Voronoi-based Overlay Network

Available at: http://vast.sourceforge.net/VON

1. Executive Summary

Massively Multiplayer Online Games (MMOG), where tens of thousands of players participate in dynamic and socially engaging adventures, have risen quickly in recent years as a major form of entertainment, particularly in North America and Asia. *Lineage,* the current number one MMOG and its sequel has a combined subscription of 4 million people worldwide, with peak-time simultaneous players exceeding 140,000 people. *World of Warcraft*, a new MMOG released just in mid-2004, quickly achieved 2 million subscribers in just a few months [1]. Since its inception in the late 90s, MMOG has been fast becoming a multi-billion dollar industry [2].

As we extrapolate current advances in CPU, bandwidth and consumer-grade 3D accelerators, it is not unreasonable to expect MMOG of larger scale and greater diversity to emerge and integrate to our computing and even living experience in the near future, so that we may work, learn, and play in highly interactive, immersive 3D *networked virtual environments* (or NVEs, the more general term for MMOG).

To reach a truly massive audience and evolve NVE into a global phenomenon, its scale must go up while the cost must go down. Therefore, two important issues are *scalability* and *affordability*. Basically, how do we create a NVE that allows *millions of users to participate at the same time*? And, how can we lower the costs in development and deployment so that *NVE may become as commonly and easily accessible as today's WWW?* This work attempts to find an initial answer to these questions. In other words, how do we design the next generation, highly scalable and affordable multi-user networked virtual environments?

Existing client-server based architecture has certain inherent difficulties and limitations. Server-side bandwidth and processing power needs to be increased accordingly if more users were to be supported. To support orders of magnitude more users than existing size (of about 10,000 to 150,000 users), not only server design and maintenance complexity will increase, building such NVEs will also become prohibitively expensive to medium and small developers.

We seek to address these problems by using a relatively recent computing paradigm called *peer-to-peer* (P2P) computing, in hope that highly scalable and affordable systems may become feasible. P2P is characterized by utilizing end-users' system resource collaboratively, so that useful resource can grow as the number of participants increases. The scalability and affordability of P2P has been publicized and demonstrated by many file-sharing applications such as Napster, Gnutella, Kazaa, and eDonkey, as well as the more recent VoIP application Skype. However, so far P2P has not yet been successfully applied to NVE.

We propose a new class of P2P network called *Voronoi-based Overlay Network* (VON) to realize highly scalable NVE. Although issues still remain to support the full functional requirements of NVE, we have devised a scalable, efficient, and easily-extensible foundation on which NVE applications may be built. The scalability potential of the design has been demonstrated by simulations. Our work lays the foundation for many exciting and promising research directions that may realize NVE development and deployment on a massive scale, for fields as diverse as entertainment, military and science.

This work was originally described in the 2004 ACM SIGCOMM workshop Netgame2004 [3], with a summarized version recently accepted by IEEE Network [4]. An open source implementation has also been released and hosted at SourceForge [5].

2. Problem Definition

2.1 The scalability problem

There are a number of criteria that need to be addressed when designing an NVE, including: consistency, responsiveness, security, scalability, persistency, and reliability [6]. This work focuses on the scalability problem, which is defined in terms of the number of concurrent users in a NVE system.

The general NVE communication problem is defined as: each participant assumes a representation (called *avatar*) and uses a computer terminal (PC or workstation) to access the NVE. For our purpose, an avatar is a *node* on the network and is represented as a point on a 2D coordinate plane (see Figure 1). The visibility of a node is called its *Area of Interest* (AOI), and is represented as a circle centered on the node. Although many nodes can exist in the whole system, a single node is only aware of its *AOI neighbors* at any given time. As each participant moves or makes an action (such as chatting, trading, or fighting), a message is generated and delivered to all other nodes that may see the action. The basic problem then is: how can each node receive the proper/relevant messages from other nodes within its AOI? And the scalability problem would be *how can these nodes communicate on a massive scale*?

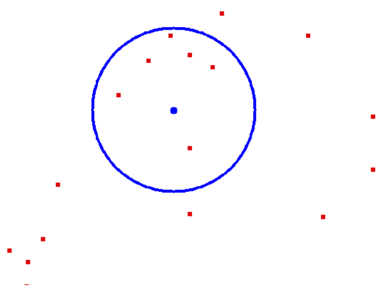
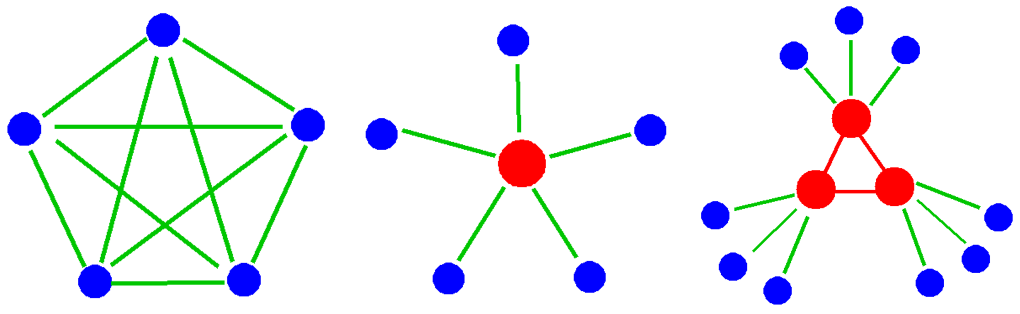


