To appear in *IEEE Network Magazine*

VON: A Scalable Peer-to-Peer Network for Virtual Environments

Shun-Yun Hu[1], Institute of Physics, Academia Sinica, Taiwan
Jui-Fa Chen[2] and Tsu-Han Chen[3],
Dept. of Computer Science and Information Engineering, Tamkang University

Abstract:

The scalability of large-scale *networked virtual environments* (NVEs) such as today's *Massively Multiplayer Online Games* (MMOGs) faces inherent limits imposed by the client-server architectures. We identify an emerging research direction that applies peer-to-peer (P2P) networks to realize more scalable and affordable NVEs. The central issue for P2P-based NVE (P2P-NVE) systems is to correctly and efficiently maintain the topology of all participating peers by solving the *neighbor discovery problem*. We also propose *Voronoi-based Overlay Network* (VON), a simple and efficient design that maintains the P2P topology in a fully-distributed, low-latency, and message-efficient manner. Simulation results show that by bounding the per-node resource consumption, VON can be *fundamentally* more scalable than existing methods while achieving high topology consistency and reliability.

Keywords:
peer-to-peer (P2P), Voronoi, overlay network, networked virtual environment (NVE), massively multiplayer, online games, MMOG, scalability, interest management

---

[1] E-mail: syhu@yahoo.com
[2] E-mail: alpha@mail.tku.edu.tw
[3] E-mail: bkyo0829@yahoo.com.tw

1. Introduction

Networked virtual environment (NVE) [1, 2], also known as *distributed virtual environment*, is a relatively new field that combines 3D graphics with networking to provide simulated yet immersive experiences for people across the globe. NVEs originated in the '80s with military simulators [3] and had since evolved to today's *Massively Multiplayer Online Games* (MMOGs) [4], where *hundreds of thousands* of people interact simultaneously in fantasy worlds. As of early 2006, the most popular title, *World of Warcraft*, has attracted over 6 million subscribers within two years, while *Lineage*, also has over 4 million subscribers worldwide. MMOG has grown rapidly since its inception in the mid-90s, making the genre a multi-billion dollar business and a major Internet application [5].

Over 100,000 peak concurrent users are frequently reported for major MMOGs, making *scalability* a defining trait. NVE scalability is fundamentally a resource issue, and is determined by whether the system has enough resources to continuously accommodate new participants. However, existing server-based architectures pose inherent scalability limits and significant costs. Though much effort has been spent on improving scalability, no real system yet exists that could feasibly support over *one million* concurrent users. The popularity and rapid growth of large-scale NVEs such as MMOGs likely will pose serious challenges to existing networks and architectures in the near future. On the other hand, a new class of peer-to-peer (P2P) applications that seeks to realize large-scale NVEs has recently appeared as an alternative.

P2P applications are distributed systems where each node runs the same software [6]. It differs from *grid computing* in that the basic units are commodity PCs instead of well-provisioned facilities. Although the early P2P systems adopt point-to-point topologies that scale poorly, P2P file-sharing made of millions of nodes (e.g. *Gnutella* and *Kazaa*) has publicized *scalable P2P systems*, which also include significant recent research on *distributed hash table* (DHT) [6], where a given key is mapped to a node and may be searched efficiently (allowing, for example, finding files without any centralized directories). P2P systems promise scalability and affordability by bringing user-resources to the system and are composed of commodity hardware. Yet they also create new challenges such as *topology maintenance* (as nodes can join or leave at any time) and efficient *content search* (as the global topology is unknown to each node).

This article extends our previous work [7] and identifies an emerging field that applies P2P techniques to realize NVEs on scales unattainable by existing architectures. A simple and efficient design based on Voronoi diagrams [8], called *Voronoi-based Overlay Network* (VON), is also presented. VON exploits the locality of user interests to maintain the P2P topology with small overhead, and achieves high scalability by *bounding* resource use at each participating node.

The contributions of this article are the analysis of the NVE scalability problem, the identification of the emerging P2P-based NVE (P2P-NVE) research, and the presentation of Voronoi-based Overlay Network. In the following sections, we first provide a background on NVE systems, and explain the scalability problem with respect to P2P adaptation, followed by examinations of recent P2P-NVE proposals. We then explain the design rationales and procedures of VON, and demonstrate its scalability, consistency and reliability with simulations. Discussions of VON's properties and comparisons with other systems are made, before we conclude this article with a summary and some future perspectives.

1.1 Characteristics of NVE

NVEs are synthetic worlds where each user assumes a virtual identity (called *avatar*) to interact with other human or computer players. Users may perform different actions such as moving to new locations, looking around at the surroundings, using items, or engaging in conversations and trades. Actions are encoded and transmitted with specialized messages over the network to notify other users. Message contents and transmission protocols are usually application-specific and proprietary.

A number of issues are involved in building a NVE system, including:

Consistency: As NVEs are shared environments, it is important that each participant perceives the same states and events. Consistency in NVEs usually refers to consistent *object states* and *event orderings* [1], which are maintained through the transmission of event messages. However, event messages may be delayed or lost, making consistency a challenging issue.

Responsiveness: To provide a sense of natural interactions, the delay between a user's action and observable consequence should be minimized despite network latency. NVE systems thus have a *real-time* requirement. However, latency tolerance can be application-dependent [9]. Responsiveness and consistency are usually seen as tradeoffs, as waiting for late or retransmission of missing packets results in delays.

Scalability: The true potentials of NVEs lie in networking large number of people, which makes scalability a key aspect for future NVEs. It is also important in terms of service availability (i.e. whether systems can sustain rapid usage increase) and content possibilities (i.e. large participant size enables new types of designs and interactions).

Persistency: Contents within NVEs (e.g. virtual objects and computer characters) may need to exist persistently to provide a sense of realism. Current MMOGs provide persistent-world experiences that operate 24 hours a day, where users can keep virtual items and player profiles between login sessions via centralized database storage.

Reliability: As NVEs become scalable and persistent, they will also need to sustain hardware or software failures, in order to ensure service availability and smooth user experiences. Reliable and fault-tolerant systems will increase in importance as NVEs become more service-oriented.

Security: Many types of NVE maintain user accounts and allow competitions between users. Security is therefore important to ensure fairness during interactions and privacy of user information. It is especially emphasized in commercial systems.

Our focus is on scalability, as it allows new design possibilities. Responsiveness and reliability are also discussed, however, consistency is only considered for user position states, while persistency and security are beyond our scope. Two main types of resources in NVE systems are processing power and bandwidth [1]. As bandwidth tends to be the main resource bottleneck, it will be our focus (see p.228 in [4]).

1.2 MMOG vs. NVE

MMOG is an important and arguably the most successful subset of NVE. To provide an entertaining experience, MMOGs focus more on game designs, continuous supplies of game contents, and security than other types of NVEs [2, 4]. *Game states* for users, virtual items, terrain, and computer-controlled characters are maintained at the servers. Typical network messages include environmental changes (e.g. door opening), movements, interactions (e.g. chatting, trading, combating), and system services (e.g. login, friends search, software or content updates) [4, 9]. However, in this article we will focus on *position updates*, as they represent the most fundamental and largest share of message traffic [10] (see also p. 278 in [4]).

