A Forwarding Model for Voronoi-based Overlay Network

Tsu-Han Chen [1]
Dept. of Computer Science and Information Engineering, Tamkang University

Jui-Fa Chen[2]
Dept. of Computer Science and Information Engineering, Tamkang University

Shun-Yun Hu[3]
Institute of Physics, Academia Sinica

ABSTRACT

One recent approach to build highly scalable and robust networked virtual environments (NVEs) is by using peer-to-peer overlay networks. Voronoi-based Overlay Network (VON) has been proposed that promises to maintain high overlay topology consistency in a bandwidth-efficient manner. However, the initial proposal requires all nodes to connect directly with their relevant neighbors (the direction-connection model), this limits the number of neighbors that may appear within the area of interest (AOI) of a given node. A new forwarding model for VON is proposed in this paper to solve this problem by requiring connection with only the nearest neighbors (called enclosing neighbors), and propagate position updates to other interested nodes by message forwarding. This way, AOI may be more flexibly expanded and different bandwidth capacities may be more efficiently utilized at the expense of increased latency.

CR Categories and Subject Descriptors: C.2.1 [COMPUTER-COMMUNICATION NETWORKS]: Network Architecture and Design - Distributed networks

Keywords: Peer-to-peer (P2P), overlay network, networked virtual environment (NVE), Voronoi, Voronoi-based Overlay Network (VON)

1 INTRODUCTION

Massively multiplayer online game (MMOG) is a class of networked virtual environments (NVEs) that is experiencing rapid growth in recent years. However, scalability of such systems face challenges in server design complexity, hardware-provisioning, load-balancing and fault-tolerance issues. Efforts have recently been put into utilizing peer-to-peer (P2P) networks to support such large-scale NVE systems [1] [2] [3] [4]. Among these proposals, Voronoi-based Overlay Network (VON) promises to provide scalability and overlay topology consistency in a bandwidth-efficient manner [4] [5]. However, the current proposal requires nodes to directly connect to all neighboring nodes of interest (called Area-of-Interest, or AOI neighbors). With limited per-node bandwidth, such design may limit the number of visible nodes seen by any given node. In this paper, we propose the forwarding model, an extension to the original direct-connection model of VON, by allowing messages relayed by neighbor nodes, and restrict direct connections with only the nearest neighbors immediately surrounding each node. In a realistic network environment, every node has different network bandwidth, thus it is also preferable that each node can control its own transmission size. By using message forwarding and data compression, each node may potentially see a larger number of AOI neighbors and more efficiently utilize its bandwidth resource, at the expense of increased network latency.

2 RELATED WORK

2.1 Peer-to-peer network

Peer-to-peer network is a non client-sever network overlay. Data is transferred without centralized severs in a distributed fashion. In other words, every machine is not only a client, but also a server. An important issue is to whom each node should connect to and communicates with in a P2P network.

2.2 Terminology in VON

Voronoi Diagram is a mathematical construct that, given *n* nodes on a plane, can partition the plane into *n* non-overlapping Voronoi *regions* [6]. We can define *enclosing neighbors* (EN) as those nodes whose regions immediately surround a particular node. In Figure 1, if node S is the node in discussion, then all the pink nodes are its enclosing neighbor, because their Voronoi regions immediately surround S's Voronoi region. When nodes moved, each node's enclosing neighbors will update frequently. We also define the *AOI* (area of interest) of S as the area indicated by the green circle. Nodes within the AOI are called *AOI neighbors*, which are the nodes S is interested in. In Figure 1, the pink nodes and green nodes are both AOI neighbors of S. *Boundary neighbors* (BN) are node S's AOI neighbors whose enclosing neighbors are only partially inside S's AOI.

