HyVVE
A Voronoi based Hybrid Architecture for Massively Multiplayer On-line Games

Laura Ricci, Luca Genovali and Barbara Guidi
Department of Computer Science, University of Pisa, Pisa, Italy

Keywords: Distributed Virtual Environments, Voronoi, Delaunay, Load Balancing.

Abstract: Massively Multiplayer Online Games (MMOGs) have gained a lot of popularity in recent years. The problem of defining a proper architecture supporting MMOGs is still a research challenge because the classical client server architecture mainly adopted in commercial applications presents several drawbacks like unsatisfactory scalability and limited fault tolerance. This paper presents HyVVE (Hybrid Voronoi Virtual Environments), an hybrid architecture exploiting both server and P2P nodes. HyVVE exploits a Voronoi tessellation of the virtual world to distribute the load for the management of the virtual entities among the server and the peers. The paper presents a set of experimental results proving the effectiveness of our approach.

1 INTRODUCTION

Distributed Virtual Environments (DVE) (Ohnishi et al., 2005a; Bharambe et al., 2006; Yu and Vuong, 2005; Knutsson et al., 2004; Carlini et al., 2013) enable geographically distant users to communicate, interact and collaborate within a virtual environment. In particular, online gaming entertainment has acquired lots of popularity in the last years from both industry and research communities. The market size of online gaming has received a 5 billion evaluation in 2010, while the number of total users have reached around 20 million worldwide.

Currently, most commercial Massively Multiplayer Online Games (MMOGs) rely on a client server architecture which supports a straightforward management of the main functionalities of the MMOG, such as user identification, management of the state of the virtual world, synchronization between players, and billing. However the most important drawback of these architectures is their limited scalability which prevent a satisfactory playability when huge amounts of concurrent users are playing simultaneously.

Recently several solutions based on P2P based MMOG have been presented. The main advantage of these solutions is that they are inherently scalable because when the amount of users grows, more resources are added to the infrastructure. However, a pure P2P-based approach is barely feasible. As a matter of fact, the lack of a centralized authority makes it complex to enforce security, consistency of concurrent updates to the state of the virtual world and persistence of its state when a few players are present in the virtual world.

An interesting alternative exploiting the advantages of both the client/server and of the P2P architecture is to define an hybrid solution which properly distributes the functionalities of the MMOG among the server and the peers. For instance, the state of the virtual world may be partitioned and distributed to the server and to the peers by exploiting the locality property characterizing most MMOGs. As a matter of fact, an avatar generally interacts with other entities (avatars and passive entities) located in its proximity, i.e. in its *Area of Interest, AOI*. A simple solution is to define a partition of the state of the MMOG where a peer manages the entities in its AOI, while the server manages entities located in areas not covered by the AOI of any peer. This solution is simple, however it does not resolve the problem of the ownership of the entities which are located in the intersection of the AOI of a set of peers.

In this paper we propose *HyVVE, Hybrid Voronoi-based Virtual Environment*, an hybrid architecture based on a Voronoi Tessellation (Aurenhammer, 1991) of the virtual world which exploits the locality of MMOGs.

Given *n sites* in an euclidean space, a *Voronoi tessellation* partitions the virtual world into *n areas* such that the area corresponding to a site *n* includes all the points which *are closer* to *n* with respect to any other site.

Ricci L., Genovali L. and Guidi B..
HyVVE - A Voronoi based Hybrid Architecture for Massively Multiplayer On-line Games.
DOI: 10.5220/0004531900150023
In *Proceedings of the 4th International Conference on Data Communication Networking, 10th International Conference on e-Business and 4th International Conference on Optical Communication Systems* (DCNET-2013), pages 15-23
ISBN: 978-989-8565-72-3

In a *Voronoi based* MMOG, the position of each peer in the virtual world is exploited to define the *Voronoi tessellation* of the world which is exploited to assign entities to the peers/server. In *HyVVE* the entities assigned to a peer are those belonging to the intersection of its *CCAOI*, a circular area representing a super-set of its *AOI*, and of its Voronoi area. We consider a superset of the *AOI* of a peer to implement a prefetching of the entities close to the border of the *AOI* of the peer. The server manages all the entities belonging to areas not covered by the *CCAOI* of any peer. The solution where a peer manages all the entities in its Voronoi area is not feasible, because when a few peers are present in the virtual world, the number of entities assigned to a peer would be huge, and the peer would not be able to support such a load, especially if its computational capability is not high. *HyVVE* supports a load distribution mechanism which scale in a natural way. When the world is scarcely populated, most entities are mapped to the server. However, the server load is not high in this case, because a few entities are accessed and modified by the peers. When the number of the peers increases, the amount of entities assigned to the peers increases proportionally so that the load is distributed between the peers. In a crowding scenario, i.e. when a huge amount of peers gather in the same region of the world, the entities located in that region are managed by these peers, while the server manages entities belonging to inhabited regions of the virtual world. In this scenario, where peers are close to each other and their *AOIs* intersect, the ownership of the entities is defined by considering the Voronoi regions of the peers. When a conflict occurs because an entity is located in the *CCAOI* of a set of peers, the Voronoi tessellation is exploited to determine the owner of the entity. Since each point of the space is mapped to a unique Voronoi Area, each entity of the virtual world is assigned to a unique node peer.

