.txt.

DO, T., HUA, K., JIANG, N., AND LIU, F. 2009. PatchPeer: A scalable video-on-demand streaming system in hybrid wireless mobile peer-to-peer networks. *Peer-to-Peer Netw. Appl. 2,* 3, 182–201.

DO, T., HUA, K., AND TANTAOUI, M. 2008. Robust video-on-demand streaming in peer-to-peer environments. *Comput. Comm. 31,* 3, 506–519.

DOULKERIDIS, C., NORVAG, K., AND VAZIRGIANNIS, M. 2007a. DESENT: Decentralized and distributed semantic overlay generation in p2p networks. *IEEE J. Selected Areas Comm. 25,* 1, 25–34.

DOULKERIDIS, C., VLACHOU, A., KOTIDIS, Y., AND VAZIRGIANNIS, M. 2007b. Peer-to-peer similarity search in metric spaces. In *Proceedings of the 33rd International Conference on Very Large Databases (VLDB'07).* 986–997.

DUMINUCO, A. AND BIERSACK, E. 2008. Hierarchical codes: How to make erasure codes attractive for peer-to-peer storage systems. In *Proceedings of the 8th International Conference on Peer-to-Peer Computing*. IEEE Computer Society, Los Alamitos, CA, 89–98.

DUPUY, A., SCHWARTZ, J., YEMINI, Y., AND BACON, D. 1990. Nest: A network simulation and prototyping testbed. *Comm. ACM 33,* 10, 63–74.

EGER, K., HOSSFELD, T., BINZENHOFER, A., AND KUNZMANN, G. 2007. Efficient simulation of large-scale p2p networks: Packet-level vs. flow-level simulations. In *Proceedings of the 2nd Workshop on Use of P2P, GRID and Agents for the Development of Content Networks (UPGRADE'07).* ACM Press, New York, 9–16.

FENG, C. AND LI, B. 2008. On large-scale peer-to-peer streaming systems with network coding. In *Proceedings of the 16th ACM International Conference on Multimedia (MM'08).* ACM Press, New York, 269–278.

GARBACKI, P., EPEMA, D. H. J., AND VAN STEEN, M. 2007. An amortized tit-for-tat protocol for exchanging bandwidth instead of content in p2p networks. In *Proceedings of the 1st International Conference on Self-Adaptive and Self-Organizing Systems (SASO'07).* IEEE Computer Society, Washington, DC, 119–128.

Ghinita, G., Kalnis, P., and Skiadopoulos, S. 2007. MOBIHIDE: A mobile peer-to-peer system for anonymous location-based queries. In *Proceedings of the 10th International Conference on Advances in Spatial and Temporal Databases (SSTD'07).* Springer, 221–238.

Gross, C., Lehn, M., Stingl, D., Kovacevic, A., Buchmann, A., and Steinmetz, R. 2010. Towards a common interface for overlay network simulators. In *Proceedings of the 16th International Conference on Parallel and Distributed Systems (ICPADS'10)*. IEEE, 59–66.

Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., and Zhang, X. 2007. A performance study of bittorrentlike peer-to-peer systems. *IEEE J. Selected Areas Comm. 25,* 1, 155–169.

Gurbani, V. K., Basu, A., Schmidt, T., Fleming, S., Kolberg, M., and Waehlisch, M. 2011. Peer-to-peer simulation frameworks: A survey. IETF Internet-draft, work in progress, draft-gurbani-p2prgsimulation-survey-00

Haridasan, M. and van Renesse, R. 2008. Gossip-based distribution estimation in peer-to-peer networks. In *Proceedings of the 7th International Conference on Peer-to-Peer Systems (IPTPS'08).* USENIX Association, 13–19.

Hefeeda, M. and Saleh, O. 2008. Traffic modeling and proportional partial caching for peer-to-peer systems. *IEEE/ACM Trans. Netw. 16*, 1447–1460.

Henderson, T. 2008. The NS3 discrete event simulator. http://www.nsnam.org/docs/manual/html/events.html.

Hendler, J. and Golbeck, J. 2008. Metcalfe's law, web 2.0, and the semantic web. *Web Semantics Sci. Serv. Agents World Wide Web 6,* 1, 14–20.

Huang, C.-M., Hsu, T.-H., and Hsu, M.-F. 2007a. Network-aware p2p file sharing over the wireless mobile networks. *IEEE J. Selected Areas Comm. 25,* 1, 204–210.