2. The Scalability Problem

Scalability in this article refers to whether a NVE system can accommodate a wide range of concurrent users without compromising certain quality of service (e.g. sufficient message updates that ensure consistency or responsiveness). The issue at heart is whether system resources do not deplete as new users join. In a generalized NVE, users can be conceptually seen as coordinate points on a 2D plane (as *nodes*), each with a visibility range called *Area of Interest* (AOI). AOI is a fundamental NVE concept, as even though many users and events may exist in the system, each user, as in the real world, is only affected by nearby users or events. AOI thus specifies a scope for information which the system should provide to each user. For simplicity, we assume that AOI is a circle centered on the user, and nodes move at discrete *time-steps*. The central communication problem thus is: *how does each node receive relevant messages (generated by other nodes) within its AOI as they move around?*

2.1 Evolution of scalability solutions

The earliest solution to this problem is to simply *broadcast* updates, or connect everyone in a *point-to-point* manner. SIMNET and DIS [3] are the classic examples. Broadcast is applicable only to LANs, while point-to-point is not scalable as connections grow at $O(n^2)$ with user size. *Client-server* architectures improve the situation, where each user communicates with a centralized server (e.g. MASSIVE-3 [11]). Relevant messages are relayed by the server, reducing message traffic to $O(n)$ (Fig. 1a). However, the server becomes the bottleneck and its resources availability dictates the system's scale. To increase server-side resources, *server-clusters* are devised to combine multiple servers on a high-speed LAN with large amount of bandwidth for external connectivity (e.g. [12], RING [13]).

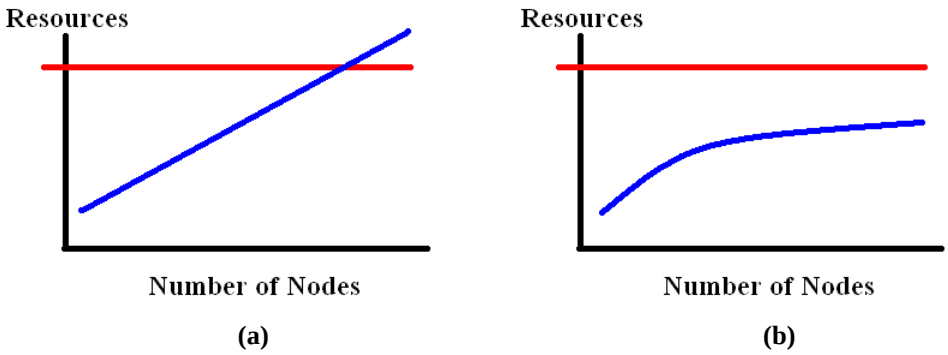


Figure 1: Scalability analysis

X-axis denotes system size; y-axis denotes the resource use at the *bottleneck node* (i.e. the node that first depletes its resources); top line is the resource limit. (a) Linear growth of resource use in client-server (per server). (b) Bounded growth of resource use in multicast groups (per node).

Contrary to “increase resources” methods, the “decrease consumption” strategies try to reduce message transfers by packet *compression* or *aggregation* (i.e. combining several messages into a coarse representation [14]), or through *interest management* [15], which seeks to filter out irrelevant messages (i.e. those generated outside a user’s AOI should not be received).

Interest management is of particular importance, as compression or aggregation does not exclude irrelevant messages, while *ideal* interest management prevents messages outside the AOI from ever being received and processed. Most interest management schemes to date utilize *multicast* (e.g. NPSNET [16], DIVE [17], Spline [18], MiMaze [19], and VELVET [20]). In a typical scenario, the NVE is partitioned into some messaging *groups* (e.g. spatially divided *regions*, functionally divided *classes* such as soldiers and aircrafts [16], or *locales* [18]), each with a pre-specified multicast address. A node sends messages to the address of its residing group, and subscribes for groups that coincide with its AOI. Ideally each node would then only process a finite amount of relevant messages, making the system more scalable (Fig. 1b). Various designs mainly differ in the grouping methods and abstraction that is assigned an address (e.g. NPSNET divides the world into hexagonal cells, while VELVET assigns each user a multicast address). Multicast has two major benefits: messages are sent only once to the network, and are filtered by the infrastructure without host processing. However, the messages received may be irrelevant or insufficient if group scopes are improperly defined [21]. The most serious problem, however, is that multicast has not been widely deployed and relatively few addresses are available for large-scale applications.

Application-layer multicast [22] has been proposed to address the lack of network-layer multicast by building *overlay networks*. Nodes self-organize into multicast trees and deliver messages to many recipients via node relays. However, defining the appropriate multicast groups is still challenging, while the latency due to relays may not be appropriate for NVE’s real-time requirements.

2.2 Scalability bottlenecks

Today's MMOGs overwhelmingly adopt server-clusters due to well-supported security and persistency mechanisms. They nevertheless face three scalability issues: 1) insufficient total resources 2) high user density, and 3) excessive inter-server communications. The first issue is dealt by provisioning more hardware. However, MMOG traffic analysis [10] shows that human behaviors cause bursty traffic, so instantaneous resource use can be much higher than average. Over-provisioning thus is likely to happen. Even with enough total resources, users would often concentrate at *hotspots* where interesting events occur (known as user *crowding* or *flocking*), and overload the servers handling the hotspots. Though server-cluster load balancing has been proposed [23], migrating users to other servers may induce the third issue of excessive inter-server communications that consume bandwidth and introduce latency (e.g. when transferring user profiles across servers, or when supporting interactions between users on different servers). P2P designs may alleviate these issues because: 1) resources are brought into the system with each additional node joined; 2) nodes may individually determine and reduce their messaging scopes when user densities increase; 3) message exchange with only the relevant (i.e. AOI) neighbors may effectively restrict inter-node traffics.

2.3 P2P-based NVE

Due to NVE's unique requirements, we argue that P2P-NVE poses as a new class of P2P systems worthy of study. One notable difference between P2P-NVE and DHT or file-sharing is that the problem of content search is greatly simplified: as each node's AOI is limited, the desired content is *localized* and easily identified as the messages generated by other nodes within the AOI. This differs from file-sharing where the desired content may potentially be on *any* node. The central *content search* in P2P systems thus becomes a *neighbor discovery problem*, and may be stated as: *given a node's position and AOI, find all neighboring nodes within the AOI*. NVE design criteria also transform into the following issues and metrics:

Consistency: As consistency issues other than user positions are outside our scope, here we focus on *topology consistency* [24], which can be defined as the percentage of correctly known AOI neighbors (e.g. for a node that is aware of 4 out of 5 AOI neighbors, topology consistency is 80%). Another useful metric is *drift distance* [19], defined as the distance (in absolute value) between observed and actual coordinates of a node. Two closely related concepts are *global connectivity* (i.e. whether the overlay is fully-connected) and *local awareness* (i.e. whether each node is fully aware of its AOI neighbors) [25]. Ensuring these properties are the challenges for any P2P-NVEs.

