This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE P2P 2011 proceedings.

Towards a Comparative Performance Evaluation of Overlays for Networked Virtual Environments

Christian Gross[1], Max Lehn[2], Christoph Münker[1],
Alejandro Buchmann[2], Ralf Steinmetz[1]
Multimedia Communications Lab[1], Databases and Distributed Systems[2]
Technische Universität Darmstadt
Email: {chrgross, cmuenker, steinmetz}@kom.tu-darmstadt.de
{mlehn, buchmann}@dvs.tu-darmstadt.de

***Abstract*—Peer-to-peer overlays for Networked Virtual Environments have recently gained much research interest, resulting in a variety of different approaches for spatial information dissemination. Although designed for the same purpose, the evaluation methodologies used by particular authors differ widely. This makes any comparison of existing systems difficult, if not impossible. To overcome this problem we present a benchmarking methodology which allows for a fair comparison of those systems. We, therefore, define a common set of workloads and metrics. We demonstrate the feasibility of our approach by testing four typical systems for spatial information dissemination and discovering their specific performance profiles.**

***Index Terms*—Peer-to-Peer, Networked Virtual Environments, NVE, Benchmarking, Performance Evaluation, Workload, Metrics, Comparison, Comparative Evaluation**

I. Introduction

With the increasing research interest in Networked Virtual Environments (NVE), including Massively Multiplayer Online Games (MMOG), there have been several proposals to peer-to-peer (P2P) systems designed specifically for those applications. A basic requisite of data management for NVEs is the alignment of objects and participants (more precisely: their avatars) in the virtual space. The awareness and interaction of participants are in most cases related to a spatial context. To this end, each participant has a limited vision and interaction range. The network systems for NVEs thus need to be efficient in querying for objects and disseminating data in a specific region of the virtual world. A second important aspect of NVEs is their real-time demand, requiring a high responsiveness and low update delays.

In this paper we focus on systems for NVEs which disseminate spatial information among participants moving in a continuous virtual (game) world. Each participant is interested in events occurring in a particular part of the world, called the *area of interest* (AOI). The AOI, which we assume to have a circular shape, is determined by the vision range of the participant. Position updates are disseminated to neighboring participants that reside in one's AOI.

The design of current solutions ranges from traditional client-server systems over Distributed-Hashtable-based systems [1], [9], [20] to unstructured overlays [6], [18]. All of those systems have highly different performance characteristics and their respective publications vary in the evaluation scenarios, metrics, and assumptions being used. Hence, it is infeasible to draw conclusions about the relative performance of only two systems from their available evaluation results. Often, the authors focus on micro metrics for evaluating their respective approaches, e.g., metrics specific to Distributed Hashtables (DHTs) such as the routing table size or hop count. This is of course justifiable for investigating detailed aspects of a specific system, but systems not using a DHT can only be compared using macro metrics defined on the functional interface.

To overcome this problem, the contributions of this paper are twofold: (i) in order to determine which system performs best under certain conditions, we propose a benchmarking methodology, comprising a set of common tests, metrics, and workloads, such that the characteristics of systems can be compared in a fair and unbiased way. This allows for the identification of advantages, drawbacks, and performance bottlenecks in the design of existing systems. Furthermore, it can be used to tune system parameters as well as for the performance evaluation during the development of new systems. (ii) We show the feasibility of our approach by benchmarking four typical systems (VON [6], pSense [18], Mercury [1], and a client-server implementation) using our benchmarking methodology.

The rest of the paper is structured as follows: Section II summarizes the related work in the area of P2P benchmarks and approaches for interest management in NVEs. Section III describes our proposed benchmarking methodology followed by the benchmarking results presented in Section IV. Finally, Section V concludes the paper and gives an outlook on the future work.

II. Background and Related Work

The related work section consists of two parts. First, we discuss existing approaches in the area of performance evaluation and benchmarking for P2P systems. Afterwards, we present the most prominent P2P spatial information dissemination approaches.

[1,2]Authors supported by the German Research Foundation, Research Group 733, "QuaP2P: Quality Improvement of Peer-to-Peer Systems".

A. P2P Benchmarks

In the area of P2P there are still only a few benchmarking approaches. General ideas for a P2P benchmarking platform and an analysis of existing tools for P2P network simulation have been presented by Kovacevic et al. [11]. A concrete benchmarking scenario for structured overlays in the context of NVEs focuses on lookup and routing latencies as well as the overlay message distribution [10]. Unlike our approach, the authors use metrics specific to search overlays to quantify quality aspects, e.g., the hop count. Those metrics are not applicable to the overlays investigated in this paper.

A comprehensive benchmark for P2P web search engines was proposed by Neumann et al. [14]. The benchmark suggests the freely available Wikipedia content as the document corpus and queries taken from Google's Zeitgeist archive. Nocentini et al. present a performance evaluation of implementations of the JXTA rendezvous protocol [16]. The paper specifies the metrics lookup time, memory load, CPU load, and dropped query percentage, as well as the parameters query rate, presence of negative queries, and type of peer disconnections. Furthermore, a performance vs. cost framework for DHTs has been proposed by Li et al. [13]. Quétier et al. presented a testbed which focuses on the performance evaluation of large-scale distributed systems [17]. P2PTester [3] is a project aiming to provide a tool for measuring large-scale P2P data management systems. The project focus is on the applicability to various kinds of systems using a modular measurement architecture. Blanas and Samoladas [2] propose performance metrics for responsiveness and throughput using a rather abstract model, focusing on contention in P2P systems.

B. P2P Spatial Information Dissemination Systems

Spatial information dissemination systems are primarily applied in NVEs for low-latency avatar position updates. According to the classification of design issues as described by Fan et al. [5], they incorporate the two issues interest management and game event dissemination. Further issues, such as object management, are considered orthogonal and will be addressed separately as future work.

Current P2P-based systems for spatial information dissemination can be classified into Distributed-Hashtable-Table-based (DHT) and unstructured approaches. SimMud [9], Mercury [1], and Mopar [20] belong to the first category. They disseminate position updates among nodes using a Publish/ Subscribe approach on top of a DHT. The virtual world is split into regions. Each region is assigned to a peer, which is responsible for managing the subscriptions. Nodes subscribe to a set of fixed regions or an arbitrary part of the virtual world. In unstructured systems, such as pSense [18] or VON [6], peers directly exchange information about surrounding peers and their current position. Therefore, each peer maintains an individual set of neighbors in its AOI.

pSense maintains a *near node list* of AOI neighbors and a *sensor node list* of up to eight peers that are outside the AOI. The purpose of the sensor nodes is to maintain the connectivity of the network by introducing approaching neighbors.