Huang, Y., Farn Chen, Y., Jana, R., Jiang, H., Rabinovich, M., Reibman, A., Wei, B., and Xiao, Z. 2007b. Capacity analysis of mediagrid: A p2p itv platform for fiber to the node networks. *IEEE J. Selected Areas Comm. 25,* 1, 131–139.

Huebsch, R. J. 2008. PIER: Internet scale p2p query processing with distributed hash tables. Ph.D. thesis, EECS Department, University of California, Berkeley.

Iliofotou, M., Kim, H.-C., Faloutsos, M., Mitzenmacher, M., Pappu, P., and Varghese, G. 2009. Graph-based p2p traffic classification at the internet backbone. In *Proceedings of the 28th IEEE International Conference on Computer Communications Workshops (INFOCOM'09).* IEEE Press, 37–42.

Jelasity, M., Voulgaris, S., Guerraoui, R., Kermarrec, A.-M., and van Steen, M. 2007. Gossip based peer sampling. *ACM Trans. Comput. Syst. 25*, 8:1–8:36.

Jennings, C., Lowekamp, B., Rescorla, E., Baset, S. A., and Schulzrinne, H. 2012. REsource location and discovery (reload) base protocol. IETF internet-draft (work-in-progress), draft-ietf-p2psipbase-20.

Johansson, B., Rabi, M., and Johansson, M. 2007. A simple peer-to-peer algorithm for distributed optimization in sensor networks. In *Proceedings of the 46th IEEE Conference on Decision and Control*. IEEE, 4705–4710.

Kantere, V., Tsoumakos, D., Sellis, T., and Roussopoulos, N. 2009. GrouPeer: Dynamic clustering of p2p databases. *Inf. Syst. 34*, 62–86.

Karakaya, M., Korpeoglu, I., and Ulusoy, O. 2008. Counteracting free riding in peer-to-peer networks. *Comput. Netw. 52*, 675–694.

Karger, D., Lehman, E., Leighton, T., Panigrahy, R., Levine, M., and Lewin, D. 1997. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*. ACM Press, New York, 654–663.

Klingberg, T. and Manfredi, R. 2002. Gnutella 0.6. http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.

Kulbak, Y. and Bickson, D. 2005. The eMule protocol specification. http://www.cs.huji.ac.il/labs/danss/p2p/resources/emule.pdf.

Lakshman, A. and Malik, P. 2010. Cassandra: A decentralized structured storage system. *SIGOPS Oper. Syst. Rev. 44*, 35–40.

Laredo, J. L., Eiben, A. E., Steen, M., Castillo, P. A., Mora, A. M., and Merelo, J. J. 2008. P2P evolutionary algorithms: A suitable approach for tackling large instances in hard optimization problems. In *Proceedings of the 14th International Euro-Par Conference on Parallel Processing (Euro-Par'08).* Springer, 622–631.

Leonardi, E., Mellia, M., Horvath, A., Muscariello, L., Niccolini, S., and Rossi, D. 2008. Building a cooperative p2p-tv application over a wise network: The approach of the European FP-7 strep. NAPA-WINE. *IEEE Comm. Mag. 46,* 4, 20–22.

Leonini, L., Riviere, E., and Felber, P. 2009. Splay: Distributed systems evaluation made simple (or how to turn ideas into live systems in a breeze). In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*. USENIX Association, 185–198.

Likert, R. 1932. A technique for the measurement of attitudes. *Archives Psychol. 22,* 40, 1–55.

LIN, S., PAN, A., GUO, R., AND ZHANG, Z. 2005. Simulating large-scale p2p systems with the wids toolkit. In *Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*. IEEE, 415–424.

LIU, Y. 2007. On the minimum delay peer-to-peer video streaming: How realtime can it be? In *Proceedings of the 15th International Conference on Multimedia (MULTIMEDIA'07).* ACM Press, New York, 127–136.

LIU, Z., SHEN, Y., ROSS, K., PANWAR, S., AND WANG, Y. 2008. Substreamtrading: Towards an open p2p live streaming system. In *Proceedings of the IEEE International Conference on Network Protocols (INCP'08).* IEEE, 94–103.

MAGHAREI, N. AND REJAIE, R. 2009a. Overlay monitoring and repair in swarm-based peer-to-peer streaming. In *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'09).* ACM Press, New York, 25–30.

MAGHAREI, N. AND REJAIE, R. 2009b. PRIME: Peer-to-peer receiver-driven mesh-based streaming. *IEEE/ACM Trans. Netw. 17,* 4, 1052–1065.

MAVLANKAR, A., NOH, J., BACCICHET, P., AND GIROD, B. 2008. Peer-to-peer multicast live video streaming with interactive virtual pan/tilt/zoom functionality. In *Proceedings of the 15th IEEE International Conference on Image Processing (ICIP'08).* IEEE, 2296–2299.

