Dynamic Clustering in Delaunay-Based P2P Networked Virtual Environments *

Matteo Varvello
Eurecom-Thomson
France
matteo.varvello@eurecom.fr

Ernst Biersack
Eurecom
Sophia-Antipolis, France
ernst.biersack@eurecom.fr

Christophe Diot
Thomson
Paris, France
christophe.diot@thomson.net

ABSTRACT

Classical client/server approaches for Networked Virtual Environments (NVEs) require considerable server resources to support a large number of players. On the opposite peer-to-peer (p2p) architectures achieve the required scalability at much lower cost. The feasibility of a p2p approach to NVEs depends heavily on the fact that a given user is interested only in a small part of the virtual world. For this reason, the Delaunay network is an appealing solution, which organizes peers according to their position within the NVE. However, in a NVE players typically tend to aggregate around some attractive points. As the cost of maintenance of the Delaunay network increases with player density and velocity, some peers may see a considerable volume of maintenance traffic. To address this issue, we propose a dynamic clustering algorithm: each peer in the network monitors his cost of maintenance and triggers the creation of a cluster as soon as the volume of traffic generated exceeds a given threshold. Members of a cluster then *expand* their coordinates to increase their reciprocal distances. In this way, decreasing the concentration of players we achieve a diminution of the maintenance cost. To evaluate our clustering scheme we simulate a simple NVE and run an experiment in Second Life. The results we obtained show that our solution is effective in keeping the amount of maintenance traffic below a chosen threshold.

Keywords

NVE, P2P, Delaunay triangulation, clustering, Second Life

1. INTRODUCTION

A Networked Virtual Environment (NVE) is a synthetic world where multiple participants interact via a virtual character called *avatar* [12]. NVEs were introduced in the 80s for military simulators, afterwards they were rapidly applied to games. A recent example is World of Warcraft (WoW), a NVE-based role playing game where hundreds of thousands of players interact simultaneously [15].

Between late 2006 and early 2007, Second Life (SL) became the most popular NVE. SL is a complex on-line society and economy within a 3D virtual world. Its users can explore, socialize, create and trade items and services [11].

Currently, WoW counts more than 8 Million subscribers and SL more than 7 Million users. These numbers are expected to grow in the next years and this makes scalability a critical issue in the deployment of future NVEs.

1.1 P2P Networked Virtual Environments

NVEs are designed on a client-server architecture. To sustain a continuously growing demand, multiple servers are used. For instance SL currently runs on 2579 servers and each server is responsible for an individual *sim* or 16 acres of virtual land [11]. The deployment and maintenance of such an infrastructure is complex and expensive.

Recently, p2p architectures have been proposed to solve the scalability issue of NVEs [5, 6, 2, 7]. Ideally, a p2p NVE can achieve self-scalability, i.e. exploiting end-users resources it can sustain a theoretically unbounded growth with a sub-linear impact on the performance.

An overlay network is an application-layer network topology built by p2p hosts. A naive approach for p2p NVEs consists in relying on a full mesh overlay where each peer periodically sends state updates to all the participants in the NVE [4]. This approach works at small scale over dedicated networks. For large NVEs, a smarter data dissemination strategy needs to be adopted.

A given user within an NVE is only interested in a small part of the virtual world. This awareness area is called *Area-Of-Interest* (AOI) and represents an important feature for the overlay construction. Ideally, state updates are shared only by players with overlapping AOI, limiting parallel connections and bandwidth consumption. This locality of interest is a key point for the feasibility of p2p NVEs.

1.2 Related Work

Colyseus [2, 1] is an interesting p2p approach to NVEs. The system uses as a publish/subscribe mechanism and is based on a range queries DHT [1]. Users *publish* objects in the DHT and *subscribe* to zones of the NVE to discover objects in their surroundings. This approach suffers from the indirection introduced by the DHT and from the overhead for the maintenance of a DHT in a real environment.

A different approach is adopted by Solipsis [6]. Each peer is aware of all entities lying in his AOI and collaborates with his neighbors to detect when new entities entered his own AOI. To ensure global connectivity, each node needs to be inside the polygon formed by its neighbors. However, neighbor discovery is occasionally incomplete, as incoming nodes may be unknown to directly-connected neighbors.