Figure 1: Concept of Area of Interest (AOI)

2.2 Previous approaches

The most straightforward solution to this communication problem is to let each node broadcast update messages to all other nodes (see Figure 2a). This method is simple and works well when the system is small. However, the amount of messages grows exponentially at a rate of $O(n^2)$. Therefore, it is clearly not scalable.



(a) **(b)** **(c)**

Figure 2: Evolution of scalability solutions

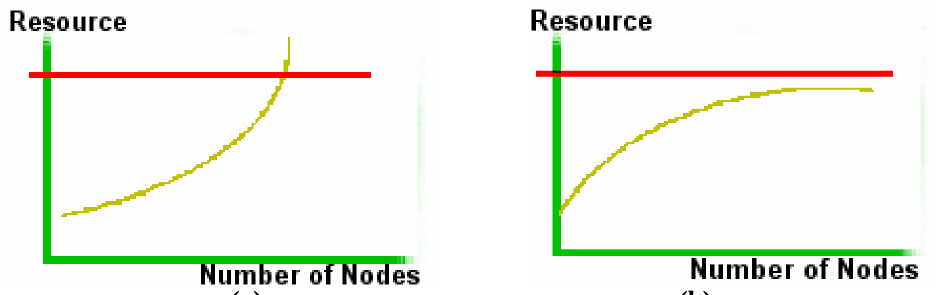
(a) point-to-point (b) client-server (single server) (c) server-cluster (multiple servers)

A more scalable approach is to let a server be responsible to relay all message traffics (see Figure 2b). The server can act as a filter, preventing a node from receiving irrelevant (i.e. beyond AOI) messages. This greatly reduces the number of connections from $O(n^2)$ to $O(n)$ and message traffics. However, a certain amount of server-side processing and bandwidth resource is consumed with each additional user. Scalability therefore is limited by the amount of resource available on a single server.

To increase server-side resource, single server design has evolved into multiple servers, forming a *server-cluster* (see Figure 2c). Server-cluster can support even larger number of users, and is the state-of-the-art for current MMOGs. However, balancing user load across servers can be a delicate process, which introduces new problems such as *player handoff* (or *avatar migration*, which is when a user needs to move from one server to another without noticeable delay). Often, server responsibility is determined geographically by splitting the virtual world into several *regions*, if too many users *crowd* or *flock* to the same region (as happened when an interesting event takes place), server overload may still occur.

While server-cluster offers a viable scalability solution, it requires costly funding and increases the amount of design and maintenance complexity. Issues of load balancing, fault-tolerance, and maintenance can all become problematic as the number of users scales. Usually, manual efforts and over-provisioning of server resource are also required (as peak usage can come unexpectedly). For a given cluster, an inherent upper limit in the number of participants still exists (assuming users are spread out evenly across servers, if not, that limit is reached even sooner). Is there any alternative to support larger number of users without the problems faced by server-cluster?

One useful way to look at the scalability problem is from a resource-constrain perspective. Independent of the underlying architecture, if we assume that every system is made up of a number of components, the system remains scalable as long as resource is available to accommodate new components. Every system also has a *limiting component* (for example, the server in client-server architecture), if resource consumption pattern at this limiting component is like Figure 3a, at some point the system will cease to be scalable due to resource depletion. On the other hand, if a system can keep consumption within the limit of available resource as Figure 3b, then the system could be highly scalable.



(a) **(b)**

Figure 3: Scalability analysis

X-axis denotes the number of nodes; y-axis denotes resource consumption at the limiting component; top horizontal line indicates the resource limit.
(a) Non-scalable system. (b) Scalable system.

2.3 NVE in P2P

A new class of peer-to-peer (P2P) applications have emerged in recent years and caught public attention. The most publicized is perhaps file-sharing P2P such as Napster, Gnutella, e-Donkey, among others [7]. P2P has been defined as “distributed systems without any centralized control or hierarchical organization, in which each node runs software with equivalent functionality” [8]. Although P2P can be considered as a form of *distributed computing*, it differs from *grid computing* (another well-known type of distributed computing) in the sense that the constitute units are commodity PCs which may join or fail the network anytime, while grid computing usually consists of well-provisioned facilities overseen by administrators [9].

P2P offers two attractive promises: as each participating machine contributes its own resource, system size can be highly *scalable* as total resource grows with system size; it is also a form of very *affordable* computing resource as the users themselves donate the necessary resources without requiring provisioning. Given these qualities, it may be desirable to apply P2P to improve NVE scalability, but how?