In contrast to that, each VON peer maintains a Voronoi diagram based on its neighbors' positions. According to the AOI, neighbors are divided into boundary- and enclosing neighbors. Boundary neighbors inform a local node about new potential neighbors.

III. Methodology

We define our benchmark according to our common methodology [12], which comprises the following components: (i) the system specification and requirements the system under test (SUT) has to fulfill, (ii) the workload schemes being applied on the SUT, and (iii) the metrics being measured during the benchmark. The SUT is in our case the spatial information dissemination system.

A. System Specification and Requirements

A benchmark for a specific class of systems must consider the functional and non-functional requirements of a system.

1) Functional Requirements: Based on the functional requirements, an interface can be derived and used later for applying the load on the system. Listing 1 shows the abbreviated functional interface, which we use for our benchmark. The interface provides means for joining and leaving the overlay network. When joining the network the node has to specify the position within the virtual world where the node should be inserted. The method `leave()` initiates a graceful leave, if supported by the system. Using the `disseminatePosition()` method a node specifies the position information to be disseminated to its surrounding neighbors. `setAOISize()` specifies the size of the AOI. We assume a circular AOI, as this is the most common and simplest shape. (But other shapes can be considered as well.) The `getNeighbors()` method returns a list of all known AOI neighbors including their positions.

```
– void join(Point p);
– void leave();
– void disseminatePosition(Point p);
– void setAOISize(int radius);
– List<NodeInfo> getNeighbors();
```

Listing 1. Interface of the System under Test (SUT)

2) Non-Functional Requirements: We identified the following non-functional requirements, which we call *Quality Aspects*: (i) The system should be *scalable* with respect to the number of nodes, (ii) it should be *robust* against massive joining or leaving of participants, (iii) be *stable* in terms of high churn rates, packet loss, and high mobility rates, (iv) always deliver *valid* (*correct*, *complete*, and *consistent*) results, (v) the system should always be *responsive*, as a result of good *performance*, (vi) while consuming a reasonable amount of resources (low *costs*). (vii) Finally, the system should distribute the operational costs equally among the participants (or proportional to their capabilities) as well as provide the same service quality to all participants (*fairness*).

An analysis of the above non-functional requirements shows that there are requirements related to workload variation schemes such as scalability, stability, and robustness. In contrast to that there are requirements such as validity, performance, costs, and fairness, which inherently define metrics.

B. Workload

The workload should reflect the most critical situations in the lifetime of the SUT and should drive it to its performance limits in order to gain insights on performance bottlenecks. We distinguish three main categories of workload factors: peer availability, player behavior, and underlay network properties. The peer availability comprises generic aspects of user behavior in a peer-to-peer system where participants are continuously joining and leaving. The player behavior describes the system-specific user behavior, in our case the activities of players of an MMOG, or more generally participants of an NVE. Finally, the underlay network properties describe the underlaying IP network (Internet).

1) Peer Availability:

- **Number of peers (player density):** In P2P systems, the scalability with respect to the number of participants (i.e., peers) is usually a very important feature. Assuming a fixed world size, the average player density of a virtual world is proportional to the total number of players. A higher density causes a higher average number of AOI neighbors, which in turn causes a potentially higher bandwidth utilization.
- **Churn:** Churn is the effect of peers continuously joining and leaving the system. We assume an exponential node lifetime, which is quantified by the mean session length. The churn rate is defined inversely proportional to the mean session length.
- **Massive join:** A special case of churn is the massive join scenario where a certain percentage of new participants joins the network within a short amount of time. In real life, this could be caused by human flash-crowd behavior or by technical reasons like a network blackout.
- **Massive leave/crash:** Corresponding to massive join, there is a massive leave/crash scenario where a certain percentage of participants leaves the system or crashes. This stresses the overlay as it has to recover many broken neighborhood connections.

2) Player Behavior:

- **Mobility Pattern:** The player mobility model determines the movement of the individual players in the virtual space. As a default, we use a random waypoint model, which is also employed in many of the original evaluations. Each player repeatedly selects a random point in a given coordinate range and navigates to the point with a constant velocity.
  As a special case, we vary the movement model in such a way that all players in the virtual game world meet at a single point at the same time as shown in Figure 1. This particularly stresses the system as every player has to maintain a large number of players in its vision range.
- **Velocity:** Another important workload factor is the velocity of the nodes. The higher the velocity of the participants, the more rapidly change the neighborhood relations and the system has to dedicate more effort to keep the relations up to date.
- **Portals:** In some games, portals enable players to ‘teleport’, i.e., to suddenly change their position in the game world. This effect cause stress on the system as neighborhood relations change unpredictably and need to be updated in order to assure a smooth playing experience. When the portal effect is enabled, players jump to a random point in the game world as shown in Figure 2. We parameterize the workload such that each node in the simulation uses a portal at least once during the simulation.

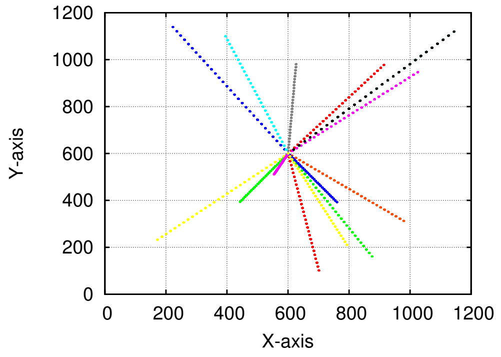
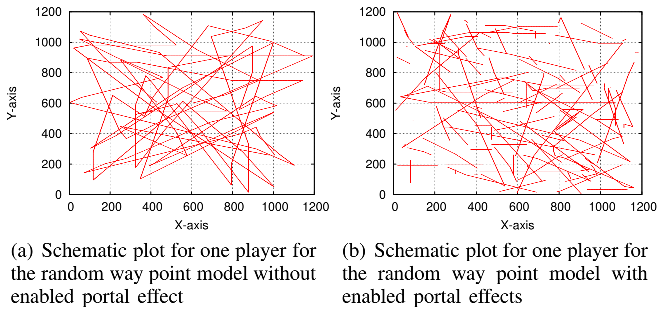


Figure 1. Schematic plot of the central point movement model with 15 player



(a) Schematic plot for one player for the random way point model without enabled portal effect

(b) Schematic plot for one player for the random way point model with enabled portal effects

Figure 2. Random way point model with and without portal effects enabled.

