On the Characterization of Peer-To-Peer Distributed Virtual Environments

Category: Research

ABSTRACT

Last years have witnessed how large scale distributed virtual environments (DVEs) have become a major trend in distributed applications, mainly due to the enormous popularity of multi-player online games in the entertainment industry. Since architectures based on networked servers seems to be not scalable enough to support massively multi-player applications, peer-to-peer (P2P) architectures have been proposed as an efficient and truly scalable solution for this kind of systems. However, in order to design efficient DVEs based on peer-to-peer architectures these systems must be characterized, measuring the impact of different client behaviors on system performance.

This paper presents the experimental characterization of peer-to-peer distributed virtual environments in regard to well-known performance metrics in distributed systems. Characterization results show that system saturation is inherently avoided due to the peer-to-peer scheme, as it could be expected. Also, these results show that the saturation of a given client exclusively has an effect on the surrounding clients in the virtual world, having no effects at all on the rest of avatars. Finally, characterization results show that the response time offered to client computers greatly depends on the number of new connections that these clients have to make when new neighbors appears in the virtual world. These results can be used as the basis for an efficient design of peer-to-peer DVE systems.

Keywords: Distributed virtual environments, peer-to-peer, multi-player online games.

Index Terms: I.3.2 [Computer graphics]: Graphics systems—Distributed/network graphics; C.2.4 [Computer-communication networks]: Distributed Systems—Distributed Applications;

1 INTRODUCTION

In recent years, large scale distributed virtual environments (DVEs) have become a major trend in distributed applications, mainly due to the enormous popularity of multi-player online games in the entertainment industry. These highly interactive systems simulate a 3-D virtual world where multiple users share the same scenario. Each user is represented in the shared virtual environment by an entity called *avatar*, whose state is controlled by the user through a client computer. The system renders the images of the virtual world that each user would see if he was located at that point in the virtual environment. Hundreds and even thousands of client computers can be simultaneously connected to the DVE through different networks, and even through Internet. DVE systems are currently used in many different applications [36], such as civil and military distributed training [27], collaborative design [35] and e-learning [5]. Nevertheless, the most extended example of DVE systems are commercial, massively multi-player online game (SMOG) environments [7, 21, 33, 2, 39].

Architectures based on networked servers have been during last years the major standard for DVE systems [36, 23, 41, 22, 4, 24, 11]. In these architectures, the control of the simulation relies on several interconnected servers. Client computers are assigned to one of the servers in the system. In these architectures, when a client computer modifies the state (usually the position) of an avatar, it also sends an updating message to its server, which in turn must propagate this message to other servers and clients. Servers in the DVE system must render different 3-D models, perform positional updates of avatars, and transfer control information among different clients. Therefore, each new avatar represents an increase not only in the computational requirements of the application but also in the amount of network traffic. For example, when the number of connected clients increases, the number of update messages exchanged by avatars must be limited in order to avoid a message outburst. In this sense, concepts like areas of influence (AOI) [36], locales [3] or auras [13] have been proposed for limiting the number of neighboring avatars that a given avatar must communicate with. All these concepts define a neighborhood area for avatars, in such a way that a given client computer controlling a given avatar i must notify all the movements of i (by sending an updating message) only to the client computers that control the avatars located in the neighborhood of avatar i. The avatars in the AOI of avatar i are denoted as *neighbor avatars* of avatar i. However, in spite of these techniques networked-server architectures do not properly scale with the number of existing users, particularly for the case of MMOGs (where up to hundreds of thousands of clients can be simultaneously connected to the system) [1].

Although techniques like Frontier Sets [40] have been proposed for structured environments and small scale online games, these solutions cannot be extrapolated to massively multi-player online games. As a result, peer-to-peer architectures (P2P) seem to be the most adequate scheme in order to provide good scalability for large scale DVE systems, and several online games based on P2P architectures have been designed [29, 28, 10]. However, P2P architectures must face the awareness problem, consisting of ensuring that each avatar (for the sake of shortness, in the rest of the paper we will use the term avatar to denote the client computer controlling that avatar) is aware of all the avatars in its neighborhood [38]. Solving the awareness problem is a necessary condition to provide a consistent view of the environment to each participant. Effectively, if two neighbors avatars are not aware of such neighborhood, they will not exchange messages about their movements and/or changes, and therefore they will not have the same vision of the shared environment. Thus, providing awareness to all the avatars is a necessary condition to provide consistency (as defined in [42, 9, 34, 37]). In this sense, different strategies for providing awareness in DVE systems based on P2P architectures have been proposed [17, 19, 25, 18, 15, 10, **?**]. Some of these proposals provide full awareness an others do not, but all of them impose both additional communications between different avatars and additional computations (performed by the client computers controlling these avatars). In order to design truly efficient P2P DVE systems, the impact that these (and other) computations and communications have on real system performance must be measured.