The paper is organized as follows. Section 2 discusses the state of art in the area of Voronoi based virtual worlds. Section 3 introduces the mathematical notions which are the basis of our approach, while section 4 discusses how Voronoi tessellation can be exploited for the definition of virtual environments. Section 5 describes the architecture of *HyVVE*. A set of experimental results are presented in Section 6, while Section 7 reports some conclusion and discusses future works.

2 RELATED WORKS

Voronoi diagrams and Delaunay-based overlays are a well know solution to maintain network topology for P2P virtual environments.

One of the first works along this line is VON (Hu et al., 2006; Hu et al., 2008; Jiang et al., 2008). VON exploits a Voronoi division of the DVE in order to manage event dissemination in a scalable manner. VON defines an overlay such that each peer maintains a direct connection with all the peers within its AOI. In order to maintain overlay connectivity, each peer also has a direct link with peers that may also be outside of the AOI. To reduce bandwidth consumption, VON has been further upgraded with an enhanced event dissemination system (Jiang et al., 2008) and state management (Hu et al., 2008).

VoroGame (Buyukkaya and Abdallah, 2008; Cavagna et al., 2008; Cavagna et al., 2009), proposes an hybrid architecture for the management of passive entities. Their architecture combines a Voronoi-based network and a Distributed Hash Table (DHT). Two different peers, one for each overlay, are responsible for each entity in the DVE. Voronoi nodes are responsible and maintain a copy for any of the entities that are in their Voronoi area. They also maintain, for each of these entities, a list of peers that have to be notified for a state change of the given entity. This list is periodically sent to the corresponding DHT node, whose task is to broadcast state updates.

The work in (Varvello et al., 2007) proposes a solution to deal with cluster of players in Delaunay-based topologies. They employ a flooding messaging strategy to spread notification inside the AOI. However, when a peer detects the message rate to exceed its maximum capacity, it triggers a procedure for cluster management. This procedure logically collapses the cluster to a single point, allowing communications to temporarily skip many small neighbouring Voronoi regions, which helps reduce the communication overhead. The approach has been proved effective with realistic movement traces from Second Life.

The definition of a P2P overlay for *MMOG* based upon *Voronoi Diagrams* has been investigated in (Bonotti et al., 2007; Ricci and Salvadori, 2007; Genovali and Ricci, 2009; Genovali and Ricci, 2008a; Genovali and Ricci, 2008b; Ricci et al., 2011) (Jiang et al., 2008; Lee and Lam, 2008; Ohnishi et al., 2005b; Hu et al., 2006). Some recent proposals (Ohnishi et al., 2005b; Varvello et al., 2007; Hu et al., 2006) have discussed the benefits of defining an overlay where the P2P connections correspond to the links of a Delaunay Triangulation generated by considering the locations of avatars of the DVE (Ghaffari et al., 2009). According to this proposal, each peer is paired with a site of a Voronoi diagram defined on the virtual space and the position of the peer is exploited to define the space partition. In this way, the area corresponding to a peer *P* includes all the points of the DVE which *are closer* to *P* with respect to any other. The *Delaunay Triangulation* corresponding to the *Voronoi tessellation* defines the *P2P* overlay connecting the peers.

In a *MMOG* any event generated by a peer should be notified to any other peer in its *AOI*. This notification may be implemented through a *AOI-cast* mechanism (Albano et al., 2009), i.e. an application level multicast constrained within the boundary of the area of interest. *Flooding* the heartbeat through the *Voronoi links* generates a large amount of redundant messages and presents evident scalability problems. A more refined approach dynamically computes a *spanning tree* on the *Voronoi links* including all the peers of the *AOI* and exploits this tree to notify the heartbeat. Both solutions are based on *forwarding*, i.e. any heartbeat is routed to the peers in the *AOI* through a sequence of intermediate peers. An obvious drawback of these solutions is the high latency in the delivery of an event, especially in *crowding scenarios*, i.e. when a set of peers lie close to each other in the *virtual world* and their *AOI* overlap. In this case *several routing hops* may be required to notify an event due to the large amount of peers located in the *AOI*. The resulting *latency* may be not tolerable in *MMOG* and may compromise the interactivity of the application. On the other hand, since the number of *Voronoi neighbours* is 6 on average (Aurenhammer, 1991), in this solutions a peer manages a small number of connections.

An alternative solution (Hu et al., 2006) defines *direct links* between a peer and any other one in its *AOI*. The resulting overlay includes these links besides the Voronoi ones, which have to be maintained to guarantee the connectivity of the overlay. This solution minimizes the latency, but increases the number of connections of each peer. In a crowding scenario like the one previously described, a peer should manage a large number of connections, since a large amount of peers are located in its *AOI*.

The problem of maintaining the *Voronoi* structure of the overlay in a dynamic *P2P* environment, like a *MMOG*, where the positions of the peers change continously and no centralized coordination entity does exist, is a challenging issue. It is worth noticing that several distributed algorithm for the management of the Voronoi overlay have been recently proposed. (Baraglia et al., 2012; Lee and Lam, 2008; Ohnishi et al., 2005a; Kato et al., 2006) propose protocols to build and maintain Delaunay triangulation-based overlay networks. The approach present in (Baraglia et al., 2012) is based on a gossip approach which results particularly suitable in a dynamic environment like a DVE.