3) Underlay Network Properties:

- **Bandwidth:** The available bandwidth per node is a crucial factor for the performance of a peer-to-peer system. Exceeding the available bandwidth results in packets being dropped, which e.g., causes position updates not being disseminated. For the sake of simplicity we assume asymmetric and homogeneous node bandwidths, except for the server in the client/server scenario, whose bandwidth is assumed to be unlimited.
- **Delay:** The second important underlay factor is the end-to-end message delay. For varying the delay, we applied a simple delay model, which generates normal distributed delays with a mean delay of 50, 100, and 200ms.
- **Message loss:** Finally, the underlay network reliability is represented by the message loss. Message loss is modeled by independently dropping a certain percentage of the

messages. The loss of position updates might cause the view of different participants to become inconsistent or even the overlay topology to collapse.

C. Metrics

This section describes the metrics used to capture the SUT characteristics. To be independent from a concrete SUT implementation, we focus on macro metrics, i.e., metrics that can be measured on top of the SUT interface (see Section III-A). In contrast to micro metrics, which refer to internals of specific systems, only macro metrics enable a comparison of all systems implementing the given interface.

1) Definitions: We use the following symbols for specifying the metrics:

- $P(t)$ as the set of all participants participating in the overlay at time t.
- T as the global set of sampling timestamps.
- $vis(i)$ as the vision radius of participants $i \in P(t)$, which is considered to be constant over time.
- $pos(i,j,t)$ as the position of participant i perceived by participant j at time t. We assume that $pos(i,i,t)$ is the 'true' position of participant i.
- $N(i,t) = \{j \in P(t) \mid \Delta pos(i,j,t) \le vis(i)\}$ with $\Delta pos(i,j,t) = \|pos(i,i,t) - pos(j,j,t)\|_2$ as the *ideal* set of neighboring participants for a given participant $i \in P(t)$ at time t.
- $M(i,t)$ as the set of neighboring participants *known* by a given participant $i \in P(t)$ at time t.
- $cen(i,t)$ as the centroid of the perceived positions of participant i by the neighbors $j \in M(i,t)$ at time t. The centroid is calculated as
$$cen(i,t) = \frac{\sum_{j \in M(i,t)} pos(i,j,t)}{|M(i,t)|}.$$
- $wcc(t)$ as the weakly connected component at time t. It describes the maximum subset of nodes in the connection graph such that two arbitrarily chosen nodes from the subset are connected via an undirected path.

2) Performance Metrics: The following metrics are measured per participant in constant time intervals resulting in a set of samples.

- Using confusion matrices, one can divide the neighborhood set of a participant i into the four categories as shown in Table I. Based on the confusion matrix the metrics recall and precision are defined as:
$$recall(i,t) = \frac{tp(i,t)}{tp(i,t) + fn(i,t)}$$
$$precision(i,t) = \frac{tp(i,t)}{tp(i,t) + fp(i,t)}.$$
Recall, thus, describes the ratio between the known relevant neighbors ($M \cap N$) and all relevant neighbors (N) in a node's neighbor set. Precision defines the fraction of relevant participants (($M \cap N$) / M). In case of an ideal system both recall and precision are equal to 1.

	$j \in M(i,t)$	$j \notin M(i,t)$
$j \in N(i,t)$	true positive, $tp(i,t)$	false negative, $fn(i,t)$
$j \notin N(i,t)$	false positive, $fp(i,t)$	true negative, $tn(i,t)$

Table I
CONFUSION MATRIX FOR PARTICIPANT i

We define the the following metrics based on the neighborhood relation of the participants:

- The mean position error at participant i as:
$$err(i,t) = \frac{\sum_{j \in M(i,t)} \|pos(i,j,t) - pos(i,i,t)\|_2}{|M(i,t)|}.$$
Note that this metric only includes the neighbors actually *known* by participant i.
- The dispersion of the perceived position of participant i relative to the centroid $cen(i,t)$:
$$s_{pos}(i,t) = \sqrt{\frac{\sum_{j \in M(i,t)} (\|pos(i,j,t) - cen(i,t)\|_2)^2}{|M(i,t)| - 1}}.$$
In an ideal system the standard deviation is zero as the actual position of the participant $pos(i,i,t)$ and the centroid of the perceived positions of $cen(i,t)$ are identical.

3) Cost Metrics:

- $traf(i,t)$ as the traffic in bytes per second on participant i at time t. The traffic is measured in order to quantify the *costs* of a system. We assume other costs such as CPU, memory, storage, and energy to be uncritical as most of the modern gaming PCs provide sufficient ressources.

4) Global Metrics: The following metric is calculated on the neighborhood relations among the peers:

- The Global Connected Component Factor (gccf), which is defined as follows:
$$gccf(t) = \frac{|wcc(t)|}{|P(t)|}.$$
It is 1.0 in case that there exist a path between any two arbitrarily chosen nodes in the network. In other words, the overlay is not split into several partitions. In case of a partition the gccf describes the share of the biggest partition of the total network.

5) Aggregation of Metrics: Based on the per-participant samples specified above, the following aggregation methods can be applied:

- The average of a metric x over the set of participants at time $t \in T$:
$$\overline{x}(t) = \frac{1}{|P|} \sum_{i \in P} x(i,t).$$
- The average of a metric x over the set of samples per participant $i \in P$:
$$\widetilde{x}(i) = \frac{1}{|T|} \sum_{t \in T} x(i,t).$$

- The total average of a metric x:
$$\widehat{x} = \frac{1}{|T||P|} \sum_{t \in T} \sum_{i \in P} x(i,t).$$
- Jain's fairness index $F(x)$ for a given metric x with
$$F(x) = \frac{(\sum_{i \in P} (\widetilde{x}(i))^2}{(|P| \sum_{i \in P} \widetilde{x}(i)^2)},$$
which is 1 in the best case and converges to $\frac{1}{|P|}$ in the worst case [8]. The best case occurs when the same amount of resources is allocated to every user in the system. In contrast, the worst case describes the situation when all resources are allocated to a single user and the remaining users starve. In case that the performance and costs should be distributed according to the peer capacities, the values $\widetilde{x}(i)$ can be normalized according to the peer capabilities before calculating Jain's fairness index.

D. Benchmark Implementation

Having introduced the requirements, metrics, and workload schemes, we now combine those, forming the concrete benchmarking scenarios. We vary the workload in four scenarios (baseline, scalability, stability, robustness). Thereby, we investigate the characteristics of the SUT with respect to performance, validity, costs, and fairness.