NetGames'07, September 19-20, 2007, Melbourne, Australia

S.-Y. Hu et al.[10] propose the usage of a Voronoi network, i.e. the dual of the Delaunay network, to eliminate the neighbor discovery problem of Solipsis. Each user maintains a Voronoi diagram of all AOI neighbors and directly connects to them. Object updates are handled by object owner, removing the indirection introduced by Colyseus.

Steiner and Biersack[13] propose an algorithm for the clustering of peers in a 3D Delaunay network for NVE. The main idea is to classify links between adjacent peers in the Delaunay network into *short intra-cluster* and *long inter-cluster* links. The clustering is useful for faster navigation in the Delaunay network and to reduce the number of messages a node receives when he travels through the NVE. However the proposed scheme suffers from the difficulty to define a clustering threshold and does not take into account mobility.

1.3 The Delaunay Network

A Delaunay Network (DN) is an overlay network topology based on the Delaunay Triangulation (DT). *A DT for a set of points S is constructed by all the possible triangles in S whose circumcircles do not contain any point of S.*

The coordinates of players in the NVE are used to generate the DT. In this way, a flexible overlay that maps the concept of neighborhood in the NVE is built. Logical links in the overlay reflect player connections in the NVE.

An inner node of a two dimensional (2D) DT with a sufficiently large population, i.e. a node not on the convex hull[1] of the triangulation, has an average number of neighbors limited to 6 [3]. This means, that each peer in the respective DN is currently connected to a finite number of virtual neighbors independent from the population of the NVE.

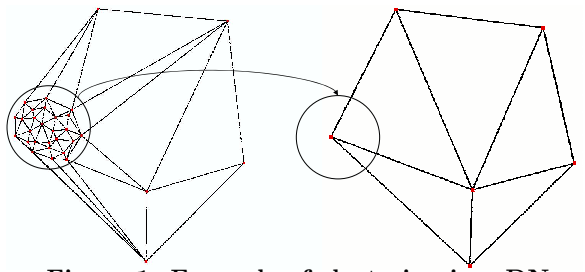


Figure 1: Example of clustering in a DN

To maintain a valid triangulation, each node has to monitor the position of its one hop neighbors [9]. In this way the overlay built on top of the triangulation at each instant t is always reflecting effective positions of players in the NVE.

We define the maintenance cost of a DN as the number of messages each peer exchanges in the network to maintain a valid DT. As it can be expected, in a dynamic environment this cost increases with player density and velocity.

Since in existing NVEs the popularity of different areas is believed to follow a power-law distribution [2], players concentrated around some attractive points may see a considerable volume of maintenance traffic. To address this issue, we propose a dynamic clustering algorithm: each peer in the network monitors his own maintenance cost and triggers the creation of a cluster as soon as the volume of traffic generated exceeds a given threshold.

Members of a cluster are organized via an *expanded* DT, i.e. inter-node links are stretched to reduce the effect of the density. The cluster is represented in the original DT as a node with coordinates equal to the center of the cluster. This node is monitored by a cluster-head to maintain a connection between the cluster and the rest of the NVE. Figure 1 shows intuitively how a DN looks like after clustering.

The rest of the paper is organized as follows: Section 2 proposes a cost model to understand the impact of dynamic nodes in a DN. In Section 3 we describe a dynamic clustering for Delaunay-based p2p NVEs which we evaluate in Section 4. Finally, Section 5 concludes the paper.

[1]The convex hull for a set of points X in a vector space $V \in \Re^k$ is the minimal convex set containing X.

2. PROBLEM FORMALIZATION

In this Section, we propose a metric describing the maintenance cost of a DN. In the following we call a *peer* the end-user participating in the NVE. The virtual representation of a peer is called an *avatar*. Avatar coordinates are used to define the position of a *node* for the DT. We assume a finite population of N peers that never leave the network, i.e. there is no churn. Table 1 summarizes the parameters we use in the analysis along with a brief explanation.