A P2P network forms when each participating node connects to some other nodes according to certain rules. For example, in file-sharing P2P, a node connects to other nodes that host files which the user wants. However, finding the right nodes to connect is challenging as a target file can potentially be located on *any* node in the vast P2P network. Luckily for NVE, each node generally is interested only in the messages generated by its AOI neighbors (i.e. those nodes within its AOI). In other words, due to this *locality of interest*, connecting only to AOI neighbors is sufficient for the system to function properly. Ideally, each node would connect *directly* to its AOI neighbors and only exchange messages with them, so that latency is minimized (which is important to keep the real-time requirement of NVE) and bandwidth is only used to transfer relevant information. However, because neighbor relationships may change as nodes move around, finding the proper AOI neighbors becomes the central problem (i.e. a *neighbor discovery problem*) for P2P-based NVE.

Some solutions that utilize P2P for NVE have been proposed in the past two years, each of them has certain unaddressed areas. For brevity they will not be described here, but readers are encouraged to refer to [6].

3. Approach

Our approach is to use *Voronoi diagram* to solve the neighbor discovery problem. We first describe what Voronoi is, and then explain the designs of the P2P network.

3.1 Voronoi explained

Given *n* points (referred to as *sites*) on a 2D plane, we can partition the plane into *n* non-overlapping *regions* such that all points within a region are closer to the region's site than to any other site (see Figure 4a). The region can be seen as the *sphere of influence* of a particular site. Voronoi diagram has certain nice properties: for example, once constructed, it can be used to easily identify the *k-nearest neighbors* for any given site. Voronoi diagrams have been studied extensively for nearly a century and applied in diverse fields. Voronoi diagram construction is beyond the scope of this work and readers may refer to [10] for existing algorithms.

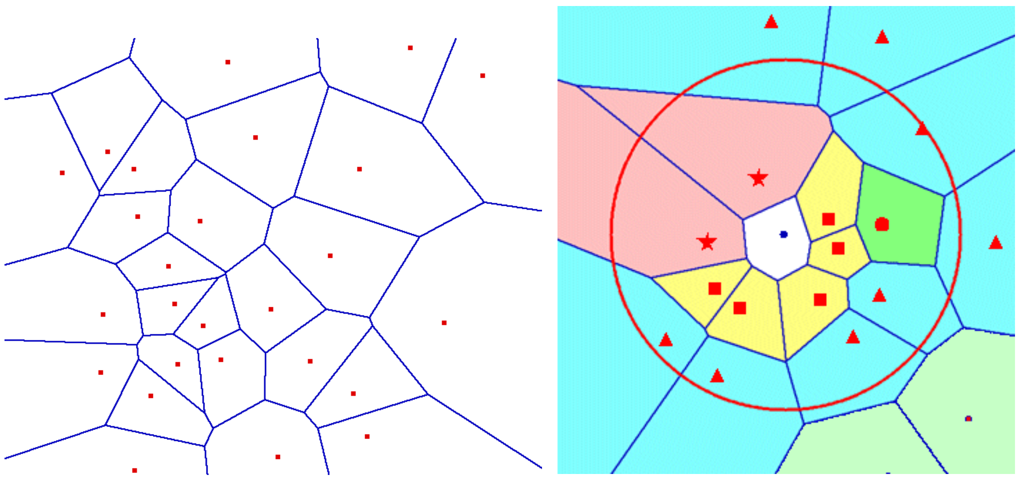


Figure 4: Voronoi diagram

(a) The dots indicate *sites*, and lines define boundaries for *regions*. (b) Squares (▩) represent *enclosing neighbors*; triangles (▲) represent *boundary neighbors*; stars (★) are *both* enclosing and boundary neighbors; circle (●) represents a regular *AOI-neighbor*.

3.2 Voronoi-based P2P

As stated earlier, the ideal P2P design for NVE is to let each node connect directly and exchange messages with only its AOI neighbors. The key problem, however, is to discover new neighbors to connect to, as nodes move around.

In our design, we require each node to maintain direct connections with all of its *AOI neighbors* (i.e. nodes within the AOI of a given node). Given the coordinates of its neighbors, a node may then construct a Voronoi diagram containing all of its AOI neighbors. Discovery of new neighbors is done with the help from *boundary* *neighbors* (defined as AOI neighbors whose Voronoi regions overlap with the AOI boundary) as they may know what other nodes exist beyond the AOI (see Figure 4b). As each node moves, it sends updates to all of its AOI neighbors. If a boundary neighbor receives the update, it would check for potential new neighbors on behave of the moving node and send out notifications if new AOI neighbors are found. To ensure that the P2P topology remains fully-connected even when a node has no AOI neighbors, we also require each node to minimally maintain its *enclosing neighbors* (defined as the nodes whose Voronoi regions immediately surround the given node, see Figure 4b).