Baseline. Initially, we define a constant baseline workload setup, representing an environment with idealized conditions, which should give an impression on the system behavior under such conditions. For the benchmark we used the baseline parameters as shown in Table II. The parameter values haven been chosen based on the work by Schmieg et al. [18]. Furthermore, to enable best possible comparability to the related work, we selected the random waypoint model for the baseline workload. The update frequency of 5 units per second was chosen based on an analysis of MMOGs conducted by Chen et al. [4]. The simulation setup is as follows: Initially, 250 nodes join the system within 8 min of simulated time. Afterwards, a stabilization phase of 2 min takes place. The measurement interval starts at minute 10 and lasts until the end of the simuation after 30 min of simulated time.

To investigate the *performance* and *validity*, we measure (i) the neighbor set recall ($recall(i,t)$) and precision ($precision(i,t)$) as well as (ii) the position error ($err(i,t)$) and the perceived position dispersion ($s_{pos}(i,t)$). Precision is less relevant than recall, since—besides a potential traffic penalty—false positives in the neighbor set usually do not hurt the application. The traffic $traf(i,t)$ quantifies the operational *costs*. *Fairness* according to service quality is measured by $F(recall)$, $F(precision)$, $F(err)$, and $F(s_{pos})$. Cost fairness is measured by $F(traf)$.

Scalability. For investigating the scalability of the system, we vary the workload in a horizontal and vertical manner. The former scales the total number of nodes while preserving the virtual world size. Horizontal scaling, thus, affects the avatar density. Vertical scaling is realized by increasing the velocities. We define a system to be scalable up to a certain workload level if the performance, validity, and costs do not exceed or drop below predefined thresholds. These thresholds, however, are application-specific.

Parameter	**Value**
Game World Size	1200 x 1200 units
Area of Interest radius	200 units
Number of nodes	250
Simulation duration	30 min
Movement model	Random way point model
Movement speed	20 units/s
Update frequency	5/s
Underlay model	GNP Latency Model [15]
Upload capacity	16 kB/s
Download capacity	128 kB/s

Table II
BASELINE PARAMETER SETUP

Stability. The stability of the SUT is investigated in a heavy churn scenario with an exponential session length with an average of 50, 25, and 10 min. The relevant metrics for stability are those for validity and fairness. While it is expected that the system responses are valid at all times, a degradation of fairness is an indicator for upcoming stability problems. A low cost fairness reflects potentially overloaded nodes, whereas a low service quality fairness indicates possibly starving nodes.

Robustness. In order to test the robustness capabilities of the system, we apply a massive join, a massive crash, as well as a packet loss scenario. For the first scenario we assume 50% and 100% new participants respectively joining the system at the same time. In the second scenario the percentage of failing participants is set to 25% and 50%. For the packet loss scenario we define loss rates of 5%, 10%, and 25%. We consider the system to be stabilized after a massive join or leave of nodes, if the moving average of a given metric stabilizes at a certain level. A similar method has been suggested by Jain [7] in order to detect a steady state within a set of measurements. In the message loss scenario, the relevant metrics are those for validity and performance.

IV. BENCHMARKING RESULTS

The goal of this evaluation section is to demonstrate the feasibility of our benchmarking methodology and give a first impression on the characteristics of the tested systems. We implemented and tested four different systems for spatial information dissemination, satisfying the functional interface as described in Section III-A. The systems are: the unstructured approaches (i) VON and (ii) pSense, (iii) the content-based Publish/Subscribe approach Mercury, which is built on top of the Chord DHT, and (iv) a simple client-server-based approach (C/S), which serves as a performance reference. The client-server approach works as follows: All nodes participating in the NVE are connected to a central server and send their position updates every 200ms. From these position updates the server computes an updated neighbor set per participant and disseminates this information to all participants in the network every 200ms.

All SUTs have been implemented in the discrete event-based overlay simulator PeerfactSim.KOM [19]. In the following, we present the most important results for each of the benchmark phases as specified in Section III.

A. Baseline

The results for the baseline benchmark are shown in Table III. In the benchmark we assume idealized conditions with no peer churn or message loss taking place. When looking at the average recall ($\widehat{recall}$) of all four systems, we notice that VON provides the lowest neighbor detection rate whereas the remaining three systems show a recall of above 0.9. With respect to the position error ($\widehat{err}$), however, VON shows the best performance with a position error of arround 2 units, which is about half of the position error of the C/S-based approach. The reason for the good performance of VON is that updates are disseminated directly over one hop, whereas the C/S approach needs always two hops (Player A to Server and Server to Player B). Mercury shows the worst performance with respect to the position error as update are disseminated over multiple hops in the DHT.

System	Unit	C/S	pSense	VON	Mercury
$\widehat{recall}$	1	0.9791	0.9304	0.8304	0.9162
$F(recall)$	1	1.0000	1.0000	0.9586	0.9989
$\widehat{err}$	units	5.2043	9.8512	2.0186	13.5779
$F(err)$	1	0.9558	0.9865	0.6921	0.8146
$\widehat{traf_{up}}$	kB/s	0.2181	15.5601	3.5732	5.5020
$F(traf_{up})$	1	1.0000	1.0000	0.8469	0.8435
$\widehat{traf_{down}}$	kB/s	2.3199	15.5601	3.5733	5.4955
$F(traf_{down})$	1	0.9965	0.9995	0.8471	0.8554

Table III
RESULTS FOR THE BASELINE BENCHMARK AVERAGED OVER TIME AND PEERS

In order to evaluate the fairness of the system with respect to the position error, we compute Jain's fairness index ($F(err)$). VON shows a fairness index of 0.69, which indicates that the position error that peers in the VON overlay perceive is unequally distributed. In contrast, the C/S-based approach as well as pSense show a fairness value above 0.95, thus, providing a good service fairness. From the upload traffic point of view pSense utilizes the full upload bandwidth of 16 kB/s as position updates include the whole receiver list with all neighboring peers in the vision range. As expected, the C/S approach consumes the least bandwidth as a node in the C/S approach sends its position update only to the server instead of multiple neighboring peers. When investigating the cost fairness of all systems, we notice that VON and Mercury distribute the load less equal than the C/S approach and pSense.