3 VORONOI DIAGRAMS AND DELAUNAY TRIANGULATIONS

This section introduces the basic mathematical concepts we have exploited for the definition of HyVVE.

A *Voronoi diagram*, (Aurenhammer, 1991) also referred as *Voronoi tessellation*, is a special kind of decomposition of a metric space determined by the distances of the points of the spaces to a specified discrete set of entities in the space, i.e. the *sites*.

Let us denote the Euclidean distance between two points *p* and *q* by *dist(p,q)*.

Definition 1. *Let* $S = \{s_1, s_2, \ldots, s_n\}$ *be a set of n distinct points in the plane, i.e.* the sites. *The* Voronoi Diagram *of S is a partition of the plane into n cells, one for each site in S, such that the point q belongs to the cell corresponding to a site* s_i *if and only if* $dist(q, s_i) < dist(q, s_j) \forall s_j \in S, i \neq j$.

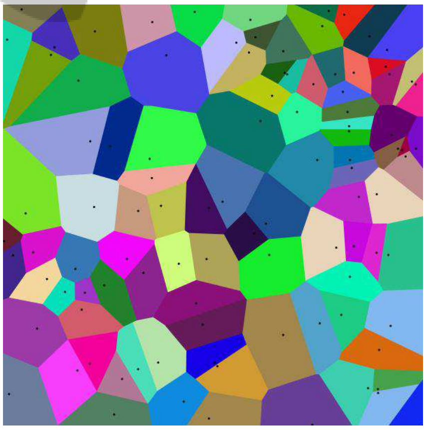

Figure 1: A Voronoi diagram.

In the following, we will denote the Voronoi Diagram of S by $Vor(S)$ and the cell corresponding to a site s_i by $V(s_i)$. Fig. 1 shows the Voronoi Tessellation defined by the set of sites represented by black dots. Each colored region represents $V(s_i)$, where s_i is the site corresponding to the black dot belonging to the region.

A *Delaunay Triangulation* is a mathematical structure dual with respect to the *Voronoi Tessellation*.

A *Delaunay triangulation Dt(P)* for a set *P* of sites in the plane is a triangulation, i.e. a partition of the plane into a set of triangles, such that the circumcircle of any triangle in *Dt(P)* is empty, i.e. it does not include any other point in *P*.

Given a set of *n* sites $S = \{s_1, s_2, ..., s_n\}$ of the plane, the *Delaunay triangulation* is the *dual structure* of the Voronoi diagram, where the sites correspond to the vertexes of the triangles, and an edge of a triangle connects two vertexes s_1, s_2 if and only if $V(s_1)$ and $V(s_2)$ share a common edge, i.e. s_1 and s_2 are Voronoi neighbours.

Figure 2 shows a *Delaunay Triangulation* on the top of s *Voronoi diagram*, where the borders of the Voronoi regions are shown by dotted lines and the corresponding *Delaunay Triangulation* links are shown by continuous lines.

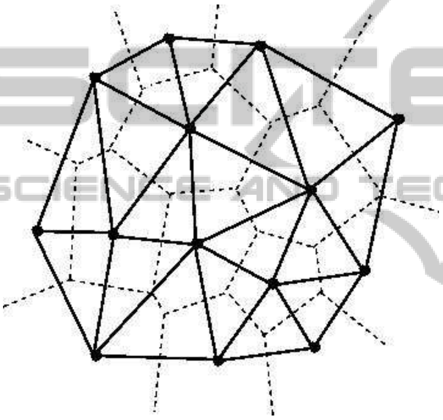

Figure 2: A Delaunay triangulation on top of a Voronoi diagram.

4 VORONOI-BASED MMOG

In a *Voronoi based* approach, the position of each peer in the virtual world is exploited to define a *Voronoi tessellation* of the virtual world. Given *n sites* corresponding to the peers, a *Voronoi tessellation* partitions the virtual world into *n areas* such that the area corresponding to a site *n* includes all the points which *are closer* to *n* with respect to any other site. Two sites are *Voronoi neighbours* iff the borders of their areas overlap. The connected graph defined by linking neighbour sites corresponds to the *Delaunay Triangulation* associated to the *Voronoi tessellation*. A *P2P* overlay is defined by connecting peers whose sites are Voronoi neighbours. In the following, the links of this overlay will be referred as *Voronoi links*.

. The adoption of this solution presents relevant advantages:

- *Mapping of Entities to the Peers*: a straightforward *mapping of entities to the peers* assigns each entity to the peer which manages the Voronoi region where the entity is located.
- *Bandwidth Saving*: since each site of a Voronoi tessellation has on the average 6 neighbours (Aurenhammer, 1991), each peer manages a bounded number of connections with other peers, i.e. those corresponding to the Delaunay links.
- *Overlay Connectivity*: the connections corresponding to the Delaunay links guarantee that the overlay is connected. Even if a peer is located in an uninhabited region of the virtual world, it remains connected with the rest of the DVE through the Delaunay connections.
- *Existence of Routing Algorithms for Delaunay Networks*: compass routing is based on a fast-to-compute angle argument which exploits the mathematical properties of geometric networks and has been proved to be cycle free for Delaunay networks. The algorithm can be exploited to define efficient AOI-cast mechanisms.