We call this scheme *Voronoi-based Overlay Network* (or VON for short, *overlay* is the academic term for P2P network, as it is built on top of the physical Internet). Compared with existing architectures or other P2P-based NVE designs, VON may achieve better *scalability* (as each node only maintains a limited number of nodes, resource consumption is thus bounded), *responsiveness* (because latency is minimized by the direct connections between nodes), and *message-efficiency* (requests to check for potential new neighbors are embedded in regular position updates, and neighbor notification is sent only when necessary, see [6] for a complete comparison).

3.3 Dynamic AOI adjustments

To prevent a node from message overload when *crowding* occurs (i.e. when many nodes come nearby to each other, see Figure 5), we also propose dynamic adjustments to AOI boundary when necessary. The rules are simple: 1) AOI-radius is shrunk when the number of connections exceeds a predefined *connection limit.* 2) AOI-radius grows back when number of connections has decreased. 3) Mutual awareness should be maintained. So if node A cannot see node B due to overload, node B should also shrink its AOI-radius to exclude node A.

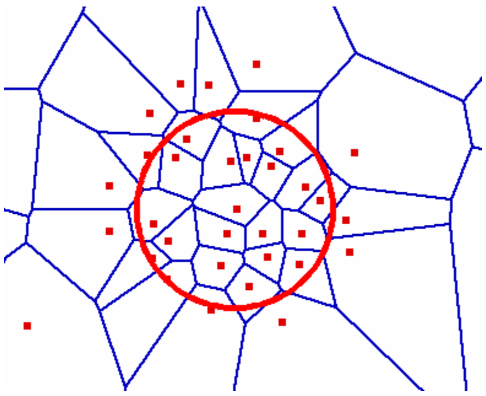


Figure 5: Example of a crowding situation

4. Results

To evaluate VON, we run a number of simulations to test the scalability, topology consistency, and reliability of the design. Simulation parameters are shown in Table 1 and two metrics are used during the simulations:

Topology consistency [11] - defined as the ratio between the numbers of AOI neighbors that should be seen versus those actually observed. For example, if 5 neighbors exist within a node's AOI, but only 4 are known, then topology consistency would be 4/5 = 80%).

Drift distance [12] – Topology consistency does not capture the inconsistencies between actual and observed coordinates of a remote node. The absolute difference in coordinate values for each known node is called *drift distance*.

Table 1 Simulation parameters

	Simulation Type	
Parameters	**Scalability**	**Reliability**
World dimension (units)	1000x1000	1000x1000
Initial AOI-radius (units)	150	150
Packet loss rate*	0%	**0% ~ 100%**
Simulation time-steps	1000	1000
Velocity (units/time-step)	5	5
Maximum connection (in dynamic AOI)	10	10
Number of nodes (in increment of 20)	**10 ~ 250**	150

* Packet loss rate indicates the loss in all messages regarding position updates and neighbor discovery, which makes up 95% of all transmission. Other control messages (5%) are still delivered reliably.

4.1 Scalability

A system remains scalable as long as each system component does not deplete its resource with the addition of new nodes. In the case of P2P-based NVE, as long as bandwidth use at each node is kept within the node's limit, the system may scale continuously. Therefore, to qualitatively evaluate a NVE system's scalability, a simple way is to observe its bandwidth use at each node as a function of system scale.

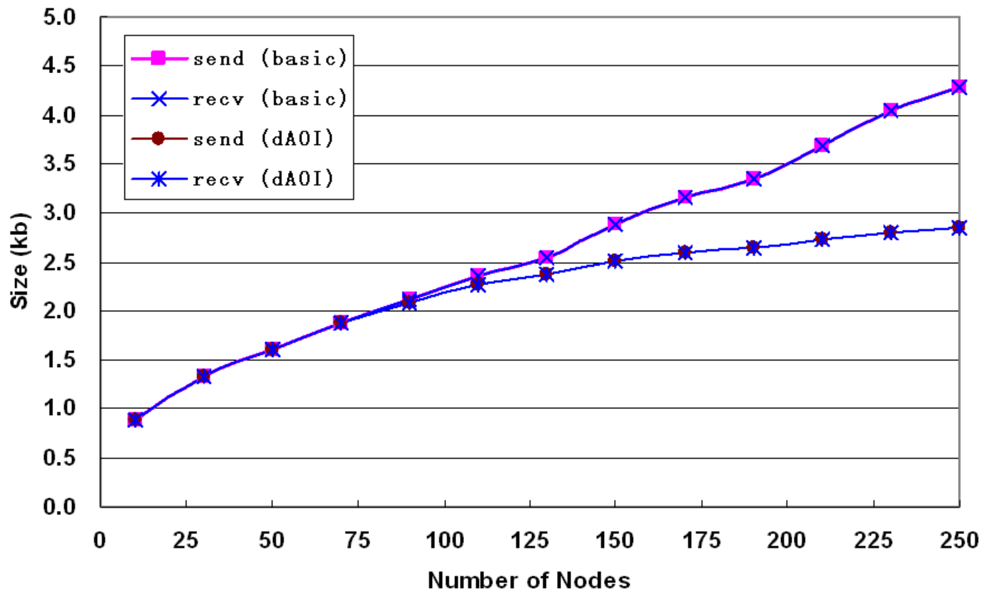


Figure 6: Average transmission size per node per second

Figure 6 shows that as the number of nodes increases, the average transmission size for each node increases *linearly* in the basic mode (i.e. fixed AOI radius) and *logarithmically* in the dynamic AOI model (i.e. adjustable AOI-radius). This shows that resource consumption of VON is at least as good as existing client-server architecture's O(n) for the basic model, and may achieve potentially much better scalability with the sub-linear consumption growth under dynamic AOI adjustments.