Responsiveness: To fulfill the real-time requirement, message latency should be minimized while bandwidth use should be efficient. Ideally, nodes should connect *directly* to their AOI neighbors to minimize latency.

Scalability: In a P2P environment, the key scalability issues are whether resource use is *bounded* at each node, and if potential traffic bursts can be handled. Average and maximum transfer-sizes on each node within a period are thus useful indicators.

Reliability: As packet loss, delay, and node failures can adversely affect topology consistency, robustness against these problems is the main reliability issue. *Recovery steps*, defined as the number of time-steps to recover to 100% topology consistency, may be a useful metric.

P2P-NVEs can potentially be more scalable, responsive, and reliable than server-based solutions, while it remains as future research topics to provide adequate support for consistency, persistency, and security in a P2P environment.

2.4 Recent proposals

We now describe and categorize some recent P2P-NVE proposals:

Enhanced Point-to-point: Novel message filtering techniques have been devised for point-to-point architecture where all nodes know each other. The idea is to allow transmission only under mutual visibility. *Update-free regions* (UFR) [26] define pairs of mutually invisible regions in the NVE. Message exchange is not needed if two nodes stay within their respective UFRs, and only resumes if one of them steps out. Then, either the UFRs are renegotiated (if visibilities are still blocked) or position exchange begins. However, UFR does not scale as each node needs to negotiate with *all* other nodes. Also, excessive message exchanges can happen if many neighbors are mutually visible (e.g. when sights are not blocked or crowding occurs). *Frontier sets* [27] provide an improved approach of UFR but suffer from the same limitations.

DHT-based: Significant research efforts were into interest management via multicast. As multicast is not widely used, *application-layer multicast* provides a workaround. *SimMud* [9] uses *Scribe*, an application-layer multicast built on the DHT *Pastry*, and divides the NVE into some fixed-size regions. Each region is managed by a promoted *super-node*, which serves as the root of a multicast tree. A super-node receives position updates from all regular nodes within a region and relays the updates to them. Links are maintained between super-nodes to aid user transitions across regions. However, a super-node can be overwhelmed by crowding in the region, and message latency may be up to several seconds due to relays. To reduce latency, *Zoned Federation* [28] uses DHT only for topology connectivity, while regular nodes connect directly to the region's super-node. However, crowding can still be a concern.

Neighbor-list exchange: To discover neighbors without resorting to centralized servers or super-nodes, knowledge from existing neighbors may be exploited. Kawahara *et al*. describe a fully-distributed scheme [24] where each node directly connects with a fixed number of nearest neighbors and constantly exchanges neighbor-lists to discover new nodes. Although direct connections minimize latency, constant list exchanges incur transmission overheads. When separated by large distances, nodes may also lose mutual contacts and cause *overlay partitions* [24]. To ensure global connectivity, the *Message Interchange Protocol* (MIP) [29] keeps at least one neighbor in each of the four cornering directions of a node. However, the frequency of list exchange is still a delicate tradeoff between neighbor discovery timeliness and bandwidth use.

Mutual notification: Neighbor list exchange may not be bandwidth-efficient, as much of the transmission is redundant. A better approach is to notify new neighbors only when necessary. *Solipsis* [25] is another fully-distributed system, where each node connects to all the neighbors within its AOI. Neighboring nodes serve as the "watchmen" for approaching foreign nodes and neighbor discovery is achieved by notifications from known neighbors. As Solipsis maintains direct connections among neighbors, latency is minimized. To ensure global connectivity, each node needs to be inside the polygon (i.e. a *convex hull*) formed by its outmost neighbors. However, neighbor discovery is occasionally incomplete, as incoming nodes may be unknown to directly-connected neighbors. In other cases, active queries are required when the within-convex-hull property is violated, which slows down neighbor discovery [30].

Client-server hybrid: *Federated peer-to-peer* [31] is a hybrid between client-server and P2P architectures. Nodes organize into various groups and are managed by provisioned hardware called *multicast reflectors*. Groups are created or modified by *control servers*, from which participating nodes obtain mappings between the groups and multicast reflectors. Nodes subscribe and send updates to multicast reflectors by individual interests, and discover neighbors via messages relayed by the multicast reflectors. Strictly speaking, client-server hybrids are server-cluster variants where the combined bandwidth of specialized servers determines the scalability limit.

P2P hybrid: *MOPAR* [32] is an interesting mix of DHT, neighbor-list exchange, and direct transfer. The NVE is divided into hexagonal cells (like NPSNET), and DHT is used to ensure global connectivity. Within each cell, a *master node* maintains a list of *slave nodes* and exchanges the list periodically with neighboring masters. Slave nodes are notified of new neighbors by the masters. To avoid latency, slave nodes exchange messages directly between themselves. However, the selection of adequate masters is a concern, especially when crowding is considered. The frequency of slave-node list exchange is also a difficult balance between bandwidth use and neighbor discovery timeliness, as in other neighbor-list exchange schemes.

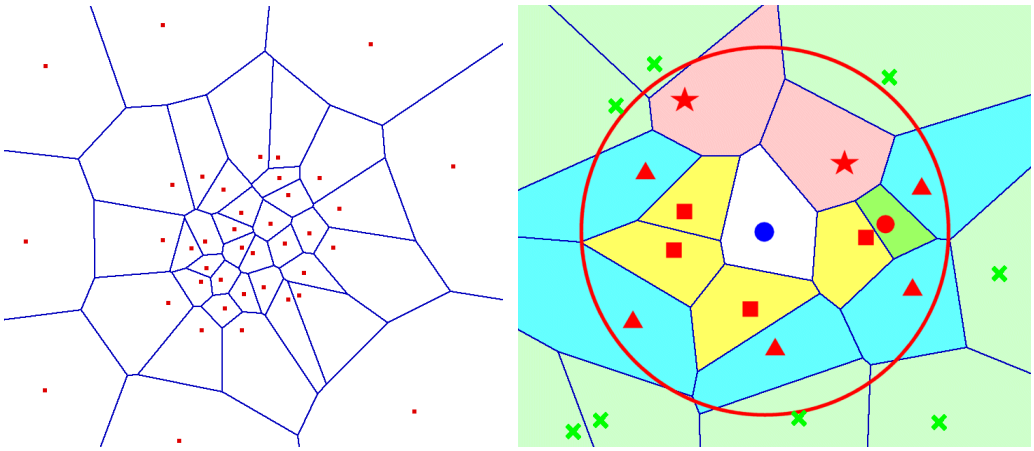
Various P2P-NVE schemes differ in their degree of decentralization. In general, the more distributed the design, the more responsive the system becomes, however, at the expense of increased bandwidth utilization. Super-nodes may use less bandwidth at the cost of worse tolerance to node failures and crowding. Other differences include neighbor discovery timeliness and event ordering support. Although these designs are potentially more scalable than client-server, correct and timely neighbor discovery and user crowding tolerance are still the common issues.