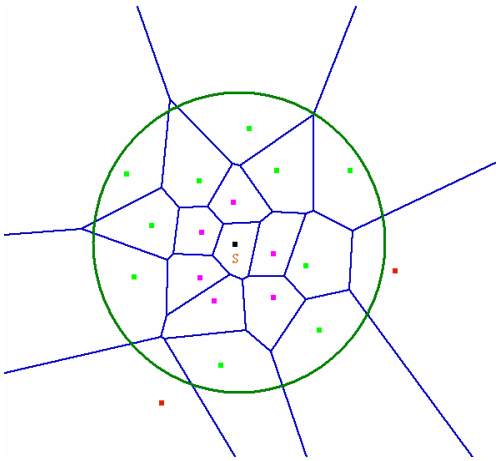

Figure 1. Voronoi Diagram

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

2.3 VON - direct-connection model

In this section, we briefly describe the direct-connection model, and try to describe some of its problems that we attempt to address All nodes make direct connection with their AOI neighbors, and send position updates to AOI neighbors at each time-step. As nodes move, new neighbors are discovered by notification from the boundary neighbors, as they know both the moving node and other potentially visible new AOI neighbors. Details of the direction connection model can be found in [4] and [5]. However, as nodes need to connect to all of their AOI neighbors directly, if there are too many AOI neighbors the transmission size will be too big to transfer in time. Dynamical adjustment to AOI radius is therefore proposed [5] to limit the transmission size by imposing a *connection limit*. However, shrinking AOI-radius will reduce the visible area for a given node. Direction connection model also requires all nodes to be mutually visible, which decreases visibility unnecessarily for nodes that have larger bandwidth capacity.

3 FORWARDING MODEL ALGORITHM

In this section, we will describe the algorithm for the forwarding model. The main idea is that if each node only directly connects to its enclosing neighbors, who then forward relevant messages to other nodes, then we may achieve larger AOI visibility with less number of active connections. The major goal we want to achieve is to make sure that messages can be forwarded by enclosing neighbors, and we will describe three major procedures. For the forwarding model to work, it is important to make sure that all nodes know their enclosing neighbors correctly, so the first issue is how to join this VON and get the enclosing neighbor list (JOIN procedure). Afterwards, we will need to update the enclosing neighbor list and get other AOI neighbors' state messages when nodes are moving around (MOVE procedure). Finally, how to exit the P2P overlay is also described (LEAVE procedure).

3.1 Join procedure

3.1.1 Forwarding the "JOIN" message

A node joins the VON (called joiner) by sending a "JOIN" message, which contains initial position, AOI radius and its IP/port information to a *gateway*. This gateway could be the creator of this overlay or some existing nodes in the VON. The message is forwarded until reaching an *acceptor*, which is the node whose Voronoi region contains the joiner's initial position.

3.1.2 Obtaining initial neighbor list

If the joiner is acceptor's enclosing neighbor, acceptor will send what it considers as the joiner's enclosing neighbor list to the joiner (note that the list may not be complete), and the joiner will send a "HELLO" message to its initial enclosing neighbors to make sure that its enclosing neighbor list is correct. The "HELLO" message contains the joiner's states (position, AOI radius and IP address) and a list of the enclosing neighbors it knows. When the new neighbors find that the list is incomplete, they will send the missing enclosing neighbors' states to the joiner. The same procedure will repeat with each new neighbor discovered, until no more missing enclosing neighbors are found.

3.2 Move procedure

3.2.1 Maintaining enclosing neighbors

When nodes move, their neighbors will change, so we need to record the enclosing neighbors' *overlap neighbors* to facilities the discovery of enclosing neighbors The overlap neighbor of a moving node S's enclosing neighbor A is the neighbor that is a shared enclosing neighbor for both S and A. When the overlap neighbors change, moving nodes are notified about these changes, then the moving nodes will determine whether these newly discovered neighbors are really new enclosing neighbors or not.

In Figure 2, left picture shows time-step 1 and right picture shows time-step 2. At time-step 1, the overlap neighbors of S and A are B and C. At time-step 2, D has moved, so the overlap neighbors of S and A change to B and D. A thus will notify S about D.

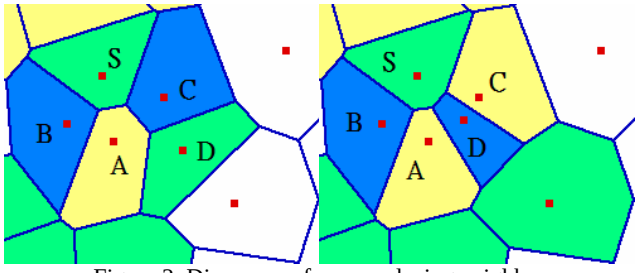

Figure 2. Discovery of new enclosing neighbor

3.2.2 Maintaining AOI Neighbors

Every node must know all neighbors in its AOI, so knowing enclosing neighbor is not enough. Properly discovering new AOI neighbors is thus the main mechanism to keep the AOI neighbors list and overlay topology correct.