MAYMOUNKOV, P. AND MAZIERES, D. 2002. Kademlia: A peer-to-peer information system based on the xor metric. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02).* 53–65.

MCCANNE, S. AND FLOYD, S. 1999. The network simulator - ns2. http://www.isi.edu/nsnam/ns/.

MITTAL, P. AND BORISOV, N. 2008. Information leaks in structured peer-to-peer anonymous communication systems. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08).* ACM Press, New York, 267–278.

MITTAL, P. AND BORISOV, N. 2009. ShadowWalker: Peer-to-peer anonymous communication using redundant structured topologies. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09).* ACM Press, New York, 161–172.

MOL, J.-D., EPEMA, D. H. J., AND SIPS, H. J. 2007. The orchard algorithm: Building multicast trees for p2p video multicasting without free-riding. *IEEE Trans. Multimedia 9*, 1593–1604.

MOL, J.-D., POUWELSE, J. A., EPEMA, D. H. J., AND SIPS, H. J. 2008. Free-riding, fairness, and firewalls in p2p file-sharing. In *Proceedings of the 8th International Conference on Peer-to-Peer Computing*. IEEE Computer Society, Washington, DC, 301–310.

NAICKEN, S., BASU, A., LIVINGSTON, B., AND RODHETBHAI, S. 2006a. A survey of peer-to-peer network simulators. In *Proceedings of the 7th Annual Postgraduate Symposium (PGNet'06).*

NAICKEN, S., BASU, A., LIVINGSTON, B., RODHETBHAI, S., AND WAKEMAN, I. 2006b. Towards yet another peer-to-peer simulator. In *Proceedings of the 4th International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs'06).*

NAICKEN, S., LIVINGSTON, B., BASU, A., RODHETBHAI, S., WAKEMAN, I., AND CHALMERS, D. 2007. The state of peer-to-peer simulators and simulations. *ACM Comput. Comm. Rev. 37,* 2, 95–98.

PIANESE, F., PERINO, D., KELLER, J., AND BIERSACK, E. 2007. PULSE: An adaptive, incentive-based, unstructured p2p live streaming system. *IEEE Trans. Multimedia 9,* 8, 1645–1660.

RATNASAMY, S., FRANCIS, P., HANDLEY, M., KARP, R., AND SHENKER, S. 2001. A scalable content addressable network. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'01).* ACM Press, New York, 161–172.

REN, D., LI, Y.-T., AND CHAN, S.-H. 2008. On reducing mesh delay for peer-to-peer live streaming. In *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM'08).* IEEE, 1058–1066.

RISSON, J. AND MOORS, T. 2006. Survey of research towards robust peer-to-peer networks: Search methods. *Comput. Netw. 50*, 3485–3521.

ROSENBERG, J., SCHULZRINNE, H., CAMARILLO, G., JOHNSTON, A., PETERSON, J., SPARKS, R., HANDLEY, M., AND SCHOOLER, E. 2002. SIP: Session Initiation Protocol. RFC 3261 (proposed standard).

ROWAIHY, H., ENCK, W., MCDANIEL, P., AND LA PORTA, T. 2007. Limiting sybil attacks in structured p2p networks. In *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM'07).* IEEE, 2596–2600.

ROWSTRON, A. I. T. AND DRUSCHEL, P. 2001. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg (Middleware'01).* Springer, 329–350.

SETTON, E., BACCICHET, P., AND GIROD, B. 2008. Peer-to-peer live multicast: A video perspective. *Proc. IEEE 96,* 1, 25–38.

Silva, A. P. C. D., Leonardi, E., Mellia, M., and Meo, M. 2008. A bandwidth-aware scheduling strategy for p2p-tv systems. In *Proceedings of the 8th International Conference on Peer-to-Peer Computing*. IEEE Computer Society, Los Alamitos, CA, 279–288.

Silverston, T., Fourmaux, O., and Crowcroft, J. 2008. Towards an incentive mechanism for peer-to-peer multimedia live streaming systems. In *Proceedings of the 8th International Conference on Peer-to-Peer Computing*. IEEE Computer Society, Los Alamitos, CA, 125–128.

Skobeltsyn, G., Luu, T., Podnar-Arko, I., Rajman, M., and Aberer, K. 2009. Query-driven indexing for scalable peer-to-peer text retrieval. *Future Gener. Comput. Syst. 25*, 89–99.