DEFINITION 1. *Circumcircle: a circle which passes through all three vertices of a triangle.*

DEFINITION 2. *The Delaunay triangulation of a set of N points in $\Re^2$ is a triangulation of points $DT(N)$ such that no point p lies inside the circumcircle of any triangle in $DT(N)$.*

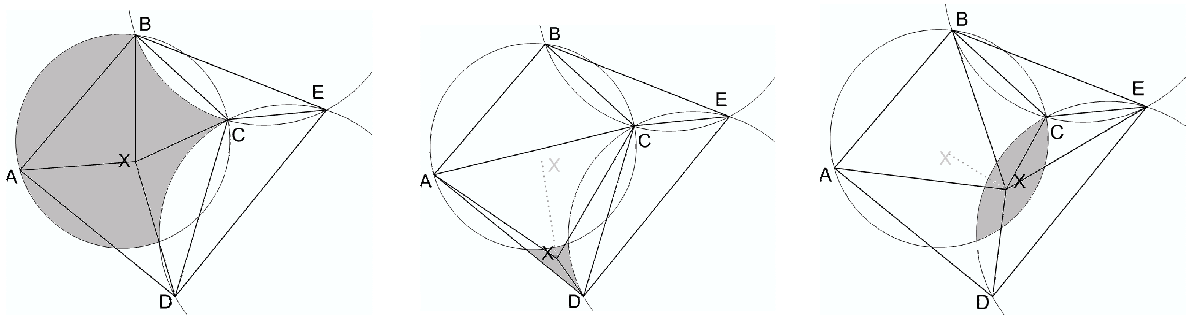
Table 1: Table of parameters

PARAMETER	DEFINITION
C_{ijk}	circumcircle of triangle T_{ijk}
T_{ijk}	triangle defined by nodes i, j, and k
r_{ijk}	radius of circle C_{ijk}
C_m	maintenance cost of a DN
$d1$	average distance among 1-hop nodes
$d2$	average distance among 1,2-hop nodes
ρ	local density in the NVE
N	number of users
N_n	number of 1-hop neighbors
R_b	rate of keep-alive messages
R_f	rate of flip operations
v_x	constant speed of node X
L_{min}	minimum allowed path before flip
L_{max}	maximum allowed path before flip
k	number of packets per flip operation

The maintenance of a DN is performed by all peers participating in the NVE. The maintenance cost consists of two different parts. On one hand, each peer has to monitor via keep-alive messages his one hop neighbors at a fixed rate R_b. In this way, each node is always aware of the positions of its neighbors. On the other hand, when avatars move and Definition 2 is violated, peers have to communicate to set links active ore inactive. We call this event a *flip operation* [3] and it is necessary to maintain a valid triangulation. Equation 1 gives a formulation for the maintenance cost expressed as rate of messages that need to be exchanged among peers.

$$C_m = N_n \cdot R_b + k \cdot R_f \quad (1)$$

The first term, $(N_n \cdot R_b)$, represents the rate of keep-alive messages a peer exchanges with his one-hop neighbors. At steady state, it has been shown that $N_n = 6$ for a 2D-DT [3].



(a) Delaunay triangulation of nodes A,B,X,D,C and E

(b) Node X leaves circumcircle C_{abc}, link XB flips to AC

(c) Node X enters circumcircle C_{cde}, link CD flips to XE

Figure 2: Evolution of a Delaunay Triangulation via flip operations

The second term, $(k \cdot R_f)$, identifies the rate at which messages are exchanged during the reconstruction of the DT. The parameter k indicates the number of packets required per flip operation and depends on the way the distributed computation of the DT is performed. R_f is the rate at which flip operations are performed, which intuitively depends on avatar density and velocity.

A flip operation is due whenever the position of a generic node X violates the basic property of a DT (see Definition 2). The gray area underlined in Figure 2 is the portion of the triangulation where node X is allowed to move without triggering any flip operation. Two scenarios can cause a flip operation and are shown in Figures 2(b) and 2(c).

Figure 2 shows the evolution of a DT via flip operations. We start with an initial DT involving nodes A,B,X,D,C and E (Figure 2(a)). Node X moves out from circumcircle C_{abc} and so triangle T_{abc} can be created; link XB flips to AC (Figure 2(b)). Node X enters circumcicle C_{cde}, triangle T_{cde} is disrupted and triangles T_{xce} and T_{xde} are created; link CD flips to XE (Figure 2(b)).