B. Scalability

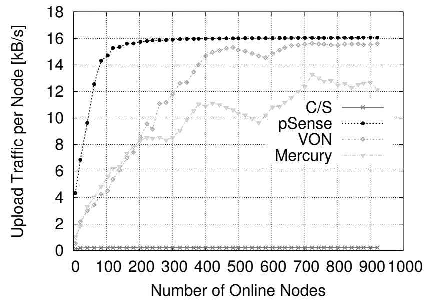
Figure 3 shows the results for the scalability benchmark using horizontal scaling (number of nodes) followed by Figure 4 showing the results using vertical scaling (node speed). When scaling horizontally, the upload bandwidth consumption of pSense is growing rapidly with the increasing number of participants and reaches the predefined maximum of 16kB/s. With more than 400 nodes, VON's upload consumption also reaches the 16kB/s maximum and VON is no longer capable of keeping the neighbor sets up to date. This results in a rapidly dropping recall and at the same time in a rapidly increasing position error. As VON remains in this state of saturation, the overlay structure collapses, resulting in a recall of almost zero. Mercury also shows a recall dropping below the clearly unacceptable value of 50% with 500 nodes. The position error of pSense and Mercury increases linearly with an increasing number of participants. Under the assumption that the server has sufficient bandwidth, the C/S results remain constant.

Figure 4(a) shows the relation between the node speed (vertical scaling) and the global connected component factor. As one can observe from the plot, the four systems remain fully connected up to a node speed of 250 units per second. Above this threshold the connectivity of VON starts to deteriorate, whereas the remaining three systems remain fully connected. The reason for this deterioration is that a VON node i does not maintain connections to other nodes beyond its vision range $vis(i)$. In case that the node speed exceeds the vision radius per second (200 units/s), VON is unable to detect new bypassing nodes because the relative speed of two nodes passing each other is then twice the average node speed. With respect to recall, all four systems show a similar behavior as their recall drops with an increasing node speed, as shown in Figure 4(b). All four systems show a linearly increasing position error up to a speed of 200 units/s (Figure 4(c)). Above this value, all peer-to-peer-based systems experience higher fluctuations in the position error. VON's position error is not further increasing as its overlay structure has completely collapsed.

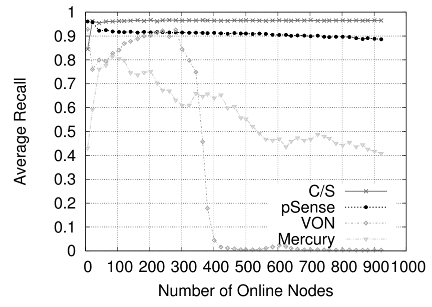
C. Stability

The results of the stability benchmark for different levels of node churn are shown in Table IV. With respect to recall, C/S, Mercury, and pSense remain stable with an increasing churn rate, whereas the recall of VON is dropping to 0.864 at 10 minutes mean session length. Another interesting observation is related to the position error. The C/S approach achieves a position error of around 5.4 units, whereas pSense and Mercury have position errors of about 9.2 and 11.5 units respectively. VON shows the best performance with a position error of about 3 units. The reason for this is again the dissemination of position updates in one hop (VON) and two hops (C/S) as already described in Section IV-A. pSense shows the highest position error because position updates are sent redundantly, saturating the upload bandwidth, which in turn causes the loss of update messages. Similarly to the recall results, Mercury's multi-hop dissemination leads to higher position errors.

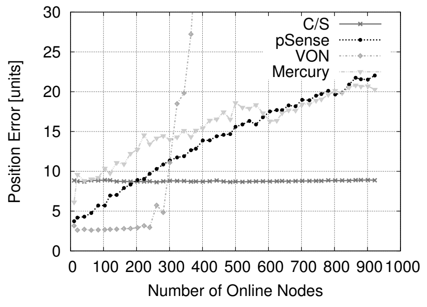
Figure 5 shows the results for the stability benchmark with portal effects enabled. As shown in Figure 5(a), VON's



(a) Average upload bandwidth consumption depending on the number of online nodes

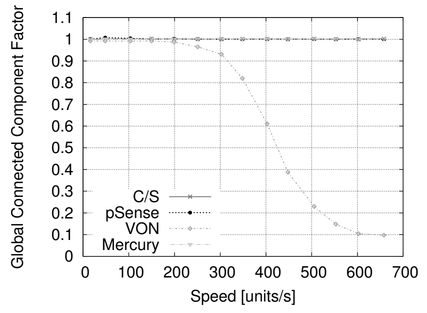


(b) Average recall of the neighbor sets depending on the number of online nodes

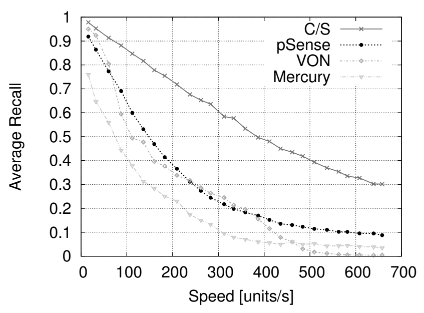


(c) Average position error depending on the number of online nodes

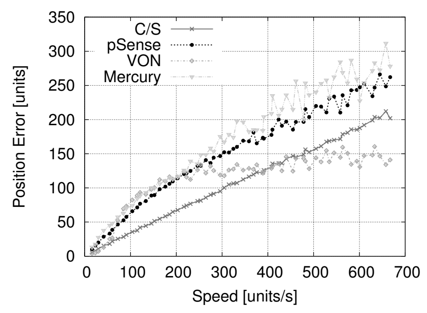
Figure 3. Results of the scalability benchmark using horizontal scaling by increasing the number of nodes



(a) Global connected component factor in relation to the average speed



(b) Average recall of the neighbor set as a function of the speed



(c) Average position error of the neighbor set as a function of the speed

Figure 4. Results of the scalability benchmark using vertical scaling by increasing the node speed

System	Unit	C/S			pSense			VON			Mercury		
Mean Session Time	**min**	**50**	**25**	**10**	**50**	**25**	**10**	**50**	**25**	**10**	**50**	**25**	**10**
$\widehat{recall}$	1	0,977	0,976	0,973	0,929	0,928	0,925	0,974	0,957	0,864	0,939	0,939	0,940
$F(recall)$	1	1,000	1,000	1,000	1,000	1,000	1,000	1,000	0,992	0,955	0,970	0,942	0,880
$\widehat{err}$	units	5,412	5,416	5,403	9,705	9,270	8,955	3,311	2,918	2,652	11,670	11,519	11,922
$F(err)$	1	0,959	0,959	0,958	0,987	0,985	0,984	0,893	0,877	0,845	0,958	0,933	0,873
$\widehat{traf_{up}}$	kB/s	0,219	0,219	0,219	15,631	15,607	15,583	9,578	9,031	7,036	7,560	6,908	5,882
$F(traf_{up})$	1	1,000	1,000	1,000	1,000	1,000	1,000	0,993	0,981	0,931	0,766	0,773	0,734
$\widehat{traf_{down}}$	kB/s	2,107	2,056	1,936	15,211	15,172	15,094	9,337	8,801	6,853	7,206	6,417	5,160
$F(traf_{down})$	1	0,997	0,996	0,995	1,000	0,999	0,999	0,993	0,981	0,931	0,941	0,922	0,886