In the dynamic AOI model, average transmission size per second for each node approaches 3kb/second at 250 nodes (assuming 10 updates are generated per second), this is well within the bandwidth provided by current broadband (which operates at about 16 ~ 32kb / second for upload and 64 ~ 128kb / second for download). However, the significance here is not the actual transmission size, which depends heavily on the implementation, but that with careful design, transmission may become *bounded*, which is the characteristic for highly scalable systems.

This bounded transmission is explained by Figure 7, which shows the average number of connected neighbors and AOI neighbors for each node. It is clear that the number of neighbors becomes bounded due to the pre-defined connection limit.

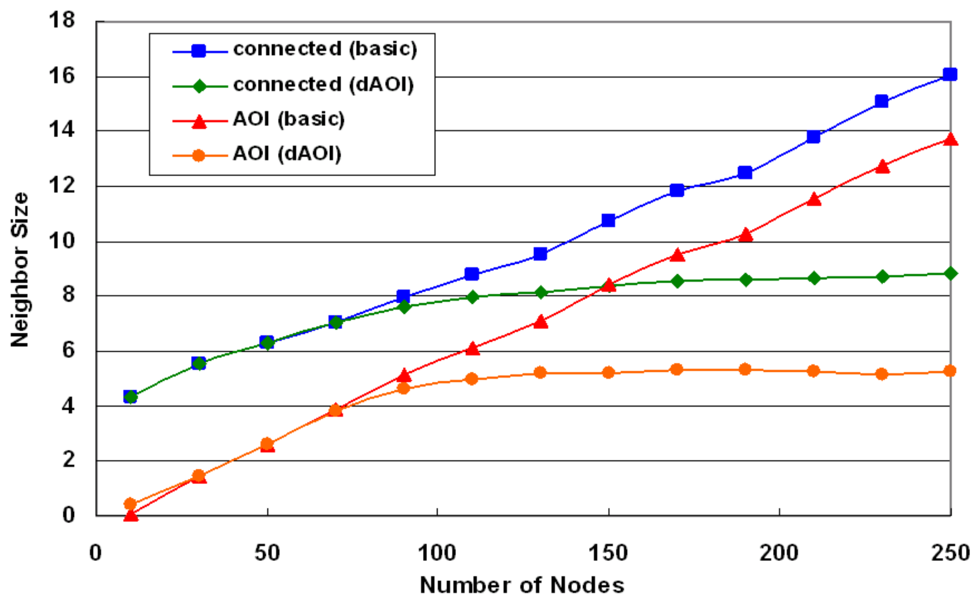


Figure 7: Average neighbor size for basic and dynamic AOI models

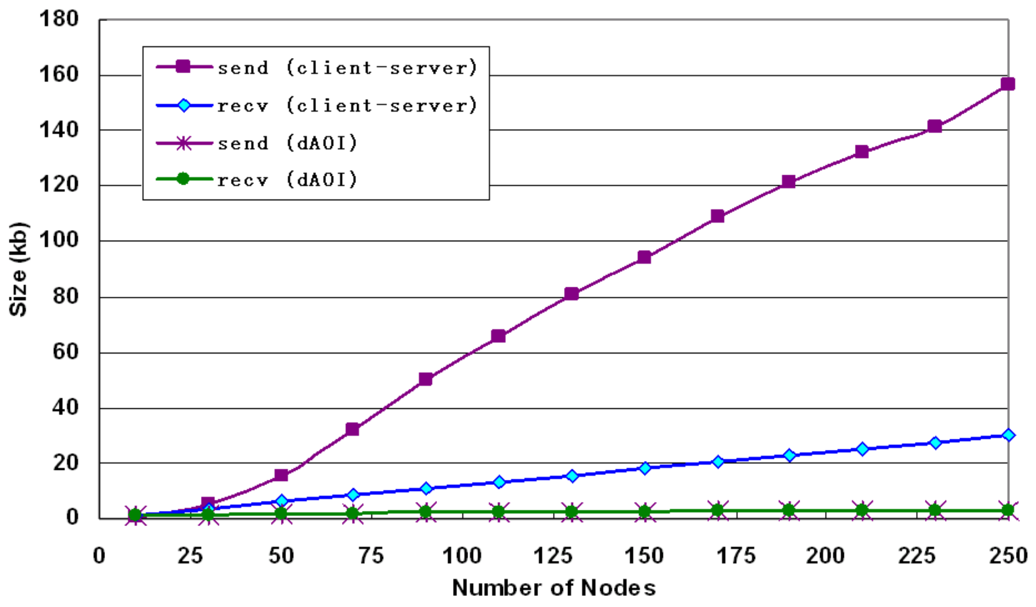


Figure 8: Comparison of transmission size between VON and client-server

If we compare VON against client-server architecture with equivalent function, we see that a single node in VON utilizes much less bandwidth than a typical server, by distributing message transmission to all participating nodes (see Figure 8). This not only lowers system cost by removing the need for a powerful, high-bandwidth server, but also increases system robustness as no single point of failure exists.