We call L_{min} and L_{max} respectively the minimum and maximum path within the gray area in Figure 2. To derive an expression of L_{min} and L_{max} and consequently R_f we construct a regular geometric structure. For this purpose we introduce parameters $d1$ and $d2$:

- $d1$: average distance between node X and his one-hop neighbors (nodes A,B,C,D in Figure 2).
- $d2$: average distance between one-hop neighbors of node X and the two-hop neighbors that are vertex of a triangle whose circumcircle intersects the convex hull defined by one-hop neighbors of node X (node E in Figure 2).

Figure 3 shows the geometrical structure obtained. The convex hull built by the $N_n = 6$ neighbors of node X is a regular hexagon with side length equal to $d1$. All triangles constructed with two-hop neighbors are isosceles triangles; the two equal sides have length equal to $d2$, the remaining side is equal to $d1$.

All vertices of the hexagon are points on the same circle with radius equal to the side of the hexagon, i.e for construction $r_{abc} = d1$. The free area of movement for node X is the grey zone in Figure 3. The maximum path on which node X is allowed to move is bounded by the radius of C_{abc}, i.e. $L_{max} = d1$.

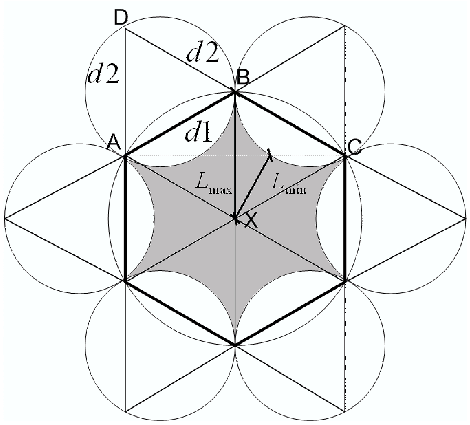


Figure 3: The regular geometric structure built with the introduction of $d1$ and $d2$

The simplifications introduced allow us to give a closed formula for $L_{min}(d1, d2)$ (Equation 2). To derive it, we consider the distance between node X and the the circular segment intercepted by chord BC. A complete derivation can be found in [14].

$$L_{min}(d1, d2) = d1 \cdot (\frac{\sqrt{3}}{2} - \frac{d1}{2 \cdot (4d2^2 - d1^2)}) \qquad (2)$$

The results above tell us that the minimum path on which a generic node X can move in the DT without triggering a flip operation is smaller than $d1$. We introduce the local density ρ as the number of avatars concentrated in the area defined by $d1$ and $d2$. It follows that increasing ρ in an area of a NVE, i.e reducing $d1$ and $d2$, reduces the minimum path on which an avatar can move without incurring in a flip operation.

Each time a flip operation is due, we reconstruct the geometric structure of Figure 3 by computing the new values of $d1$ and $d2$. Assuming as hypothesis that $d1$ and $d2$ do not vary too much in the surroundings of node X during a short time T, we can say that $\overline{L_{min}(d1, d2)} \approx L_{min}(d1, d2)$. Then, to derive an expression of R_f we compute the time before a flip operation occurs considering node X moving at a constant speed v_x on the direction of $L_{min}(d1, d2)$. Finally, Equation 3 gives a formulation of C_m.

$$C_m = 6 \cdot R_b + k \cdot \frac{v_x}{L_{min}(d1, d2)} \quad (3)$$

In Figure 4 we plot the evolution of C_m as a function of $d1$, $d2$ and v_x. Figure 4 summarizes the analytical results derived: in a dynamic environment, the maintenance cost C_m of a DN increases with the density and velocity of avatars. Moreover, the dependence of L_{min} on $d1$ and $d2$ shows how C_m for a specific peer is affected only by the local behavior of his neighbors.