Table IV
SELECTED RESULTS OF THE STABILITY BENCHMARK AVERAGED OVER TIME AND PARTICIPANTS FOR DIFFERENT LEVELS OF CHURN

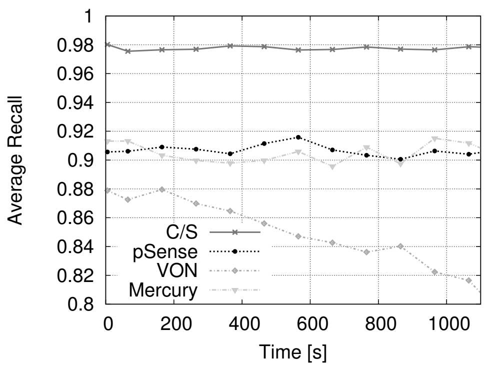
recall is dropping over time as the frequent jumps of nodes causes VON to loose those contacts and it becomes unable to reintegrate a node into the virtual world after using a portal. This loss in connectivity is also visible in Figure 5(c), which shows the Global Connected Component Factor $gccf(t)$ as function of time. VON is constantly loosing connectivity over time due to avatars using the portals. In contrast, the recall of Mercury and pSense remain constant at a level of 0.9 and remain fully connected with a Global Connected Component Factor of 1.

When looking at the position error $\overline{err(t)}$, all systems remain constant at a certain level. VON shows the best position error with about 3 units followed by C/S approach with an average position of about 5 units. This position errors are close to the theoretical minimum given the speed of 20 units/s and an update frequency of 5/s.

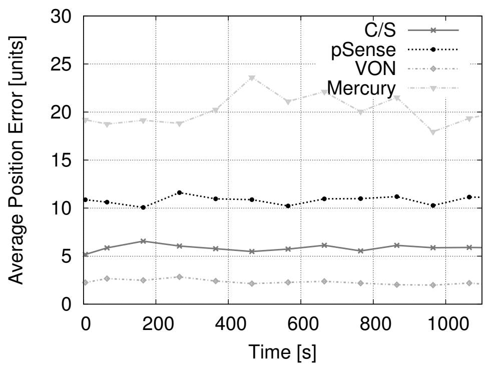
When comparing the position error results with the results obtained during the churn workload, C/S, pSense, and VON provide the same performance. Only Mercury shows a position error of twice the value comparing to the value from the churn workload.

From the upload traffic point of view, which is shown in Figure 5(d), pSense again consumes the most upload bandwidth due to updates being sent redundantly to surrounding neighbors.

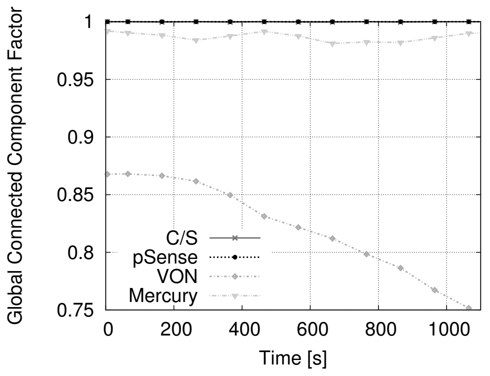
Figure 6 shows the results for the single point movement model where all players in the game world move towards a single point and meet there at the same time. We configured the movement model such that one contraction cycle (minimum density $\rightarrow$ maximum density $\rightarrow$ minimum density) takes 10 minutes of simulated time. This results in a much lower node speed of about 3 units/s in comparison to the speed in the



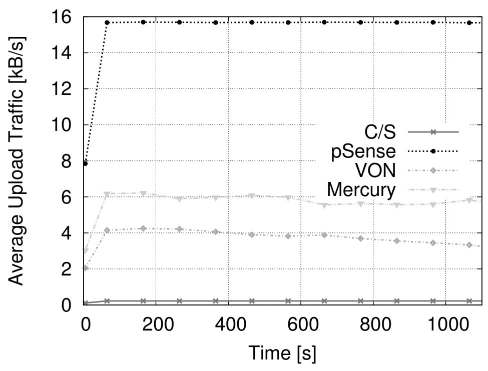
(a) Average recall with portal effects enabled



(b) Average position error with portal effects enabled



(c) Global Connected Component Factor with portal effects enabled



(d) Average recall for the nodes neighbor set for different levels of message loss

Figure 5. Selected results of the stability benchmark using the portal workload scheme

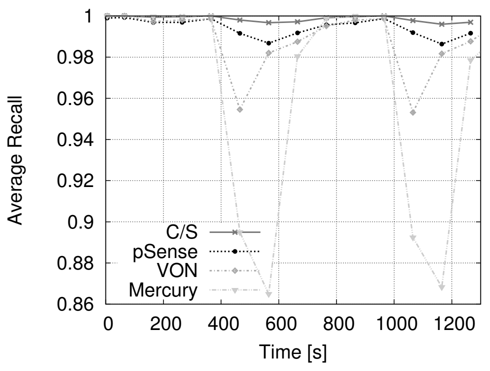
baseline benchmark. When investigating the recall of the four systems, we observe that VON's and Mercury's recall degrade in the moment of maximum density of nodes ($t = 300s$). When the nodes de-contract, resulting in a dropping node density, VON and Mercury are able to recover, resulting in an increasing recall of almost 1 and a position error drop of about 1 unit. Looking at the up- and download utilization, which is shown in Figure 6(c) and 6(d), we observe that VON shows the highest sensitivity to the node density induced by the single point movement model. Again, pSense fully utilizes the up- and downlink bandwidth. In contrast, Mercury's bandwidth usage remains almost constant at 6 kB/s. As one would expect, all three P2P-based systems show a symmetrical up- and download usage, whereas the clients' bandwidth utilization of the C/S approach is highly asymmetrical.