4.2 Topology consistency

Scalability would not be useful if consistency is greatly sacrificed. We evaluate VON's topology consistency by observing *topology consistency* and *drift distance* matrices. Figure 9 shows that the topology consistency is close to 100% for both basic and dynamic AOI models, though there is a slight drop in dynamic AOI. However, consistency still remains high (above 99.70%) for up to 250 nodes. Drift distance is likewise quite low, and remains close to 0 (see Figure 10). The slight inconsistency is likely caused by a temporary asymmetric understanding of the topology after AOI adjustments. However, inconsistency is bound to occur in a real network environment due to latency, the more important question therefore is how quickly can it recover from inconsistency?

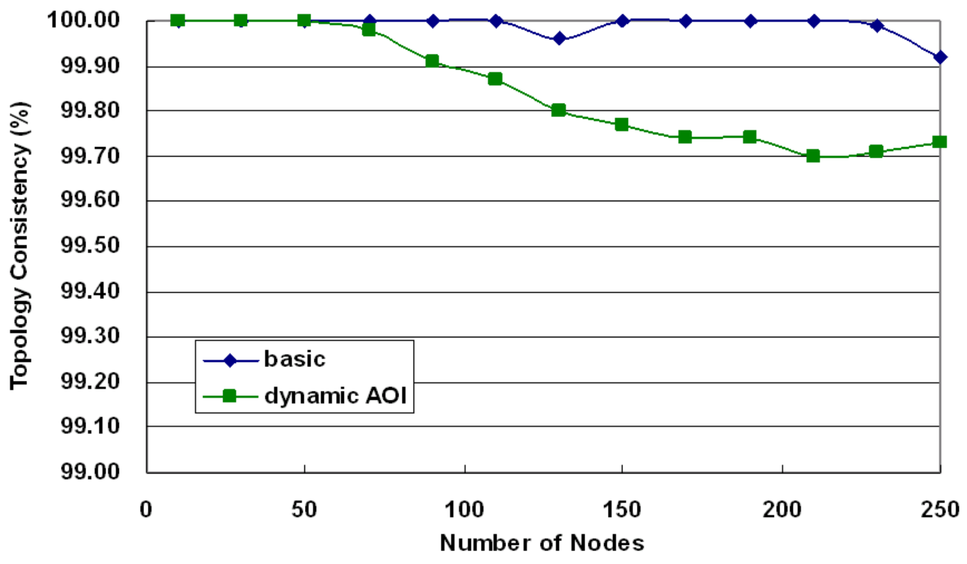


Figure 9: Topology consistency for basic and dynamic AOI models

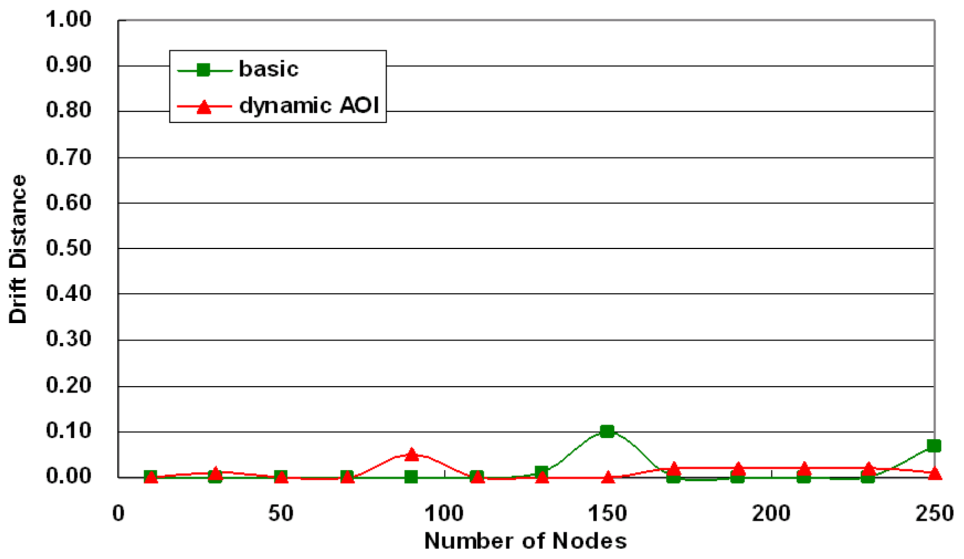


Figure 10: Average drift distance for basic and dynamic AOI models

This question is answered by Figure 11, which shows the number of steps to recover from inconsistency under dynamic AOI. Here VON recovers within an average of 1.5 steps, which is fairly fast. We can thus see that VON keeps fairly good consistency in a robust manner.