3. Voronoi-based Peer-to-Peer Design

We propose the use of *Voronoi diagrams* [8] to solve the neighbor discovery problem in a fully-distributed, bandwidth-efficient, and low-latency manner. Our design objectives are: (a) *scalability*, achieved by limiting per-node transmission; and (b) *responsiveness*, achieved by requiring direct connections for all data exchanges.

3.1 Voronoi explained

Given a number of points (called *sites*) on a 2D plane, a Voronoi diagram partitions the plane into the same number of *Voronoi regions* (or simply *regions*), such that each region contains all the points closer to the region's site than to any other site (Fig. 2a). Voronoi diagram defines a spatial relationship between various nodes and may be used to find the *k-nearest neighbor* of any site efficiently. Since Voronoi diagrams have been widely studied and applied to diverse fields (e.g. computational geometry and mobile computing), we do not consider the specifics of Voronoi construction but assume that good algorithms exist.



(a) **(b)**

Figure 2: Voronoi diagram

(a) Dots indicate *sites*, and lines define boundaries (*edges*) for *regions*. (b) The large circle is the AOI boundary for the center node. Squares (■) are *enclosing neighbors*; triangles (▲) are *boundary neighbors*; stars (★) are *both* enclosing and boundary neighbors; circle (●) represents a regular *AOI-neighbor*; crosses (✕) represent neighbors irrelevant (i.e. outside of AOI) to the center node.

3.2 Design of VON

For our purpose, each node in VON is represented as a site in the Voronoi diagram. For a given node, we define *AOI neighbors* as the nodes whose positions are within its AOI. *Enclosing neighbors* are nodes whose regions immediately surround the given node, and *boundary neighbors* are AOI neighbors whose enclosing neighbors may partially lie outside the AOI (Fig. 2b). Each node maintains a Voronoi

diagram of all AOI neighbors and directly connects them to minimize latency. As only a few neighbors are kept, the cost to maintain a Voronoi diagram at each node is low. To prevent *overlay partition* (i.e. groups of nodes become mutually unaware of each other), we also require each node to *minimally keep its enclosing neighbors* (which may be outside the AOI when neighboring nodes are sparse).

When a node moves, position updates are sent to all connected neighbors (i.e. AOI neighbors plus any enclosing neighbors beyond AOI). Neighbor discovery is via notifications from boundary neighbors, as they know both the moving node and other nodes beyond the AOI (which happen to be their enclosing neighbors). This way, potential AOI neighbors are discovered with mutual collaborations. As a node moves around, it will constantly discover new nodes and disconnect those that have left its AOI (unless they are enclosing neighbors). A node thus restricts communications with mostly the actual AOI neighbors, *independent of the scale of the system*. Keeping bandwidth consumption at each node *bounded* is the key to VON's scalability.

3.3 Dynamic AOI adjustment

One potential concern is that when *crowding* occurs (i.e. user gatherings that cause high densities of nodes, see the center of Fig. 2a), nodes are forced to connect beyond their capacities. Oliveira and Georganas introduced the notion of dynamically adjustable AOI [20], so that nodes with different bandwidths can individually determine their optimal AOI-radii. We also apply dynamic AOI adjustments with three simple rules: 1) a node shrinks its AOI when the number of connected neighbors exceeds a pre-specified *connection limit*; 2) AOI restores to the preferred size if the number of connected neighbors again falls below the limit. 3) To ensure mutual awareness between nodes, a node will not disconnect a neighbor whose AOI still covers it, even if that neighbor is outside of its own AOI.

3.4 VON procedures

We now describe VON's three main procedures, please see [30] for pseudocodes.

JOIN procedure

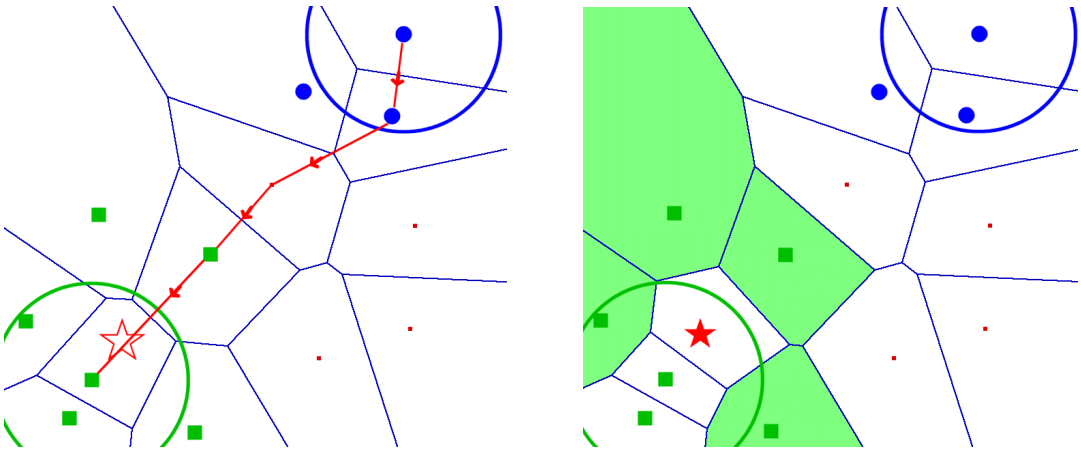
A *joining node* first contacts the *gateway server* (i.e. the first bootstrapping node in VON) for a unique ID, then sends a query with its joining coordinates to any existing node (which can be the *gateway*). Using greedy forward, the query will eventually reach the *acceptor* (i.e. the node whose region contains the joining node's coordinates, see Fig. 3a), which responds by sending a list of joining node's *AOI* and *enclosing neighbors*. The *joining node* then connects to each neighbor on the list and organizes their coordinates into a Voronoi diagram. The neighbors also update their Voronoi diagrams to account for the *joining node* (Fig. 3b).

MOVE procedure

When a node moves, position updates are sent to *all* currently connected neighbors. If the recipient is a *boundary neighbor*, it will check to see if any of its *enclosing neighbors* is now visible to, or has become the *enclosing neighbor* of, the *moving node* (Fig. 4a). If new neighbors are identified, notifications are sent to the *moving node* for initiating connections. During handshakes with a new neighbor, the *moving node* also sends its knowledge of the *enclosing neighbors* of the new neighbor, which would verify and notify the *moving node* of any missing ones. The *moving node* will also disconnect any *boundary neighbors* that have left its AOI (Fig. 4b). During each time-step, Voronoi diagrams are updated at all nodes to reflect the current topology.