D. Robustness

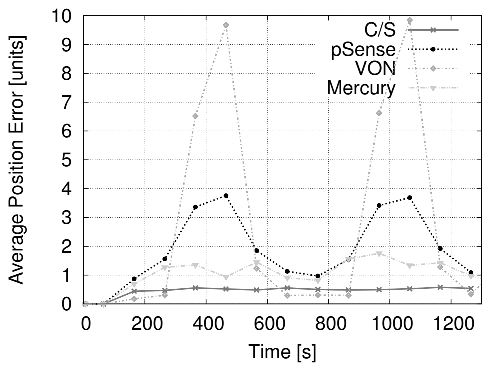
Figure 7(a) shows the recall as a function of time ($\overline{err}(t)$) in the 100% join scenario. While pSense shows a small temporary drop in recall, VON and Mercury are not able to handle the increase of nodes and level out at around 60%. Figure 7(b) shows the CDF of the position error right after the the massive join. VON's position error is much higher and also more widespread than the position error of the other systems. The higher spread can also be observed as a lower value of $F(err)$ in Table IV. Finally, Figure 7(c) shows the recall in the 50% crash scenario. Again, pSense's recall drops only for a few seconds. But both VON and Mercury drop and remain at a significantly lower level than they would be able to achieve with 250 nodes without a crash (cf. Fig. 3(b)).

E. Discussion of Results

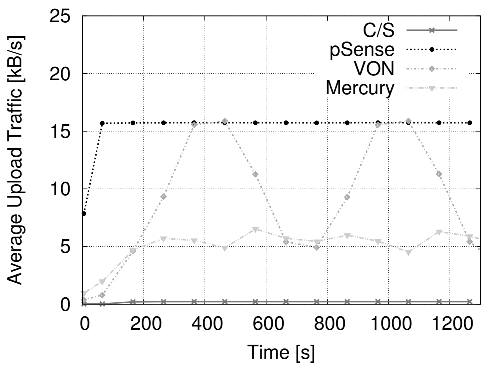
Having presented the simulation results, we now compare our findings with the results and evaluation methodologies from the respective papers. When looking at the evaluation section of the original VON paper [6], the authors evaluated their approach with respect to scalability, validity, and robustness, which the authors call reliability. For evaluating the scalability of their approach the authors increased the number of nodes in the game world and measured the load on the nodes in the system. Besides looking into the number of nodes, the speed of the nodes also has a significant impact on the system performance but this has not been considered in the original paper. The authors only mention that a "problem arises when a node moves faster than what boundary neighbors can notify". Using our benchmarking approach, however, we have shown that the performance of VON degrades significantly at higher levels of node speed. Furthermore, the mobility model also has a significant impact on the the robustness, stability, and scalability capabilities of an NVE overlay and should also be varied for the performance evaluation. Looking into robustness, VON was evaluated under different levels of



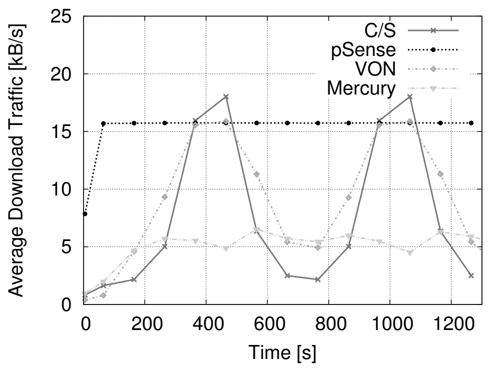
(a) Average recall for the single point movement model



(b) Average position error of nodes for the single point movement model



(c) Average upload bandwidth usage for the single movement model



(d) Average download bandwidth usage for the single movement model

Figure 6. Selected results of the stability benchmark using the single point mobility scheme

message loss but lacks an evaluation under massive join and leave of peers, which also plays an important role and has shown to be a discriminating factor.

Comparing results and methodology of the pSense paper with our results, it is noticeable that pSense has been evaluated with respect to the scalability by increasing the number of nodes. In addition, the authors considered the performance vs. cost trade-off. Again, the overlay has not been evaluated under an increasing speed of the nodes or a varying mobility model. Furthermore, the authors did not look into the stability and robustness capabilities of pSense by considering churn or an unreliable underlay (message loss or varying delay).

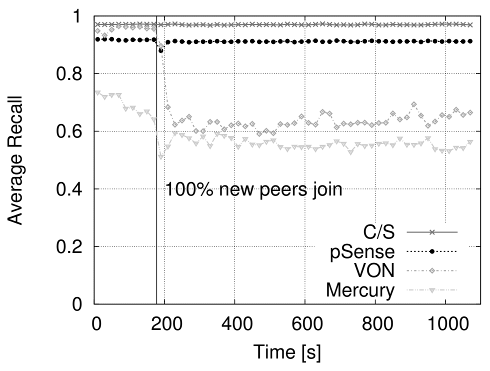
Mercury has been evaluated with respect to scalability by increasing the number of nodes in the system up to 100 peers. The authors measured the normalized load on the nodes as well as the delivery delay. The latter is correlated to our position error metrics. In their evaluation the authors assume an idealized environment with no churn or message loss. Thus, the stability and robustness capabilities of the system have not been evaluated in detail.

In summary, the evaluations lack a systematic approach. The systems have been evaluated under conditions that do not stress the systems to reveal their weaknesses. Workloads such as heavy churn, massive join and leave, or an increasing node speed are not considered. Furthermore, the metrics used for evaluation differ, making a comparison difficult.

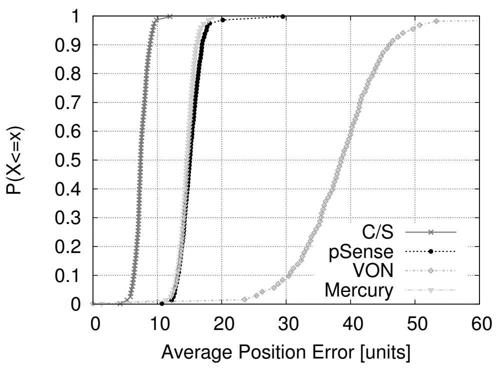
V. Conclusions and Future Work

In this paper, we have presented a benchmarking methodology for spatial information dissemination overlays for Networked Virtual Environments. We established the feasibility of our approach by applying the benchmarking methodology to four typical systems for spatial information dissemination: Mercury, VON, pSense, and a simple client-server implementation. Having presented the evaluation results in Section IV, we can conclude that in situations with a rather low workload, VON provides a better performance than pSense or the simple client server implementation with respect to the position error observed. On the other hand, VON suffers from issues in terms of high node densities, high node velocities, and high node churn rates, whereas pSense offers a degree of robustness comparable to the client-server approach. Because of its multi-hop message delivery scheme, the DHT-based system Mercury shows comparably high position errors in normal situations. But more importantly, its recall drops quickly with increasing node densities or velocities. Mercury is, thus, only applicable in applications with modest requirements. In summary, we were able to quantify performance drawbacks, which are caused by flaws in the design of the examined systems.