5 HyVVE: THE ARCHITECTURE

Even if the definition of a pure P2P network for *MMOG* is a challenging alternative to the classical *client/server* solution, several problems should still be solved for the definition of a comprehensive solution. One of the main problems still to be solved concerns the management of the *MMOG* state when the number of peers is very low or zero. The main problem when a low number of peers belong to the *MMOG* is related to the high load assigned to each peer. A further problem is the maintenance of the state of the *MMOG* when all the peers have left it, because this state should be restored later when some peer joins the *MMOG*.

To manage the problem of state persistence, we propose *HyVVE*, an *hybrid architecture* including a small number of ”classical” servers controlling the state of the *MMOG* and a huge amount of interacting peers. Note that this solution differs from a solution based on the definition of Super-Peer based architecture, because the set of servers is *statically defined*, while the Super-Peers are dynamically elected, they participate to the *MMOG* as normal peers and support the further task of routing the event notification for the peers they manage. In HyVVE, the server controls the join of peers to the *MMOG*, their authentication and manages a portion of the *MMOG* state. For the sake of simplicity, we consider a system where a single server *S* is defined. The server is a supervisor which does not belong to the P2P overlay and is connected to all the peers. When a peer enters the *MMOG* or updates its position, it notifies this event to the server so that the server continuously has a vision of the whole *MMOG* and is able to compute a Voronoi tessellation including all the peers of the *MMOG*. In this way, it is able to exploit the tessellation to distribute portions of the state of the *MMOG* to the joining peers. On the other way round, *HyVVE* differs from client/server architectures because a distributed protocol is exploited to exchange events like positional and object updates directly between the peers, without any intervention by the server. This avoid that the server becomes, like in classical client/server solutions, a bottleneck for the entire system. The server is involved at a peer bootstrap and then it receives the positional updates of the peer, but these are not forwarded to other peers, instead they are exploited by the server to decide if it must give up entities to the peers or acquire entities from them. To implement the distributed protocol supporting events exchange, in *HyVVE* the peers are connected by a Delaunay overlay which is built and maintained through *GoDel*, a gossip-based protocol proposed in (Baraglia et al., 2012). The description of this protocol is out of the scope of this paper, nevertheless it is worth noticing that it is a light-weighted protocol which maintains the overlay by a straightforward application of the *equiangular property* of the Delaunay triangulations (Aurenhammer, 1991).

Let us consider now the *Voronoi-based tessellation* enabling a natural mapping of the entities of the *MMOG* to the peers. Each entity is mapped to the peer whose Voronoi region includes it so that each entity is managed by a single peer of the *MMOG*. The problem of this approach is that, when the *MMOG* includes a small number of peers, a large number of entities may be associated to a single peer which may be not able to manage all them. As a matter of fact, when a few peers are present in the *MMOG* all entities are partitioned between them and a single peer may *result overloaded*. To avoid peer overloading, we associate with each peer a new circular area, the *CCAOI*, *Centered Coordination AOI*, whose center is the position of the peer and whose radius is larger than of the *AOI*. The goal of the *CCAOI* is to reduce the number of entities assigned to a peer when its Voronoi Area is too large. As a matter of fact, in this solution each peer manages the entities located inside the *Intersection Area*, *IA*, i.e. the area corresponding to the *intersection* between *Voronoi Area* and its *CCAOI*, while the entities located outside the *Intersection Area* of any peer are assigned to the server. Both these areas change dynamically when the peer moves like its *AOI*.

As Figure 3 shows, the area managed by the peer is only the blue one while the yellow area does not belong to any *Interaction Area* and the entities located in this area are managed by the server.

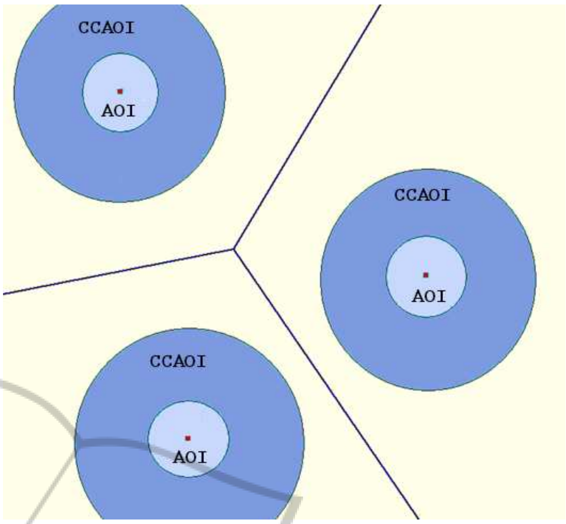

Figure 3: AOI and CCAOI.