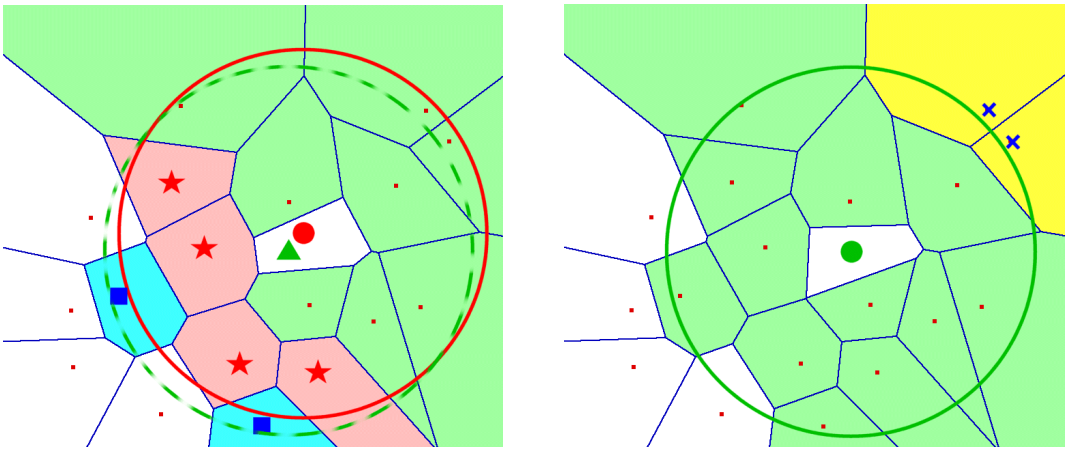
LEAVE procedure

The *leaving node* simply disconnects (no distinction is made between proper and abnormal departures from the overlay network, see Fig. 5a). Affected neighbors will update their Voronoi diagrams. In case of *boundary neighbor* departures, replacements are learned via other still-connected *boundary neighbors* (Fig. 5b).



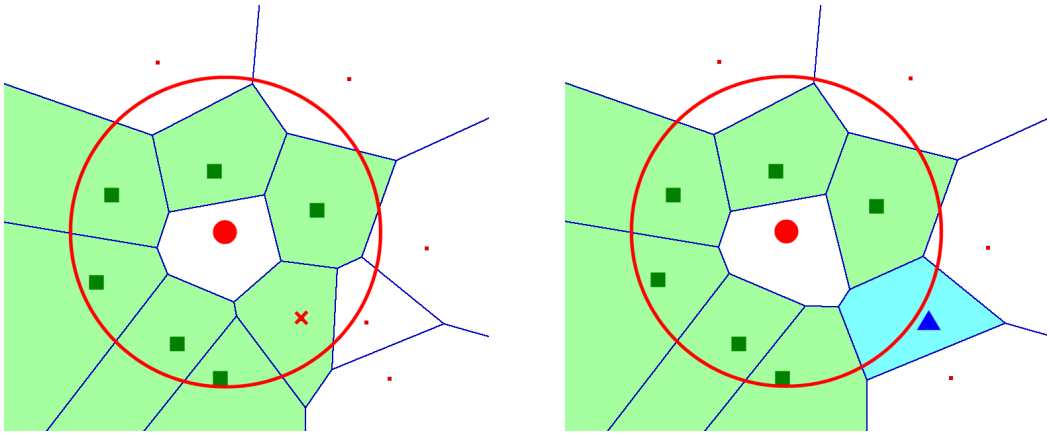
(a) **(b)**

Figure 3: JOIN procedure (a) *Gateway server* and its *enclosing neighbors* are the circles. *Acceptor* and its *enclosing neighbors* are the squares. Star (☆) is the intended joining spot. Join request is forwarded to the acceptor (b) Star (★) is the *joining node*, shades are neighbors affected by join. Note how the changes to Voronoi regions are localized.



(a) **(b)**

Figure 4: MOVE procedure (a) Triangle (▲) indicates the intended new position. Squares (■) are new neighbors about to be discovered. Stars (★) are the *boundary neighbors.* (b) After the move, crosses (╳) are the neighbors no longer inside the AOI, therefore are disconnected.



(a) **(b)**

Figure 5: LEAVE procedure (a) Cross (╳) is the *leaving node*. (b) After leaving, triangle (▲) is the new *enclosing neighbor* discovered by still-connected *boundary/enclosing neighbors* (Squares ■).

3.5 Comparison with Delaunay triangulation overlay

Significant work has been done using *Delaunay triangulation* (the dual structure of Voronoi diagram [8]) to construct overlays for application-layer multicast (*DT-overlay*) [22]. DT-overlay provides a scalable protocol where nodes use only local information for packet forwarding. However, requirements for application-layer multicast and NVE differ in important aspects: the former aims to send data to potentially many recipients, so message delivery is characterized by relays between nodes, yet NVE's real-time trait demands minimal latency (i.e. relays should be avoided if possible). Voronoi diagram (or Delaunay triangulation) thus is used differently in VON (i.e. to discover new neighbors) than in DT-overlay (i.e. to derive routing paths). VON also differs from DT-overlay in at least two respects: 1) it lacks DT-overlay's main drawbacks of high end-to-end latency and multiple transfers of the same data over a physical link [22], as all transmissions are direct; 2) it considers *constant* node movements, whereas there is no equivalent of VON's *MOVE procedure* in DT-overlay's protocol. VON is similar to DT-overlay only if all nodes just connect with their enclosing neighbors and do not move.

4. Evaluation

Voronoi diagrams simplify two difficult identifications in neighbor discovery: the nodes making the discovery decisions and the minimal set of connected neighbors to ensure complete discovery. The former is solved by boundary neighbors, while the latter by enclosing neighbors. To evaluate VON's properties, simulations using both fixed and dynamic AOI are performed (referred to as the *basic* and *dynamic AOI*, or *dAOI model*, respectively, see Table 1). The simulations proceed in over 3000 discrete time-steps, where nodes move randomly with a constant speed. As users in real NVEs do not distribute uniformly, we place up to 2000 nodes within a 1200x1200 area to simulate high density scenarios. After allowing the system to stabilize for 200 steps, we measure the transmission size per node per second, assuming 10 movements per second (for comparison, typical update rates for MMOGs are 1-5 times/second [10], and 10-20 times/second for fast action games). For simplicity, we assume constant latency for all simulations (i.e. messages sent are processed in the next time-step).

The simulation results are presented with 95 percent confidence intervals and we discuss VON's properties below:

Scalability: Fig. 6 shows that the average transmissions per node per second grow linearly for the basic, and *sub-linearly* for the dAOI model. However, transmissions level off if node density is fixed after 1000 nodes (i.e. if the world-size grows accordingly). Note that the measurements exclude packet headers and that under the zero packet loss assumption, average transmissions are the same for both outbound and inbound traffic. Fig. 7 shows that the maximum transmissions per node per second hold similar patterns, indicating that VON can be highly scalable as per-node resource use is well *bounded*.