First, a moving node will broadcast its position to its enclosing neighbors. "Broadcast" here does not mean hardware-level broadcast, but sending messages separately to every enclosing neighbor in one time-step. Then its enclosing neighbors will decide if the message should be forwarded by checking if their own enclosing neighbors are valid AOI neighbors that have not received the message (Figure 3).

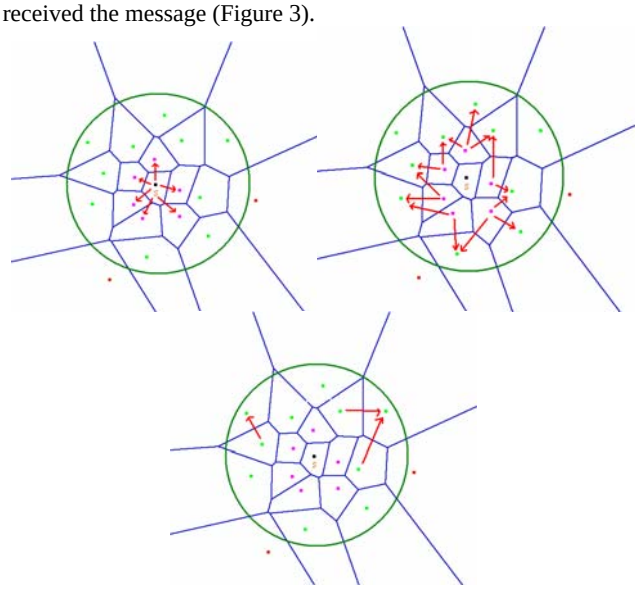
Figure 3. Message forwarding

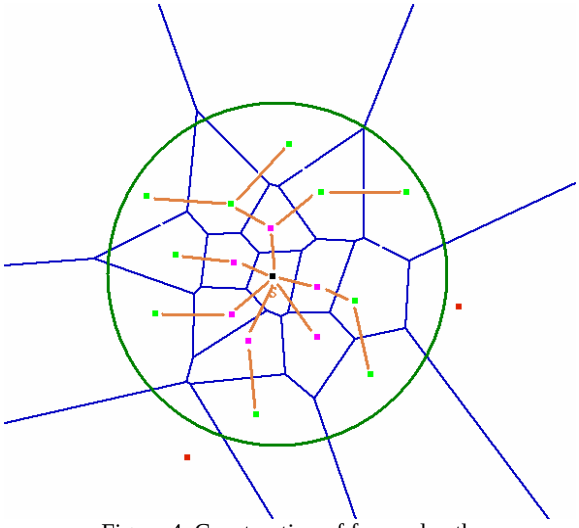

Figure 4. Construction of forward paths

If the message needs to be forwarded, the node will forward the message to its own enclosing neighbors and create a *forward path* during the repetition of this process. A forward path consists of a series of *forward records* kept at each of the intermediate *forwarder*. Each record is composed of a pair of destination's id and previous forwarder's id in a (destination, forward to) dataset (Figure 4). "Destination" indicates the moving node (or, equivalently, this update message's publisher) and "forward to" indicates the previous node which forwards this message. This process is repeated until the message no longer needs to be forwarded. This action also notifies all the AOI neighbors of the moving node of its current position.

Second, if a position update or a new neighbor notification is received, the receiving node will check whether this node is a valid AOI neighbor and process accordingly. There are two scenarios which we will discuss below:

A. Two nodes with unequal visibility

When an node broadcasts its position, it would add the ids of other nodes interested in its position (i.e. nodes whose AOIs cover the broadcasting node) to the message sent to forwarders, so that all the interested nodes may properly received its position update (even if they are outside of the broadcasting node's AOI). For example, after S broadcasts its message, a forward path of (S, A, B, P) is built, by A, B and P in a distributed fashion (Figure 5). A's forward record is (S, S), B's forward record is (S, A), and P's forward record is (S, B). Assuming A is not P's AOI neighbor, yet P is S's AOI neighbor, when it is P's turn to send position updates, it will include the id of S as one of the targets of its update and P's message will be forwarded to S. When B receives P's message, it will search from its forward records to find if the message needs to be forwarded. Then B will forward it to A because of the record (S, A). This process continues along the forward path and ensures that S eventually receives P's position update.

B. Two nodes with equal visibility

Nodes will simply broadcast their messages. As message update will propagate to the boundary of AOI, mutually visible nodes would receive each other's position update without problems. For example, assuming A and B are both within each other's AOI, then when A will receive the message B broadcasts, and vice versa.

These two scenarios combined will ensure proper neighbor discovery for all nodes.