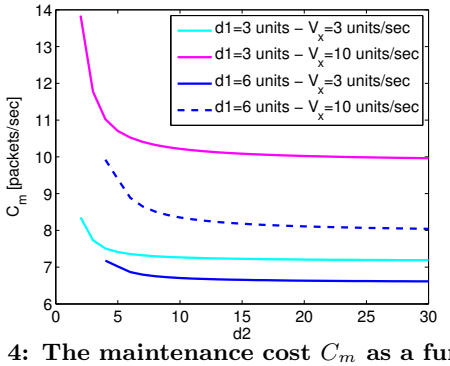


Figure 4: The maintenance cost C_m as a function of velocity and inter-node distances ($d1$, $d2$)

Figure 4 shows how the cost of maintenance C_m for fixed $d1$ and v_m decreases by increasing $d2$. In fact, the circular segment intercepted by the chord BC on C_{abc} (Figure 3) becomes more and more flat, and $L_{min}(d1, d2)$ approaches $L_{max} = d1$. For $d2 \to \infty$, $L_{min}(d1, d2)$ is approximated as $L_{max} = d1$ and C_m converges to a constant value.

These results illustrate the drawback of the DT, which maintenance traffic can actually limit the scalability of the respective DN. Therefore, we propose a dynamic clustering technique in order to limit the volume of maintenance traffic under a given threshold.

3. DYNAMIC CLUSTERING IN DELAUNAY-BASED P2P NVE

We consider the system at steady state with N peers organized in a DN. A distributed computation of the DT is used [10]. We introduce the parameter B_t, i.e. the target volume of maintenance traffic per peer within the DN.

Each peer in the DT monitors his maintenance cost C_m. If peer A notices $C_m \geq B_t$, he is implicitly detecting a local density higher than the desired threshold. Peer A reacts by proposing his neighbors the creation of a cluster and selecting a peer to be the responsible of the cluster (Algorithm 1); we refer to this node as the cluster-head (CH).

```
Compute_C_m();
if C_m ≥ B_t then
    foreach peer i in N_n do
        propose_clustering();
    end
    select_cluster-head();
end
```

Algorithm 1: Cluster initialization

We do not focus on the CH selection, since it is irrelevant for the behavior of the clustering. In the following, we suppose that the CH is represented by a super-peer that can easily manage the overhead introduced by a cluster.

The neighbors of peer A check their value of C_m and decide whether to take part in the clustering process. They spread this information via the normal exchange of messages. Peers who agree to the creation of the cluster contact the CH.

The CH collects all pending requests for clustering. For each cluster c he detects, he computes the area (S_c) occupied by the cluster and its center (X_c, Y_c). Then, the CH informs all involved peers of the cluster creation (Algorithm 2).

```
collect_clustering_requests();
foreach cluster c detected do
    compute S_c, X_c, Y_c;
    foreach peer i in N_c do
        send(S_c, X_c, Y_c);
    end
end
inform_extra-cluster_nodes();
monitor_the_cluster();
```

Algorithm 2: Cluster definition

External nodes refer to the cluster considering it as a virtual node at coordinates X_c, Y_c. The CH monitors this virtual node as if it was a normal node in the triangulation, i.e using heartbeat messages and flip operations (Algorithm 2). In this way the connection between the cluster and the rest of the NVE is kept (Figure 1).

We call N_c the population of a cluster and $\rho_c = \frac{N_c}{S_c}$ the density within a cluster c. Members of a cluster *expand* their coordinates, i.e we define $S_c^{'} = E * S_c$ with $E > 1$. The expansion is unique for every cluster member so that the angular relationship among nodes remains the same, i.e. the intra-cluster Delaunay links are not modified. The cluster density becomes $\rho_c^{'} = \frac{\rho_c}{E}$.

Cluster creation works in a hierarchical way, i.e clusters can be created within clusters if the expansion results not to be enough or if N_c grows too much.