Consistency: Table 2 shows that the topology consistency is near 100% for the basic model, and slightly less for the dAOI model (due to sudden adjustments of AOI-radii). However, the inconsistencies are quickly adjusted (within 2 time-steps on average). *Global connectivity* is maintained as all nodes at least keep their enclosing neighbors. Consistency is further confirmed by the almost-zero average drift distances.

Reliability: Our second simulation studies the effects of packet loss. Fig. 8 shows that topology consistency remains above 96% up to 40% loss, and it takes less than 10 steps to recover for loss rates up to 50%. Recovery is possible as missing neighbors are found via regular neighbor discoveries or handshakes with new neighbors (section 3.4). Higher node density and continuous movements may also facilitate the recovery. Note that the loss rate in the figure does not apply to control messages, which occupy about 5% of all bandwidth and are still transmitted reliably. Also, a maximum of 100 recovery steps is imposed on data collection.

Responsiveness: Direct connections with AOI neighbors minimize message latency.

Table 1: Simulation parameters

Parameters	Simulation Type	
	Scalability	**Reliability**
World dimension (units)	1200x1200	1200x1200
Initial AOI-radius (units)	100	100
Packet loss rate	0%	0% ~ 90% (in 10% increment)
Simulation time-steps	3000	5000
Speed (units/time-step)	5	5
Connection limit	No limit for basic, 20 for dAOI	20
Number of nodes	200 ~ 2000 (in 200 increment)	1000

Table 2: Simulation results

Metric	Model	Number of Nodes									
		200	**400**	**600**	**800**	**1000**	**1200**	**1400**	**1600**	**1800**	**2000**
Topology Consistency	**(basic)**	100.000%	100.000%	100.000%	99.999%	99.999%	99.999%	99.999%	99.999%	99.999%	99.999%
	(dAOI)	100.000%	99.992%	99.984%	99.984%	99.985%	99.985%	99.986%	99.986%	99.986%	99.986%
Avg. Drift Distance	**(basic)**	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.01
	(dAOI)	0.00	0.04	0.07	0.08	0.09	0.10	0.11	0.12	0.13	0.14
Avg. Recovery Steps	**(dAOI)**	0.07	1.88	1.91	1.75	1.71	1.56	1.48	1.41	1.32	1.27
Avg. Connection Size		9.68	15.23	17.43	18.35	18.80	19.03	19.19	19.27	19.36	19.43
Avg. AOI-radius		99.97	93.74	80.98	70.87	62.37	55.97	50.70	46.33	42.83	39.60

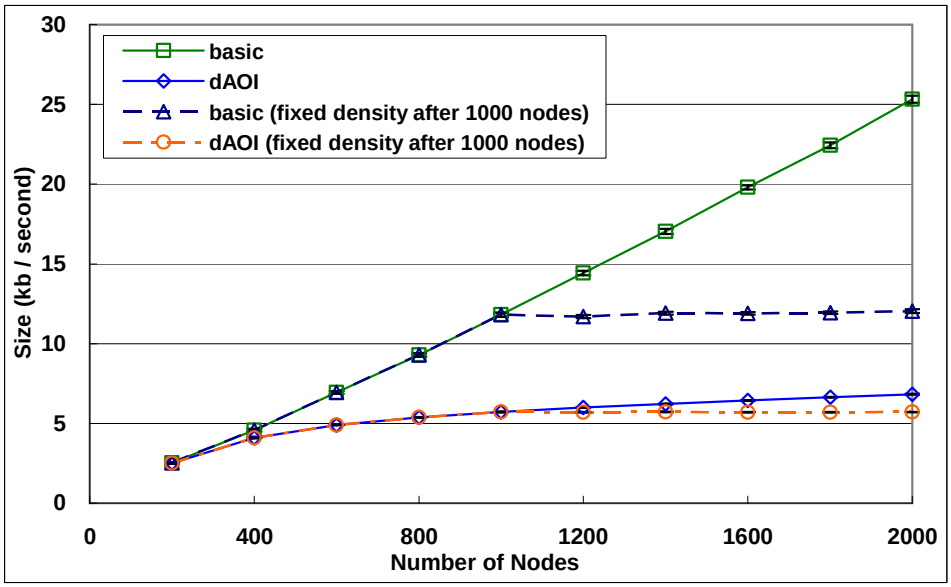


Figure 6: Average transmission size per node per second (no packet loss).

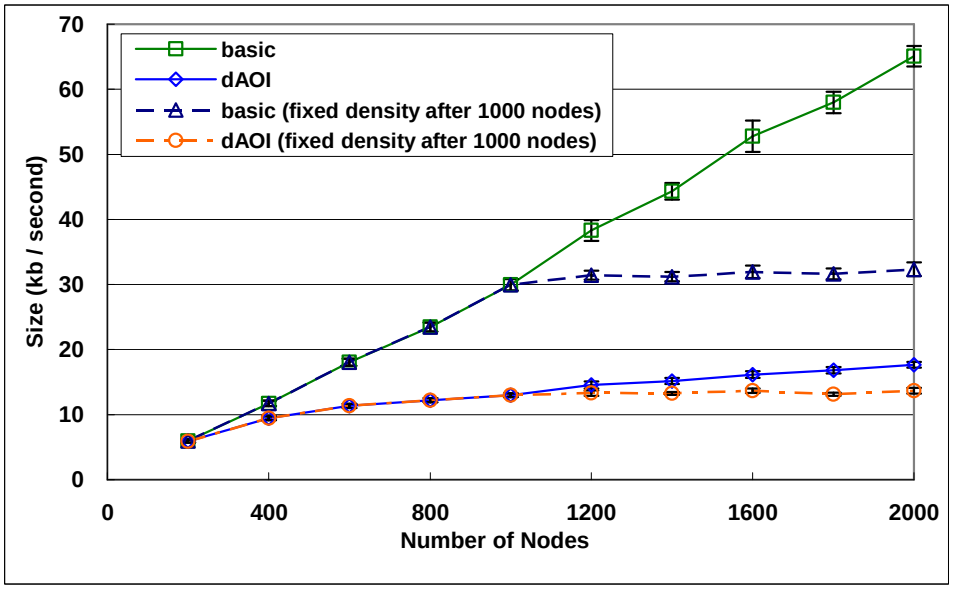


Figure 7: Maximum transmission size per node per second (no packet loss).