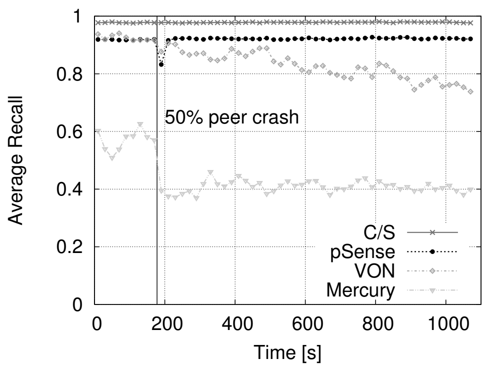
Our long-term goal is to establish a systematic approach to P2P performance evaluation. Follow-up work to this paper will be a detailed analysis of the behavior of players in MMOG to



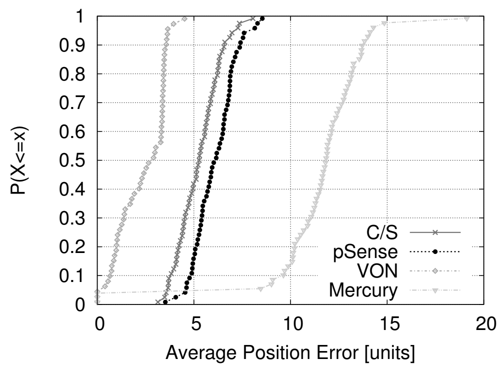
(a) Average recall with massive join from 250 to 500 nodes



(b) CDF of the average position error after a massive join from 250 to 500 nodes



(c) Average recall with massive crash from 250 to 125 nodes



(d) CDF of the average position error after a massive crash from 250 to 125 nodes

Figure 7. Selected results of the robustness benchmark using the massive join and massive leave workload

enhance the realism of the user models. These models will be used to generate more realistic workloads for future versions of this benchmark. For the metrics, we want to elaborate tolerable thresholds based on studies of real games. Eventually, it is desirable to introduce a mapping between quality of service as investigated in this paper and quality of experience as a measure of user satisfaction.

References

[1] A.R. Bharambe, S. Rao, and S. Seshan. Mercury: A Scalable Publish-Subscribe System for Internet Games. In *NETGAMES*. ACM, 2002.

[2] S. Blanas and V. Samoladas. Contention-Based Performance Evaluation of Multidimensional Range Search in Peer-to-peer Networks. *Future Generation Computer Systems*, 25(1):100–108, 2009.

[3] B. Butnaru, F. Dragan, G. Gardarin, I. Manolescu, B. Nguyen, R. Pop, N. Preda, and L. Yeh. P2PTester: a tool for measuring P2P platform performance. In *ICDE 2007*. IEEE, 2007.

[4] K. Chen, P. Huang, C. Huang, and C. Lei. Game traffic analysis: an MMORPG perspective. In *NOSSDAV*. ACM, 2005.

[5] L. Fan, P. Trinder, and H. Taylor. Design Issues for Peer-to-Peer Massively Multiplayer Online Games. *International Journal of Advanced Media and Communication*, 4(2):108–125, 2010.

[6] S. Hu and T. Chen. VON: A Scalable Peer-to-Peer Network for Virtual Environments. In *IEEE Network*, volume 20, 2006.

[7] R. Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc, 1991.

[8] R. Jain, D.M. Chiu, and W.R. Hawe. *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems*. Eastern Research Laboratory, Digital Equipment Corp., 1984.

[9] B. Knutsson, H. Lu, W. Xu, and B. Hopkins. Peer-to-Peer Support for Massively Multiplayer Games. In *INFOCOM*, 2004.

[10] A. Kovacevic, K. Graffi, S. Kaune, C. Leng, and R. Steinmetz. Towards Benchmarking of Structured Peer-to-Peer Overlays for Network Virtual Environments. In *ICPADS*. IEEE, 2008.

[11] A. Kovacevic, S. Kaune, N. Liebau, R. Steinmetz, and P. Mukherjee. Benchmarking Platform for Peer-to-Peer Systems. *it - Information Technology*, 49(5), 2007.

[12] M. Lehn, T. Triebel, C. Gross, D. Stingl, K. Saller, W. Effelsberg, A. Kovacevic, and R. Steinmetz. *From Active Data Management to Event-Based Systems and More*, volume 6462 of *LNCS*, chapter Designing Benchmarks for P2P Systems. Springer, 2010.

[13] J. Li, J. Stribling, R. Morris, M.F. Kaashoek, and T.M. Gil. A Performance vs. Cost Framework for Evaluating DHT Design Tradeoffs Under Churn. *IEEE INFOCOM*, 1, 2005.

[14] T. Neumann, M. Bender, S. Michel, and G. Weikum. A Reproducible Benchmark for P2P Retrieval. In *EXPDB*. ACM, 2006.

[15] T.S.E. Ng and H. Zhang. Towards Global Network Positioning. In *IMC*. ACM, 2001.

[16] C. Nocentini, P. Crescenzi, and L. Lanzi. Performance Evaluation of a Chord-based JXTA Implementation. In *AP2PS*. IEEE, 2009.

[17] B. Questier, M. Jan, and F. Cappello. One Step Further in Large-scale Evaluations: the V-DS Environment. Technical report, INRIA, 2007.

[18] A. Schmieg, M. Stieler, S. Jeckel, P. Kabus, B. Kemme, and A. Buchmann. pSense-Maintaining a Dynamic Localized Peer-to-Peer Structure for Position Based Multicast in Games. In *P2P*. IEEE, 2008.

[19] D. Stingl, C. Gross, J. Rückert, L. Nobach, A. Kovacevic, and R. Steinmetz. PeerfactSim.KOM: A Simulation Framework for Peer-to-Peer Systems. In *HPCS*. IEEE, 2011.

[20] A.P. Yu and S.T. Vuong. MOPAR: A Mobile Peer-to-Peer Overlay Architecture for Interest Management of Massively Multiplayer Online Games. In *NOSSDAV*, 2005.