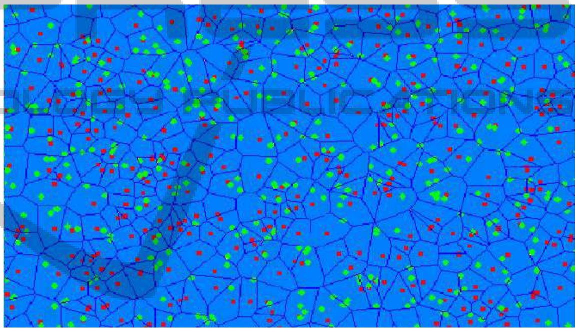

Figure 4: Each Entity is managed by a peer.

The server initially owns the state of the whole *MMOG*. When a peer joins the *DVE*, it first contacts *S* for the *authentication*, then it receives from *S*, and from its Delaunay neighbours the set of entities belonging to its Intersection Area.

It is important to note that when the the *CCAOI* is totally included in the Voronoi area of the peer, the *IA* overlaps the *CCAOI*. On the other way round, if the *CCAOI* is a super-set of the Voronoi Area, the *IA* overlaps the Voronoi Area of the peer. In a third scenario the *IA* is the portion of the *CCAOI* overlapping the Voronoi area of the peer.

The first case corresponds to a scenario where a very small number of peers are present in the *MMOG*. Note that in this case the *Voronoi Area* is much larger than the *CCAOI*. In this scenario, the introduction of the *Interaction Area* enables each peer to take care only of the coordination of the closer entities, i.e. the entities located inside its *IA* while the server manage the entities located within its Voronoi Area, but not belonging to its Intersection Area, i.e. the entities located far from it.

Note that in this scenario the server does not become a bottleneck for the system, even if a large number of entities are mapped to it because of the presence of a few peers. As a matter of fact, the probability that the peers update the entities mapped to the regions managed by the server, i.e. the yellow regions in Fig. 3, is low, because these entities are located far away the peers. Note that as the number of peers decreases, the entities are assigned back to the server that, in a natural way, acquires the total control of the system, when the last peer leaves the *MMOG*. In this case the server acts as a *backup server* for the *MMOG* state, and when each peer exits the *MMOG*, the state of the *MMOG* will be stored by the server to be restored later. Furthermore, in this way the load of the peers is reduced.

When the number of peers increases, the number of entities owned by the server decreases, because it delegates the management of the entities to the joining peers. In this scenario, as in a crowding situation, the *Interaction Area* of each peer may overlap its Voronoi region and the management of the state of the *MMOG* may be delegated entirely to the peers, as shown in Figure 4, where the management of the *MMOG* is totally delegated to the peers and the server owns no entity. In this scenario the only task of the server is to control and authenticate the peer joining the network since all the entities are managed by the peers.

Consider now a crowding scenario, for instance one where peers fight against each other and a large number of peers is concentrated in a small portion of the virtual space, as showed in Figure 5. In this case, the Voronoi Area of each peer is included in its *CCAOI*, hence the *IA* of the peer overlaps its Voronoi Area and, despite the large number of peers, the area that the server must manage is very large. Even in this situation, the server does not become a bottleneck, because it does not receive updates for the entities it owns since they are located far away from the peers. Again its task is to store the state of the entities and to decrease the load of the peer.

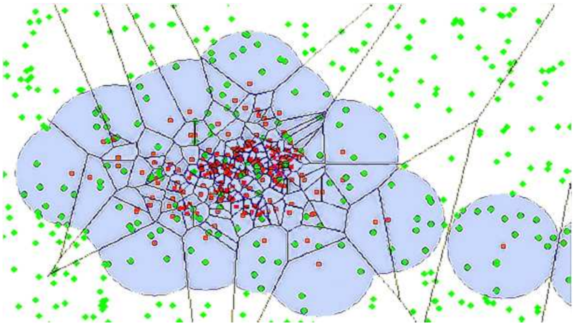

Figure 5: A Crowding Scenario.

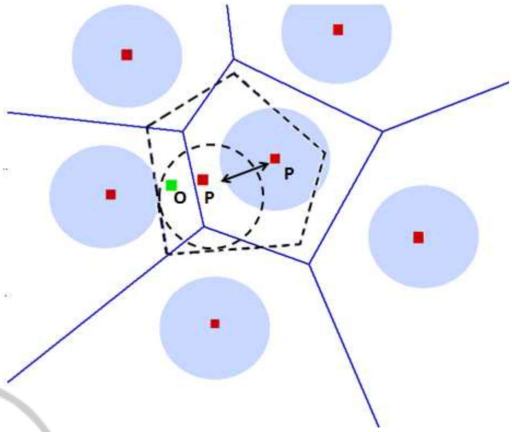

Figure 6: Entity delegation between peer and the server.

In our solution, the server itself becomes, compared to the classical client/server model, both a *backup* and a *load distribution* mechanism.

5.1 Entities Delegation

When a new peer enters the *MMOG* or the overlay is modified due to the movement of the peers, the server checks if some entity it owns falls within the IA of a peer and, in this case, it sends the entity to this peer.

The peers instead have a limited knowledge of the *MMOG* because of the local information obtained by the direct connections with peers that fall in their *AOI* and with their Delaunay neighbours. When a peer P receives information from a neighbour V about either a position update or a new neighbour notification, P updates its local Delaunay Triangulation with the new neighbours positions and check whether any of the entities owned fall in the IA of its neighbours or in the server area. In both cases, P is no longer the owner of the entity and sends the entity to the new owner.