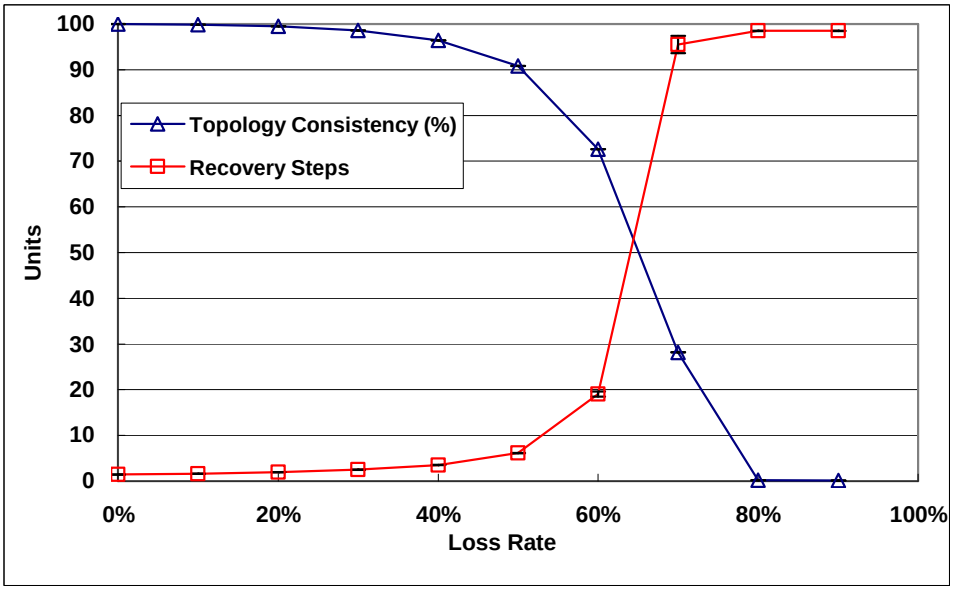


Figure 8: Effects of packet loss (1000 nodes with dynamic AOI)

4.1 Comparisons and problems

Compared with other P2P-NVEs, VON is able to handle user crowding and keep consistent topology efficiently. Detailed comparisons are given in Table 3. A problem particular to VON is that nodes that are circularly surrounded by others have many enclosing neighbors [7]. However, this is unlikely to occur during natural movements. AOI scope in VON may also be more limiting than in designs that allow message relays, because transmissions are required with *all* AOI neighbors. Another problem arises when a node moves faster than what boundary neighbors can notify as its new neighbors. A "speed limit" may thus be required for proper neighbor discovery [30].

Table 3: Comparisons of server-based and various P2P-based NVE systems

Approach		Neighbor Discovery (Topology Consistency)		Responsiveness*	Scalability †	Pro/Con Notes
Category	**Examples**	**Mechanism**	**Completeness**			
Server-cluster	**MMOGs [4, 12, 23]**	Centralized servers	Complete	Two hops	Server-side bandwidth use is *O(n)*. Faces issues of provisioning, user crowding.	Highly secured and matured but complex and costly.
Enhanced Point-to-point	**UFR [26]**	Centralized servers	Complete	One hop	Peer bandwidth use is *O(n)*.	Does not impose AOI limit on view but does not scale well.
	Frontier Sets [27]					
DHT-based	**SimMud [9]**	DHT and super-nodes	Complete	Two to many	Bandwidth use is constant if density is fixed but *O(m)* for super-nodes where *m* is user size in a given region.	DHT stably maintains consistency, but latency may be high
	Zone Fed [28]			Two hops		
Neighbor-list Exchange	**P2P-MES [24]**	Self-examination via exchanged neighbor-list	Overlay partition	One hop	Bandwidth use is constant	Bandwidth use is bounded, but list exchange is inefficient.
	MIP [29]		Complete		Bandwidth use is constant if density is fixed.	
Mutual Notification	**Solipsis [25]**	Notification by all mutual neighbors	Occasionally incomplete	One hop	Bandwidth use is constant if density is fixed. Dynamic AOI keeps bandwidth use constant under crowding.	Elaborated discovery protocol.
	VON [7, 30]	Notification by boundary neighbors	Complete			May connect excessively if nodes line up in a circle.
Client-server Hybrid	**Federated P2P [31]**	Centralized servers	Complete	Two hops	Server-side bandwidth use is *O(n)*. Faces issues of provisioning, user crowding.	Servers may locate close to clients but require provisioning.
P2P Hybrid	**MOPAR [32]**	DHT, super-nodes, neighbor list exchange	Complete	One hop	Bandwidth use is constant if density is fixed but *O(m)* for super-nodes where *m* is user size in a given region.	List exchange rate may impact discovery timeliness.

*** Responsiveness is indicated by the number of end-to-end hops for receiving a regular position update.**

† Scalability is viewed in two aspects: bandwidth consumption at the bottleneck system node, and whether *crowding* (high user density) is handled.

5. Conclusion

5.1 Summary

The growth and popularity of MMOGs suggest that large-scale NVEs may become the next major Internet applications. We survey an emerging field that utilizes P2P techniques to build scalable and affordable NVEs, and identify content discovery in P2P-NVEs as a *neighbor discovery problem*. To achieve true scalability, P2P-NVEs must be able to constrain resource-use at each node. VON is an elegant solution that organizes neighboring nodes with Voronoi diagrams, and realizes neighbor discovery through mutual node collaborations. Dynamic AOI adjustments keep bandwidth use at each node *bounded*. As shown by analysis and simulations, VON is scalable, efficient and simple. An open source implementation of VON is at http://vast.sourceforge.net.

5.2 Future perspectives

We believe that NVE is an important P2P application domain. Many interesting and challenging topics exist in its future research. Scalability and responsiveness are more or less achievable with careful designs. Of most practical concern is the support for *states* and *ordering consistency* in P2P environments. Reliability of the overlay in the face of node failures and fast-moving nodes require further investigations, while recovery from overlay partitions is also an important topic. Among the six NVE criteria, persistency and security are the more challenging issues in a P2P context.

Although VON does not yet fully support all NVE requirements, distributing movement updates (about 70% of bandwidth use in MMOG [4]) is a valid initial use. Low-security features such as voice-chat may also be supported on VON. Other uses include large-scale military or scientific simulations that are spatially-oriented and require frequent synchronizations. Regarding VON's design, 3D VON is the logical next step. Heterogeneity in P2P networks should also be considered by allowing larger AOI for users with more bandwidth. Finally, we consider streaming 3D data on P2P networks an exciting topic, as it will greatly increase NVEs' accessibility [33].

With current hardware trends, we envision future NVEs as widespread and convenient as today's WWW, and represent a major *medium* for both human and human-computer interactions. Millions of NVEs with various natures and scales will allow people to engage in diverse learning, working, and playing. Most are freely entered with simple URLs, while some may provide subscription contents. In time, a truly massive, immersive *3D cyberspace* might just emerge and be sharable by all.

Acknowledgments

This work was supported by Laboratory of Statistical and Computational Physics (LSCP), Academia Sinica, Taiwan (Grant AS-91-TP-A02). In particular, we wish to thank Dr. Chin-Kun Hu for his supports and encouragements. We would like to thank the reviewers and Dr. Chatschik Bisdikian for suggestions that have much improved this article in both content and presentation, Guan-Ming Liao for the reliable Voronoi library, Kuan-Ta Chen for proofreading the manuscript, and both Ming-Chya Wu and Hung-Shiang Chen for helps on the simulation environments. Thanks also go to Prof. Wei-Chuan Lin, Prof. Jehn-Ruey Jiang, Hua-Hsen Bai, Chun-Wen Chen, Joaquin Keller, Simon Gwendal, Jon Watte, Bart Whitebook, Lili Qui, Crosbie Fitch, Daniel Bauer, Sean Rooney and Jin Chen for helpful discussions, comments and feedbacks.