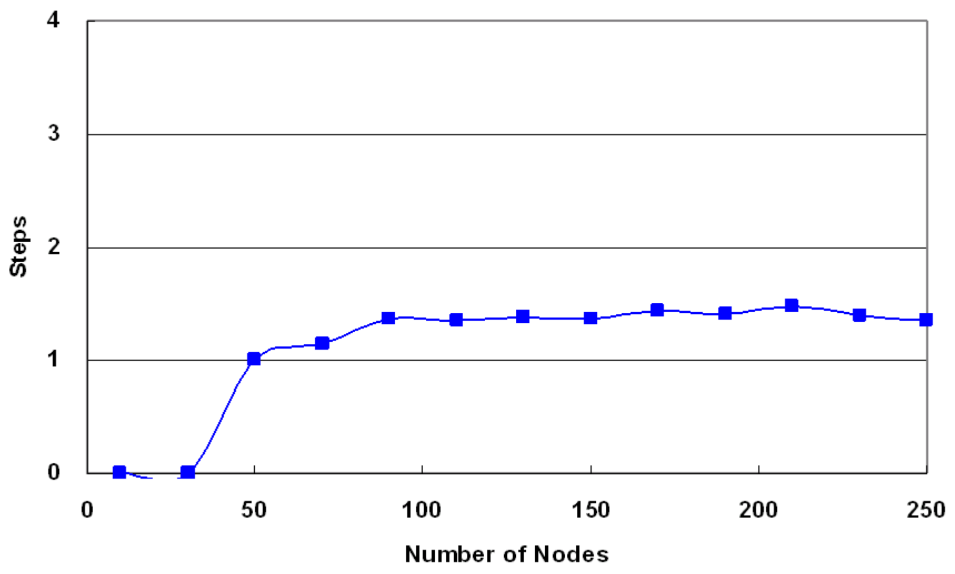


Figure 11: Average number of steps to recover from inconsistency

4.3 Reliability

How good does VON perform when the underlying network is not 100% reliable? We answer this question by performing another set of simulations where packet loss ranges from 0% to 100%. As certain messages are essential to maintain the topology (such as initial connection handshakes), we keep them to be delivered reliably, but allow other types of messages to be sent unreliably (mostly movement notifications and neighbor discovery messages, which occupy about 95% of all messages).

In Figure 12 we see that VON still maintains fair consistency for a loss rate up to 50%, and only begin to drop after a loss rate of 60%. Figure 13 shows the corresponding average recovery steps, where VON can still recovers from inconsistency within 4 steps for a loss rate up to 50%.

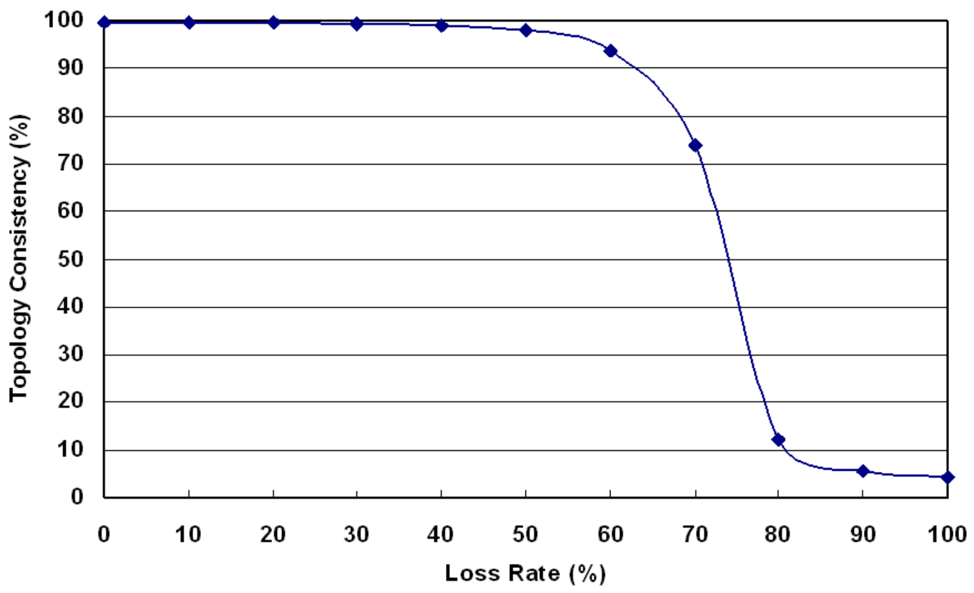


Figure 12: Effect of loss rate on topology consistency (dynamic AOI model)