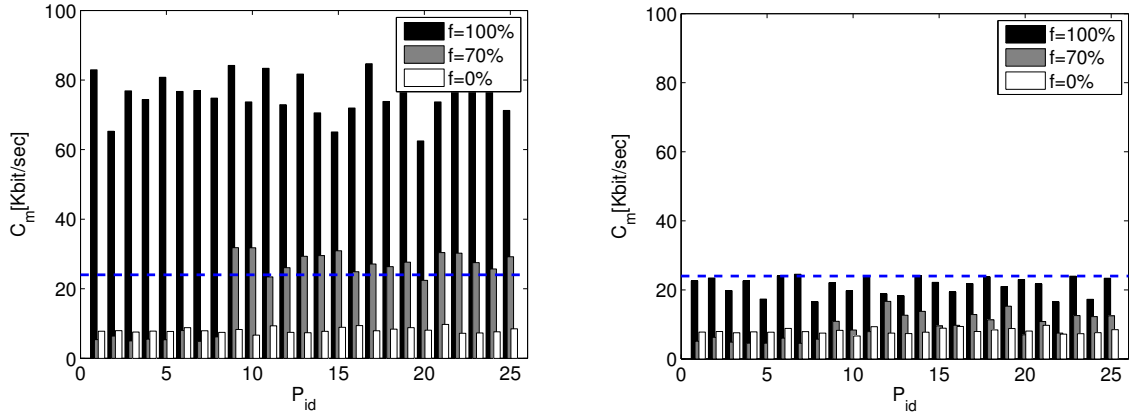
An extra-cluster node joins a cluster c when its avatar AOI intersects the area S_c. In fact, as nodes within a cluster are allowed to move within S_c, there is a non zero probability they intercept the AOI of the extra-cluster node. In the same way an intra-cluster node leaves c when his AOI does not intersect any more S_c. In this way we ensure visibility among avatars going in and out of a cluster.

The size of the cluster S_c is computed at the creation of the cluster and is not modified when avatars join and leave. This means that, the probability of the cluster to become empty decreases with avatar departures. When a cluster is empty, the CH removes it from the DT. It results as a disconnection event of a node in the DT.

The dissemination of state updates in the DN is done via flooding with AOI filtering, i.e. each node forwards data only if it is relevant for the recipients. For this reason, the overlay is constructed considering positions of avatars in the NVE.

The effect of the expansion within a cluster is to increase the distance among nodes, resulting in a loss of mapping between the DT and the effective positions of avatars in the NVE. Therefore, the AOI filtering is no more effective.

If a cluster was created, an high density of avatars was detected. So it is reasonable to assume the size of the AOI



(a) Delaunay Network

(b) Clustering in a Delaunay Network

Figure 5: Maintenance cost C_m - Synthetic Traces - $N = 25$, $X = Y = 300$ *units*, $T = 10$ *mins*, $v_i = 10$ *units/sec*

to be larger than S_c. Implicitly, the AOI filtering is no more needed and a simple broadcast operation within the cluster is enough. The introduction of clustering can give significant benefits also for an efficient distribution of the state updates in a dense environment.

4. EVALUATION

We implemented a simulator in Matlab to evaluate the performance of our clustering scheme. A centralized unit maintains the DT among nodes and manages the clustering. The CHs are represented by virtual peers and the overhead introduced by the clusters is shared among cluster members.

In the simulator, each flip operation counts as one packet sent to the N_n neighbors of the involved nodes. The packet size is set to 100 *bytes*. We do not count heartbeat messages, since they represent an additive cost for all peers. We consider one level of clustering only. The movement models of avatars are gathered in two different ways:

- *Synthetic Traces*: we generate a 2D NVE of size X, Y populated by N avatars moving according to a Random Waypoint model [8]. In the NVE we define N_d domains of size X_d, Y_d. A percentage f of the N avatars is attracted by the N_d domains, i.e. they spend the entire simulation time T within them. The motivation behind this model is to simulate users interest in some specific zones of the NVE.

- *Real Traces (Second Life)*: users in SL have the possibility to create objects via a scripting language called LSL. We deployed a crawler object to collect data about avatar positions. The traces gathered are sent back via a http POST request to an external server.

4.1 Synthetic Traces

We consider $N = 25$ avatars in a virtual world of size $X = Y = 300$ *units*, i.e. the size of a land in SL. We define $N_d = 2$ domains of size $X_d = Y_d = 25$ *units*. The simulation time is $T = 10$ *mins* and the speed of avatars is $v_i = 10$ *units/sec*. We are interested in observing interactions among avatars, so we choose high speeds during a short simulation time.