For instance, in Figure 6, the peer P moves from left to right and the entity O, first included in the $IA(P)$ managed by P, because of the shift of P, enters the area under server competence. If we observe the movement of P, from left to right, the entity O would be assigned to the server.

In Figure 7 we see the exchange of an entity between peer P and P_1. Dotted lines show the *CCAOI* and the Voronoi region of P before its movement, when the entity O just falls in $IA(P)$. When P moves following the arrow, then O enters $IA(P_1)$ and P_1 becomes the new owner of O by receiving from P all the information.

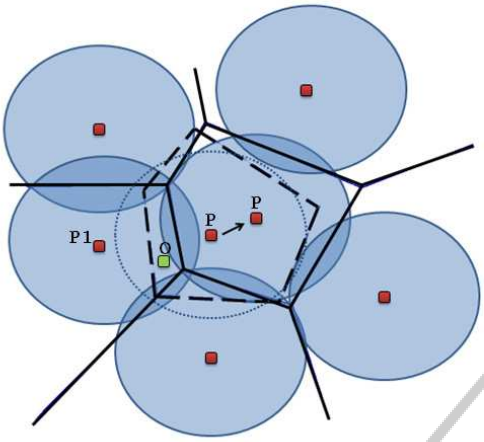

Figure 7: Entity delegation between peers.

6 EXPERIMENTAL RESULTS

This section describes a set of experiments whose goal is to evaluate the load of the server and of the peers and the number of transfers between the server and the peers for management of the entities. The experiments have been conducted by varying the number of peers, the radius of the *CCAOI* and the speed of the peers.

We have exploited PeerSim (Montresor and Jelasity, 2009), an highly scalable simulator for P2P networks. In the experiments we have considered a 2-dimensional virtual environment of size 600 × 800 and 1000 simulation cycles. At the start up of each simulation, peers are positioned at random on the map, afterwards the the peers move according to the *random way-point mobility model* (Bettstetter et al., 2004).

We consider 1000 entities uniformly positioned in the virtual world. Initially, all the entities belong to the server, then the server transfers subset of entities to the interested peers during the simulation. The entities may be then exchanged directly between the peers.

The goal of the first set of experiments is to evaluate the average number of entities owned by a peer and by the server by considering scenarios characterized by different number of peers, radius of the *CCAOI* and avatars' speed. The speed is the number of pixels covered by a single movement of a peer. We fix the number of entities to 1000, while the number of peers varies in the range $[1 \ldots 1000]$ with a step of 100. We consider two values for the *CCAOI* radius, i.e. 50 and 10 pixels, while the speed of the avatars varies in the range $[1 \ldots 2,5]$ with a step of 0.5.

Fig. 8 shows the average number of entities owned by the server, while the average number of entities owned by each peer is shown in Fig 9. Note that, when the radius of the *CCAOI* is fixed at 50 or at 10, the behaviour of the function is the same for different avatars' speeds. For this reason, the lines corresponding to different speed are completely overlapped. First of all, we observe that the server load decreases when the number of peers increases, because a larger set of peers contributes to the management of the entities. It is interesting to note that the server does not own any entity when the radius of the *CCAOI* is equal to 50 and the number of peers is larger than 300 because, in this case, the *IA* of the peers covers the whole virtual environment, no entity is managed by the server and the partition of the entities among the peers is determined by the Voronoi partition of the virtual world. Also the load of the peers decreases when the number of peers increases, but, while the reduction is remarkable when the radius of the *CCAOI* is 50, it is negligible when the radius is 10. As a matter of fact, in the latter scenario, the size of the *CCAOI* is very small and each region of the virtual world is covered by a single peer. On the other side, when the size of the *CCAOI* is larger, some portions of the virtual world are covered by a set of peers, and this implies a reduction of the average load of a peer.

Fig. 11 shows the average number of entity trans-

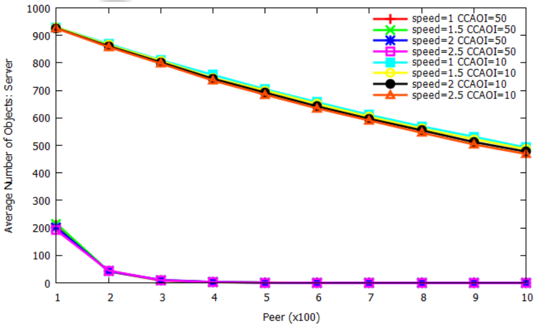

Figure 8: Average Number of Entities of the Server with 1000 objects.

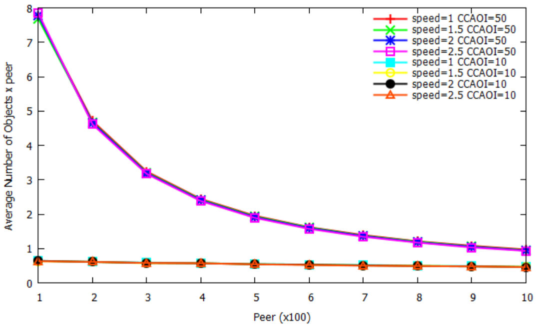

Figure 9: Average Number of Entities for each peer with 1000 objects.

fers initiated by a peer and Fig. 10 those initiated by the server, at each simulation cycle. First of all, note that the number of transfers is influenced by the avatars' speed. As a matter of fact, the difference between the *IA* of a peer before and after its movement is larger when the speed increases and this implies a larger number of entities to acquire.