Stingl, D., Gross, C., Ruckert, J., Nobach, L., Kovacevic, A., and Steinmetz, R. 2011. Peerfactsim. kom: A simulation framework for peer-to-peer systems. In *Proceedings of the International Conference on High Performance Computing and Simulation (HPCS'11)*. IEEE, 577–584.

Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., and Balakrishnan, H. 2001. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM Comput. Comm. Rev. 31*, 149–160.

Tanin, E., Harwood, A., and Samet, H. 2007. Using a distributed quadtree index in peer-to-peer networks. *VLDB J. 16*, 165–178.

Terpstra, W. W., Kangasharju, J., Leng, C., and Buchmann, A. P. 2007. Bubblestorm: Resilient, probabilistic, and exhaustive peer-to-peer search. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'07)*. ACM Press, New York, 49–60.

Urban, P., de Fago, X., and Schiper, A. 2001. Neko: A single environment to simulate and prototype distributed algorithms. In *Proceedings of the 15th International Conference on Information Networking*. IEEE, 503–511.

van Renesse, R., Birman, K. P., and Vogels, W. 2003. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Trans. Comput. Syst. 21*, 164–206.

Wang, S., Ooi, B. C., Tung, A., and Xu, L. 2007. Efficient skyline query processing on peer-to-peer networks. In *Proceedings of the 23rd IEEE International Conference on Data Engineering (ICDE'07)*. IEEE, 1126–1135.

Wang, Y., Carzaniga, A., and Wolf, A. 2008. Four enhancements to automated distributed system experimentation methods. In *Proceedings of the 30th International Conference on Software Engineering*. ACM Press, New York, 491–500.

Webb, S. and Soh, S. 2007. Round length optimisation for p2p network gaming. In *Proceedings of the 8th Postgraduate Electrical Engineering and Computing Symposium*. 23–28.

Wilcox-O'Hearn, Z. and Warner, B. 2008. Tahoe: The least-authority filesystem. In *Proceedings of the 4th ACM International Workshop on Storage Security and Survivability (StorageSS'08)*. ACM Press, New York, 21–26.

Xie, H., Yang, Y. R., Krishnamurthy, A., Liu, Y. G., and Silberschatz, A. 2008. P4P: provider portal for applications. In *Proceedings of the ACM SIGCOMM Conference on Data Communication (SIGCOMM'08)*. ACM Press, New York, 351–362.

Yang, S. J. and Chen, I. Y. 2008. A social network-based system for supporting interactive collaboration in knowledge sharing over peer-to-peer network. *Int. J. Hum.-Comput. Stud. 66,* 1, 36–50.

Yiu, W.-P., Jin, X., and Chan, S.-H. 2007. VMesh: Distributed segment storage for peer-to-peer interactive video streaming. *IEEE J. Selected Areas Comm. 25,* 9, 1717–1731.

Yunhao, L., Xiao, L., and Ni, L. 2007. Building a scalable bipartite p2p overlay network. *IEEE Trans. Parallel Distrib. Syst. 18,* 9, 1296–1306.

Zaharia, M. and Keshav, S. 2008. Gossip-based search selection in hybrid peer-to-peer networks. *Concurr. Comput. Pract. Exper. 20*, 139–153.

Zantout, B. and Haraty, R. 2011. I2p data communication system. In *Proceedings of the 10th International Conference on Networks.* 401–409.

Zhang, D., Chen, X., and Yang, H. 2009. State of the art and challenges on peer-to-peer simulators. In *Proceedings of the 5th International Conference on Wireless Communications, Networking and Mobile Computing*. IEEE, 1–4.

Zhao, B., Kubiatowicz, J., and Joseph, A. 2001. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. *Comput. 74*, 11–20.

Zhou, R. and Hwang, K. 2007. Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Trans. Parallel Distrib. Syst. 18*, 460–473.

Zhou, R., Hwang, K., and Cai, M. 2008. GossipTrust for fast reputation aggregation in peer-to-peer networks. *IEEE Trans. Knowl. Data Engin. 20*, 1282–1295.

Zhou, Y., Chiu, D. M., and Lui, J. 2007. A simple model for analyzing p2p streaming protocols. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP'07)*. IEEE, 226–235.

Zhu, W., Wang, C.-L., and Lau, F. 2002. JESSICA2: A distributed java virtual machine with transparent thread migration support. In *Proceedings of the IEEE International Conference on Cluster Computing*. IEEE, 381–388.

Zhuge, H. and Li, X. 2007. Peer-to-peer in metric space and semantic space. *IEEE Trans. Knowl. Data Engin. 19,* 6, 759–771.

Received August 2011; revised July 2012; accepted July 2012