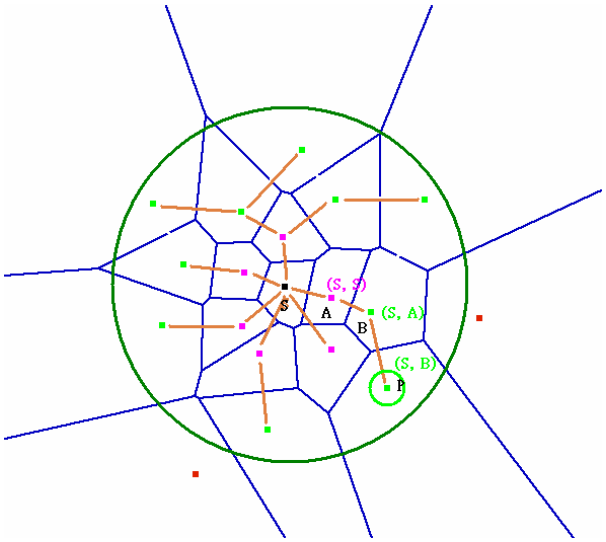

Figure 5. Example of forward records

3.3 Leave procedure

For leaving VON, a node simply disconnects. The leaving node's neighbors will recover the proper topology and discover new AOI neighbors, by following the regular MOVE procedure.

4 Discussion

We will discuss some problems in VON below:

- **Does overlay partition happen in VON?**
  Overlay partition refers to VON is cut into two or more separate partitions, and nodes in these partitions do not know each other. The key point of keeping VON alive is to ensure the full connectivity of all nodes and that Voronoi diagram is maintained correctly, so every scenario that makes Voronoi diagram incorrect has the possibility to cause overlay partition in VON. So it is possible for overlay partition to occur, for example, if nodes move too fast and cause enclosing neighbors to changed completely in one time step, then neighbor discovery may not be complete and may cause overlay partition.

- **Can VON self-recover from inconsistency?**
  When inconsistency occurs by packet lost or other causes, if enclosing neighbors are kept correctly, VON still can recovery itself by enclosing neighbor list update, but if too many nodes need to recover their neighbors, VON overlay partition may still happen.

- **Effect of network latency**
  By forwarding messages increased latency may be introduced. More forwarding will cause more latency. *Drift distance* [7] is a measurement of the distance between actual position and known position of neighboring nodes and is related to and latency. It is likely that the drift distance of forwarding model is higher than that in direct-connection model of VON. But we can always keep Voronoi regions correct if all messages sent in the next time step are confirmed. Because the latency with enclosing neighbors is lowest, so maybe we can see the nodes' position has little drift.

- **Simple comparison of forwarding model and direct-connection model**

Table 1. Comparison of forwarding and direct-connection model

	Direct-connection Model	Forwarding Model
Connection method	Direct connection with all AOI neighbors	Direct connection with only ENs, other AOI neighbors receive message through forwarding.
Drift distance	Almost 0	Related with the number of message forwarding.
Per node transmission size	Directly proportional to number of AOI neighbors	More average than DC model.
Overall transmission size (N=number of nodes, C= average AOI neighbors, E=average number of EN, M=multiplier)	C * N = O(N)	E * M * N = O(N)
Suitable environment	Small number of AOI neighbors/high latency environment	Large number of AOI neighbors/low latency environment

5 Conclusion

In this paper, we describe the design of the forwarding model of VON but without simulation results. Simulations have been done for the direct-connect model and are hosted on the VAST Project [8]. The simulation shows good results for topology consistency.

In the future, we will simulate the forwarding model and improve some problems that we have found. Combining the two models is another goal which we want to achieve in hope that VON may be used in larger number of network environments.

References

[1] J. Keller and G. Simon, "Solipsis: A massively multi-participant virtual world," in *Proc. Int. Conf. Parallel and Distributed Techniques and Applications (PDPTA 03)*, 2003, pp. 262-268.

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

[3] 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.

[4] S. Y. Hu and G. M. Liao, "Scalable peer-to-peer networked virtual environment," in *Proc. ACM SIGCOMM 2004 workshops on NetGames '04*, Aug. 2004, pp. 129-133.

[5] S. Y. 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

[6] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, "Spatial tessellations : concepts and applications of Voronoi diagrams," Chichester, Wiley, 2000. 2nd ed.

[7] 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.

[8] VAST Project homepage. Available at: http://vast.sourceforge.net/