The probability that the *IA* of a peer is a subset of its Voronoi area is higher when the radius of the *CCAOI* is small, even when the number of peers increases. In this case, a peer must acquire entities from the server at each movement. As a matter of fact note that the number of transfers from the server is larger when the size of the *CCAOI* is 10 with respect to 50.

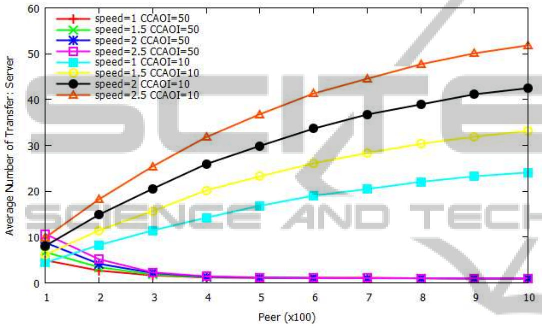

Figure 10: Average Transfers of the Server with 1000 objects.

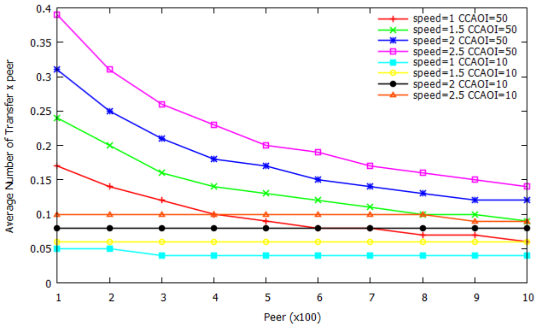

Figure 11: Average Transfer of a Peer with 1000 objects.

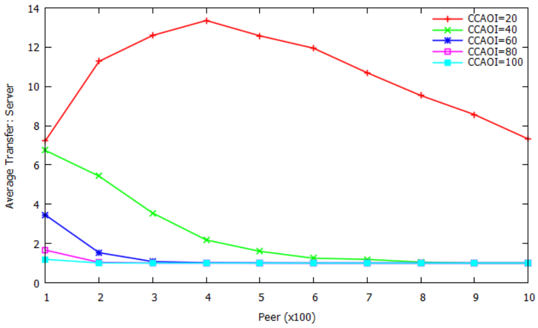

Figure 12: Average Number of Entities for the Server: Variable CCAOI.

When the radius of the *CCAOI* is 50 and the number of peers is larger than 300 no transfer between the server and the peers occurs since, as observed before, the server does not manage any entity. As far as concerns the transfers initiated by the peers, the average number of transfers for each peer does not depend from their number when the size of the *CCAOI* is small, because, as observed before, the number of entities managed by each peer is nearly constant. When the *CCAOI* is large, the average number of transfers of a peer at each cycle decreases proportionally to the number of peers, because when the number of peers is high, no entity is transferred between peers and the server.

The last set of experiments evaluates the average number of transfers initiated by the server/by the peers by considering different radius of the *CCAOI*. Note in Fig 12 that, when the radius of the *CCAOI* is equal to 20, the average transfers initiated by a the server increases until the radius of the *AOI* is 400, then it decreases. As a matter of fact, the *IA* of each peer is included in its Voronoi Area, when the number of peers is lower than 400 and the number of transfers increases with the number of peers. When the number of peer is larger than 400, the peers starts to exchange entities so that the number of transfers with the server decreases. This phenomenon occurs for smaller number of peers when the radius of the *CCAOI* is larger.

7 CONCLUSIONS

This paper presents an hybrid architecture for MMOG based on a Voronoi tessellation of the virtual world. Our approach allows a dynamical distribution of the load for the management of the entities of the virtual world among the server and the peers. We have introduced the concept of CCAOI to avoid peer overloading. The CCAOI allows to reduce the number of objects assigned to each peer. We plan to consider further mobility models to evaluate our approach in different scenarios. Further, we are considering several distributed algorithms for the management of the consistency of the entities managed by the peers.

ACKNOWLEDGMENTS

The authors thank Katia Monni for her support in developing the experiments.

REFERENCES

Albano, M., Quartulli, A., Ricci, L., and Genovali, L. (2009). Aoi cast by tolerance based compass routing in distributed virtual environments. In *Proceedings of the 8th Annual Workshop on Network and Systems Support for Games*, NetGames '09, pages 13:1–13:2.

Aurenhammer, F. (1991). Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405.

Baraglia, R., Dazzi, P., Guidi, B., and Ricci, L. (2012). Godel: Delaunay overlays in p2p networks via gossip. In *IEEE P2P, International Conference on P2P Computing*, pages 1–12.

Bettstetter, C., Hartenstein, H., and Pérez-Costa, X. (2004). Stochastic properties of the random waypoint mobility model. *Wireless Networks*, 10(5):555–567.