The goal of the simulation is to show how our scheme reacts to aggregation of avatars in a generic virtual world. To have a concrete example we had to make some assumptions. However all these parameters are application dependent and affect our solution only in the choice of B_t.

Figure 5(a) shows the maintenance cost C_m of a DN for the virtual world described above. Observing the amount of maintenance traffic in the DN we see that a meaningful value for the cluster threshold is for instance $B_t = 24$ *kbps*[2].

In Figure 5(a) we clearly see the dependence of C_m from the local density of nodes in the triangulation. When $f = 100\%$, all avatars are concentrated around the points of interest. The local density ρ and the maintenance cost C_m take large values. When $f = 0\%$, avatars are uniformly distributed in the NVE. The local density ρ and the maintenance cost C_m result very small.

Figure 5(b) shows the effect of our clustering scheme. We see how C_m is always bounded by B_t.

When $f = 100\%$, all peers agree in creating two clusters in correspondence of the two zones of attractions. Figure 5(b) shows the benefits as a reduction of C_m.

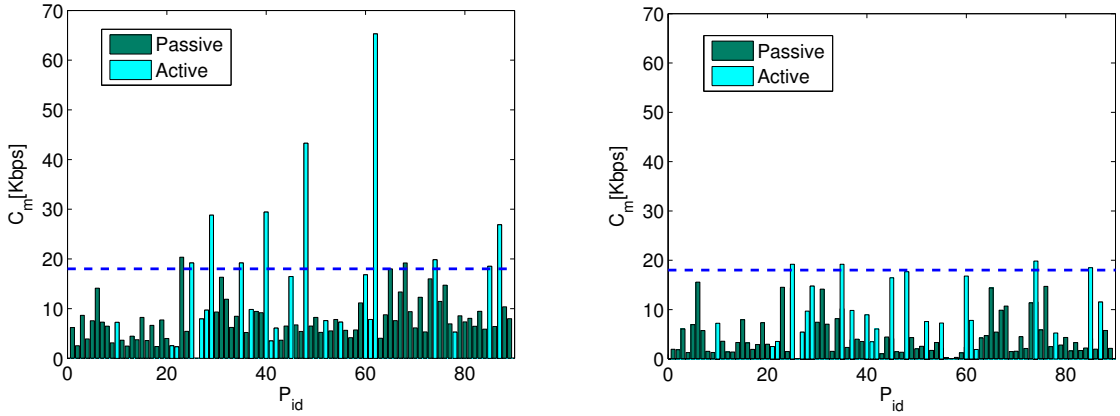
When $f = 70\%$, we see an intermediate scenario. The first 8 peers are not attracted by the two domains. Their distribution is uniform in the NVE and Figure 5(a) shows values of C_m smaller then B_t. The remaining 17 peers suffer for an high local density and find great benefits from the clustering process (Figure 5(b)).

Finally when $f = 0\%$, C_m never exceeds B_t, so the basic DN is enough. In this case no peer is ever proposing to create a cluster and the clustering algorithm is never executed.

4.2 Real Traces (Second Life)

Second Life is a NVE constituted by independents lands (*sim*). The crawler object we deployed can monitor an entire land and can be used in public or private lands where the owner allows visitors to instantiate objects. We monitored the Public Help Island for $T = 30$ *mins* and we encountered $N = 89$ different avatars. Figure 6(a) shows the maintenance cost C_m of the DN constructed by the N avatars. From this analysis, we decided to set $B_t = 18$ *kbps* as threshold for the clustering scheme.

[2]In a generic application, we could arbitrarily decide a target maintenance traffic for the overlay.



(a) Delaunay Network

(b) Clustering in a Delaunay Network

Figure 6: Maintenance cost C_m - Real Traces (Second Life) - $N = 89$, $T = 30$ *mins*

Observing Figure 6(a) we notice that C_m is pretty low for many peers. The reason is that many peers connect to Second Life only to find a place to chat, so their velocity is close to zero. We decided to distinguish between two categories of peers, *passive* and *active*. *Passive*/*active* peers have an average velocity smaller/higher than 1 *unit*/*sec*. In Figure 6 we indicate *active* peers with a light color and *passive* peers with a dark one.