References

[1] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation*, ACM Press, New York, 1999.

[2] J. Smed, T. Kaukoranta, and H. Hakonen, "Aspects of Networking in Multiplayer Computer Games," in *Proc. ADCOG*. Nov. 2001, pp. 74-81.

[3] D. C. Miller and J. A. Thorpe, "SIMNET: The Advent of Simulator Networking," *Proc. IEEE*, vol. 83, no 8, pp. 1114-1123, Aug. 1995

[4] T. Alexander, Editor, *Massively Multiplayer Game Development*, Charles River Media, 2003.

[5] IGDA, "2004 Persistent Worlds White Paper," http://www.igda.org/online/IGDA_PSW_Whitepaper_2004.pdf

[6] I. Stoica *et al.*, "Chord: a scalable peer-to-peer lookup protocol for Internet applications," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 17-32, 2003.

[7] S. Y. Hu and G. M. Liao, "Scalable peer-to-peer networked virtual environment," in *Proc. ACM SIGCOMM Wksp. on NetGames*, Aug. 2004, pp. 129-133.

[8] F. Aurenhammer, "Voronoi diagrams—a survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 345-405, 1991.

[9] B. Knutsson *et al.*, "Peer-to-peer support for massively multiplayer games," in *Proc. INFOCOM*, Mar. 2004, pp. 96-107.

[10] K. T. Chen, P. Huang, and C. L. Lei, "Game Traffic Analysis: An MMORPG Perspective," to appear in *Computer Networks*, vol. 51, no. 3, 2007.

[11] C. Greenhalgh, J. Purbrick, and D. Snowdon, "Inside MASSIVE-3: Flexible support for data consistency and world structuring," in *Proc. CVE*, 2000, pp. 119–127.

[12] T. Y. Hsiao and S. M. Yuan, "Practical Middleware for Massively Multiplayer Online Games," *IEEE Internet Computing*, vol. 9, no. 5, 2005, pp. 47-54.

[13] T. A. Funkhouser, “RING: A client-server system for multi-user virtual environments,” in *Proc. Symp. Interactive 3D Graphics*. Apr. 1995. pp. 85-92.

[14] S. K. Singhal and D. R. Cheriton, “Using projection aggregations to support scalability in distributed simulation,” in *Proc. ICDCS*, May 1996, pp. 196-206.

[15] K.L. Morse, L. Bic, and M. Dillencourt, “Interest management in large-scale virtual environments,” *Presence*, vol. 9, no. 1, pp. 52-68, 2000.

[16] M. R. Macedonia *et al.*, “Exploiting reality with multicast groups,” *IEEE CG&A*, vol. 15, no. 5, pp. 38 - 45, 1995.

[17] E. Frécon, “DIVE: Communication Architecture and Programming Model," *IEEE Comm. Mag.*, vol. 42, no. 4, pp. 34-40, 2004.

[18] J. W. Barrus, R. C. Waters and David B. Anderson, “Locales: Supporting Large Multiuser Virtual Environments,” *IEEE CG&A*, vol. 16, no. 6, pp. 50-57, 1996.

[19] C. Diot and L. Gautier, “A distributed architecture for multiplayer interactive applications on the Internet,” *IEEE Network*, vol. 13, no. 4, pp. 6-15, 1999.

[20] J. C. Oliveira and N. D. Georganas, “VELVET: An Adaptive Hybrid Architecture for VEry Large Virtual EnvironmenTs,” *Presence*, vol. 12, no. 6, pp. 555-580, 2003.

[21] E. Lety, T. Turletti, and F. Baccelli, “SCORE: A Scalable Communication Protocol for Large-Scale Virtual Environments,” *IEEE/ACM Trans. Networking*, vol. 12, no. 2, pp. 247-260, 2004.

[22] J. Liebeherr, M. Nahas and W. Si, "Application-layer Multicasting with Delaunay Triangulation Overlays," *IEEE JSAC*, vol. 20, no. 8, pp. 1472-1488, 2002.

[23] J. Chen *et al.*, “Locality aware dynamic load management for massively multiplayer games,” in *Proc. ACM SIGPLAN PPoPP*, Jun. 2005, pp. 289-300.

[24] Y. Kawahara, T. Aoyama, and H. Morikawa, “A peer-to-peer message exchange scheme for large-scale networked virtual environments,” *Telecomm. Sys.*, vol. 25, no. 3-4, pp. 353–370, 2004.

[25] J. Keller and G. Simon, “Towards a peer-to-peer shared virtual reality,” in *Proc. 22nd Int. Conf. Distributed Computing Systems (Workshops)*, July 2002. pp. 695-700. http://solipsis.netofpeers.net/

[26] A. Goldin and C. Gotsman, "Geometric Message-Filtering Protocols for Distributed Multiagent Environments," *Presence*, vol. 13, no. 3, pp. 279 - 295, 2004.

[27] A. Steed and C. Angus, “Supporting Scalable Peer to Peer Virtual Environments Using Frontier Sets,” in *Proc. IEEE Virtual Reality*, Mar. 2005, pp. 27-34.

[28] T. Iimura, H. Hazeyama, and Y. Kadobayashi, "Zoned federation of game servers: a peer-to-peer approach to scalable multi-player online games," in *Proc. ACM SIGCOMM Wksp. on NetGames*, Aug. 2004, pp. 116-120.

[29] J. F. Chen *et al.*, "A Message Interchange Protocol Based on Routing Information Protocol in a Virtual World," *in Proc. AINA'05*, Mar. 2005, pp. 377-384.

[30] S. Y. Hu, "Scalable peer-to-peer networked virtual environment," Master's thesis, Tamkang Univ., Taiwan, Jan. 2005. http://vast.sourceforge.net/docs/pub/2005_hu_master_thesis.pdf

[31] S. Rooney, D. Bauer, and R. Deydier, "A federated peer-to-peer network game architecture," *IEEE Commun. Mag.*, vol. 42, no. 5, pp. 114 – 122, 2004.

[32] A. Yu and S. T. Vuong, "MOPAR: a mobile peer-to-peer overlay architecture for interest management of massively multiplayer online games," in *Proc. NOSSDAV*, Jun. 2005, pp. 99-104.

[33] S. Y. Hu, "A case for 3D streaming on peer-to-peer networks," in *Proc. Int. Conf. 3D Web Technology (Web3D 2006)*, Apr. 2006, pp. 57-63. http://ascend.sourceforge.net/docs/pub/2006-hu-3DstreamingP2P.pdf