Bharambe, A., Pang, J., and Seshan, S. (2006). Colyseus: a distributed architecture for online multiplayer games. In *Proceedings of the 3rd conference on Networked Systems Design & Implementation - Volume 3*, NSDI'06, pages 12–12, Berkeley, CA, USA. USENIX Association.

Bonotti, A., Genovali, L., and Ricci, L. (2007). A publish subscribe support for networked multiplayer games. In *Proceedings of the Third IASTED European Conference on Internet and Multimedia Systems and Applications*, EurolMSA '07, pages 236–241.

Buyukkaya, E. and Abdallah, M. (2008). Efficient triangulation for p2p networked virtual environments. In *Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games*, NetGames '08, pages 34–39.

Carlini, E., Ricci, L., and Coppola, M. (2013). Flexible load distribution for hybrid distributed virtual environments. *Future Generation Comp. Syst.*, 29(6):1561–1572.

Cavagna, R., Abdallah, M., Buyukkaya, E., and Bouville, C. (2008). “A Framework for Scalable Virtual Worlds using Spatially Organized P2P Networks”. In *IEEE ICPADS workshop on Peer-to-Peer Network Virtual Environment, P2P-NVE*.

Cavagna, R., Abdallah, M., Buyukkaya, E., and Bouville, C. (2009). “VoroGame: A Hybrid P2P Architecture for Massively Multiplayer Games”. Ieee.

Genovali, L. and Ricci, L. (2008a). “JaDE: A JXTA Support for Distributed Virtual Environments”. In *13th IEEE Symposium on Computers and Communications Program, ISCC*, Marrakesh,Morocco.

Genovali, L. and Ricci, L. (2008b). “Voronoi Models for Distributed Environments”. In *ACM CoNEXT Student Workshop*, Madrid, Spain.

Genovali, L. and Ricci, L. (2009). Aoi-cast strategies for p2p massively multiplayer online games. In *Proceedings of the 6th IEEE Conference on Consumer Communications and Networking Conference*, CCNC'09, pages 1317–1321.

Ghaffari, M., Hariri, B., and Shirmohammadi, S. (2009). A delaunay triangulation architecture supporting churn and user mobility in mmves. In *Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video*, NOSSDAV '09, pages 61–66.

Hu, S., Chang, S., and Jiang, J. (2008). Voronoi state management for peer-to-peer massively multiplayer online games. In *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, pages 1134–1138. IEEE.

Hu, S.-Y., Chen, J.-F., and Chen, T.-H. (2006). Von: a scalable peer-to-peer network for virtual environments. *Netwrk. Mag. of Global Internetwkg.*, 20(4):22–31.

Jiang, J., Huang, Y., and Hu, S. (2008). “Scalable AOI-cast for Peer-to-Peer Networked Virtual Environments”. In *ICDCSW The 28th International Conference on Distributed Computing Systems Workshops*.

Kato, H., Eguchi, T., Ohnishi, M., and Ueshima, S. (2006). Autonomous generation of spherical p2p delaunay network for global internet applications. In *Creating, Connecting and Collaborating through Computing, 2006. C5'06. The Fourth International Conference on*, pages 184–191. IEEE.

Knutsson, B., Lu, H., Xu, W., and Hopkins, B. (2004). Peer-to-peer support for massively multiplayer games. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 1, pages 96–107. IEEE.

Lee, D. and Lam, S. (2008). “Efficient and Accurate Protocols for Distributed Delaunay Triangulation under Churn”. In *ICNP*.

Montresor, A. and Jelasity, M. (2009). PeerSim: A scalable P2P simulator. In *Proc. of the 9th Int. Conference on Peer-to-Peer (P2P'09)*.

Ohnishi, M., Nishide, R., and Ueshima, S. (2005a). Incremental construction of delaunay overlaid network for virtual collaborative space. In *Proceedings of the Third International Conference on Creating, Connecting and Collaborating through Computing*, C5 '05, pages 75–82.

Ohnishi, M., Nishide, R., and Ueshima, S. (2005b). “Incremental Construction of Delaunay Overlay Network for Virtual Collaborative Space”. In *”Third Int. Conf. on Creating, Connecting and Collaborating through Computing”*.

Ricci, L., Genovali, L., Carlini, E., and Coppola, M. (July 2011). ”AOI-cast by Compass Routing in Delaunay Based DVE Overlays”. In *The 2011 International Conference on High Performance Computing and Simulation, HPCS 2011*, Istanbul, Turkey.

Ricci, L. and Salvadori, A. (2007). “Nomad: Virtual Environments on P2P Voronoi Overlays”. In *1th Int. Work. on Peer to Peer Networks, PPN'07*, Vilamoura,Portugal.

Varvello, M., Biersack, E., and Diot, C. (2007). Dynamic clustering in delaunay-based p2p networked virtual environments. In *Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games*, NetGames '07, pages 105–110.

Yu, A. P. and Vuong, S. T. (2005). Mopar: a mobile peer-to-peer overlay architecture for interest management of massively multiplayer online games. In *Proceedings of the international workshop on Network and operating systems support for digital audio and video*, NOSSDAV '05, pages 99–104.