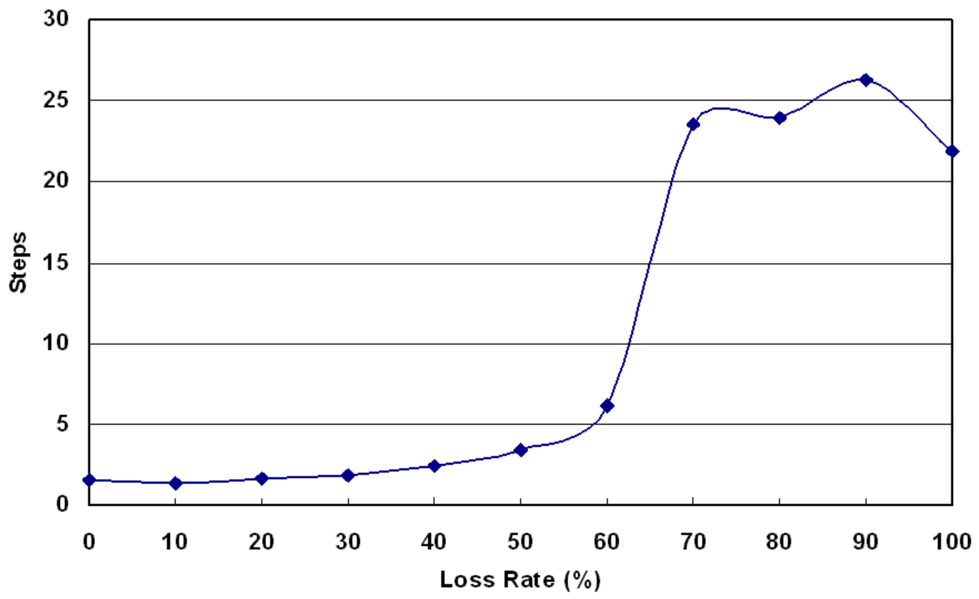


Figure 13: Effect of loss rate on average recovery steps

5. Potential Applications

5.1 MMOG

Massively Multiplayer Online Game is the initial application area in which VON was designed for. By adopting VON, highly scalable, responsive, and fault-tolerant MMOG may be developed and deployed affordably. Although VON in its current form does not yet support the full functional requirements of a MMOG, such as event consistency, data persistency, or security, VON can be used to offload position updates from the servers (estimated to be 70% of all message traffics), which can greatly reduce the cost for server-side bandwidth.

5.2 Military simulations

Large-scale military simulations where human trainees go to immersive simulated battlefields were among the first NVE applications. Security concerns regarding P2P (i.e. that the clients could be hacked) may postpone the full adoption by game industry. However, in a military context, as all the simulators are within military's control, concerns for client-side hacking is non-existent. It may be quite practical, even in its present form, to adopt VON in large-scale military applications.

5.3 Scientific simulations

Viewed from a more general perspective, VON is in fact not just a platform for games, but a P2P-based environment that supports spatially-oriented (i.e. 2D or 3D coordinate system) and tightly synchronized (i.e. state information is exchanged at every time-step) large-scale simulations. As such, another promising application would be in scientific simulations where coordinate systems are the basis of the simulation. For example, molecular dynamics (MD) that simulates physical systems at atomic scale may be distributed on VON to parallelize the calculations.

References

[1] MMOG Chart. Available at: http://www.mmogchart.com/

[2] IGDA, "2004 Persistent Worlds Whitepaper," Available at: http://www.igda.org/online/IGDA_PSW_Whitepaper_2004.pdf

[3] Shun-Yun Hu and Guan-Ming Liao, "Scalable Peer-to-Peer Networked Virtual Environment," in *Proc. ACM SIGCOMM 2004 workshops on NetGames '04,* Aug. 2004, pp. 129-133.

[4] Shun-Yun Hu, Jui-Fa Chen and Tsu-Han Chen, "VON: A scalable peer-to-peer network for virtual environments," *IEEE Network* (accepted), 2006. http://vast.sourceforge.net/docs/pub/2006-hu-VON.pdf

[5] VAST Project Home Page. Available at: http://vast.sourceforge.net

[6] Shun-Yun Hu, "Scalable Peer-to-peer Networked Virtual Environment," Master's thesis, Tamkang Univ., Taiwan, 2005. Available at: http://vast.sourceforge.net/docs/pub/2005_hu_master_thesis.pdf

[7] Wikipedia, "definition of peer-to-peer." Available at: http://en.wikipedia.org/wiki/Peer-to-peer

[8] 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.

[9] I. Foster and A. Iamnitchi, "On death, taxes, and the convergence of peer-topeer and grid computing," in 2nd Intl. Workshop on Peer-to-Peer Systems (IPTPS'03), Berkeley, CA (2003)**.**

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

[11] Y. Kawahara, T. Aoyama, and H. Morikawa, "A peer-to-peer message exchange scheme for large-scale networked virtual environments," *Telecommunication Systems*, vol. 25, no. 3-4, pp. 353–370, 2004.

[12] 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.