Avatars in SL can *listen* to chats in a range of 20 meters: for this reason, avatars tend to aggregate. Since in a DT static nodes do not cause any flip, also if the density is large, the maintenance cost C_m turns out to be low.

The dynamism of *active* peers allows to discover high density zones, since with their movement they cause several flip operations. When the maintenance cost becomes larger then B_t, *active* peers propose the creation of clusters. As a consequence, some *passive* peers are included in clusters due to their position in the NVE and both *active* and *passive* peers see a considerable reduction in C_m.

The velocity of avatars is not the only cause of flip operations as we can see in Figure 6(a): peers 23, 64 and 67 experience a considerable value of C_m but are not *active* peers. We noticed for these specific peers an high level of churn, i.e. they connected more then 10 times to the Public Help Island during the time T. A churn event counts as an insertion/deletion of a node in the DT, so it affects the maintenance cost C_m of the specific peer.

5. CONCLUSION AND FUTURE WORK

This paper presents a dynamic clustering scheme for p2p NVEs based on the Delaunay Network. We propose an analysis on the maintenance cost of a Delaunay network that shows the effect of player density and velocity. Monitoring his own maintenance cost, a peer in the network reacts to high node density by clustering. To reduce the effect of density, nodes within a cluster expand their coordinates, i.e. increase their reciprocal distances.

To evaluate our clustering scheme we simulated a NVE under a synthetic model of movement and we monitored a land in SL to obtain realistic traces of movement. The results we obtained show that our solution is effective to keep the amount of maintenance traffic below a chosen threshold.

One avenue for future work is to implement a distributed version of the algorithm and to test it in a real environment. We believe that an efficient data dissemination scheme could also be built on top of the proposed overlay. Moreover, we are working on a complete crawler for Second Life in order to understand more on avatar and player behavior. We envisage to run other experiments on data sets collected from objective-based game such as World of Warcraft.

6. REFERENCES

[1] S. S. Ashwin Bharambe, Mukesh Agrawal. Mercury: Supporting Scalable Multi-Attribute Range Queries. In *Proc. SIGCOMM 2004*.

[2] A. Bharambe, J. Pang, and S. Seshan. Colyseus: A Distributed Architecture for Online Multiplayer Games. In *NSDI '06*, 2006.

[3] A. Bowyer. Computing Dirichlet Tessellations. *Computer journal*, pages 162–166, 1981.

[4] L. Gautier and C. Diot. Design and Evaluation of Mimaze, a Multi-Player Game on the Internet. In *Int. Conf. on Multimedia Computing and Systems*, pages 233–236, 1998.

[5] S.-Y. Hu, J.-F. Chen, and T.-H. Chen. VON: A Scalable Peer-to-Peer Network for Virtual Environments. *Network, IEEE*, 20(4):22–31, 2006.

[6] J. Keller and G. Simon. SOLIPSIS: A Massively Multi-Participant Virtual World. In *Int. Conf. on Parallel and Distributed Techniques and Applications*, 2003.

[7] B. Knutsson et al. Peer-to-Peer Support for Massively Multiplayer Games. In *Proc. Infocom 2004*, Mar. 2004.

[8] Mobility Models. `http://ica1www.epfl.ch/RandomTrip/`.

[9] P. Müller. Scalable Localized Histogram Aggregation for p2p MMOGs. Master's thesis, ETH Zurich, July 2005.

[10] M. Ohnishi, R. Nishide, and S. Ueshima. Incremental Construction of Delaunay Overlay Network for Virtual Collaborative Space. In *Third Int. Conf. on Creating, Connecting and Collaborating through Computing*, 2005.

[11] Second Life. `http://www.secondlife.com/`.

[12] S. Singhal and M. Zyda. *Networked Virtual Environments: Design and Implementation*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.

[13] M. Steiner and E. W. Biersack. DDC: A Dynamic and Distributed Clustering Algorithm for Networked Virtual Environments based on P2P networks. In *Proc. of the 9th IEEE Global Internet Symposium 2006*, Spain, Apr. 2006.

[14] M. Varvello. Dynamic Clustering in Delaunay-Based p2p Networked Virtual Environments. Technical report, 2007.

[15] WoW. `http://www.worldofwarcraft.com/`.