Based on that motivation, in this paper we propose the characterization of peer-to-peer DVE systems. Since the behavior of an avatar can be determined by both the number of neighbor avatars in its surroundings and its movement rate [31], we have measured the effects that such parameters have on two well-known performance metrics for distributed systems: latency and throughput. The results show that, unlike in networked-server architectures, the system throughput increases with the number of client computers connected to the system, that is, the peer-to-peer scheme is fully scalable, as it could be expected from a fully distributed scheme. Also, the results show that the saturation of a given client exclusively has an effect on the surrounding clients in the virtual world, having no effects at all on the rest of avatars. Finally, characterization results

show that the response time offered to client computers greatly depends on the number of new connections that these clients have to make when new neighbors appears in their surroundings. These results can be used as the basis for an efficient design of peer-to-peer DVE systems.

The rest of the paper is organized as follows: Section 2 details the proposed characterization setup that allows to experimentally study the behavior of peer-to-peer DVE systems. Next, Section 3 presents the evaluation results. Finally, Section 4 presents some concluding remarks and future work to be done.

2 Characterization Setup

We propose the evaluation of P2P DVE systems by simulation, as it was done for DVE systems based on networked-server architectures [32]. The evaluation methodology used is based on the main standards for modeling collaborative virtual environments, FIPA [8], DIS [16] and HLA [20]. Since DVE systems are inherently based on networks, the metrics used for evaluating the performance of these systems include the two main metrics used for evaluating network performance. These metrics are latency and throughput [6]. Nevertheless, in order to avoid clock skewing we have selected throughput and response time (defined as the round-trip delay for the messages sent by each client) as the performance metrics to be characterized. Concretely, we have developed a simulator modeling a DVE system based on a peer-to-peer architecture. The simulator is written in C++ and it is composed of two applications, one modeling the clients and the other one modeling the central loader which the clients must initially connect with in order to join the system. Both applications used different threads for managing the different connections they must establish. Such connections are performed by means of sockets.

Each client has a main thread for managing the actions asked by the user and different threads for communicating with it s neighbor clients. For each neighbor, two threads are executed, one for listening and one for sending messages. Similarly, the central loader has two threads for communicating with each client in the system and also a main thread. It must be noted that once a client has joined the system, it is not necessary for that client to communicate with the central loader. Since the goal of this characterization is to study how the system evolves as clients move, rather than analyzing how new clients join the system, in our simulator each client is initially provided with the IP addresses of its initial neighbors.

A simulation consists of each avatar performing 100 movements. An iteration of the whole system consists of all avatars making a movement. Each avatar notifies its neighbors as well as the central loader when it reaches the 101th iteration, and then it leaves the system. We have chosen the number of 100 iterations (movements) for a simulation because it is the number of movements that the most distant avatar needs to reach the center of the square virtual world. The virtual world was a 2D square whose sides were 100 meters long. Each time an avatar moves, it sends a message to all its neighbor avatars (the client computer controlling that avatar sends a message to the client computers controlling the neighbor avatars). These destination avatars then sends back an acknowledgment to the sending avatar, in such a way that the sending avatar can compute the round-trip delay for each message send. We have denoted the average round-trip delay for all the messages sent by an avatar as the Average System Response (ASR) for that avatar (for that client computer).

Two different characterization setups have been made, depending on the metrics to be studied. In order to study the system throughput (the maximum number of clients the system can support while maintaining full awareness and reasonable latency levels), we have used a cluster of 14 nodes. One of these PCs hosted the central loader, and the rest of the 13 PCs hosted the clients in the system in a uniformly distributed way. Each node was a dual Opteron processor running Linux. Although a real system does not require communication between clients and the central loader, we have implemented a monitoring algorithm to check that the system supports a full awareness rate. This algorithm consists of each client dividing its cycle time in two phases. In the first phase, clients move following a given movement pattern (described below) and they communicate their new location in the virtual space by exchanging messages. In the second phase, each client sends the central loader a message with its new location and also which other clients it considers as its neighbors. In this way, the central loader can compute in real time if the awareness rate if 100% or not. We have used a movement rate of one client movement each 3.6 seconds. From this period, 2.1 seconds are dedicated to the first phase and 1.5 seconds are dedicate to the second phase. Finally, we used an AOI size of 10 meters.

In order to study the system latency, we used a different characterization setup. In this case, we used personal computers interconnected by a fast Ethernet network, hosting the avatars in a uniformly distributed way. The purpose of such setup was to establish an upper number for the average value of the ASR when client computers were slightly loaded. Finally, in order to study the effects that the saturation of a single client computer has on the rest of clients, we used the same setup but now one of the PCs hosted a single client, and the rest of the PCs hosted the rest of the clients in a uniformly distributed way. In these two setups we did not monitor the awareness rate, and the movement cycle time was reduced accordingly.

3 Evaluation Results

We have simulated the behavior of a a set of independent avatars in a generic DVE system based on a P2P architecture. These avatars are located within a seamless 3D virtual world [1] following three different and well-known initial distributions: uniform, skewed and clustered [22, 32]. Starting from these initial locations, in each simulation avatars can move into the scene following one of three different movement patterns: Changing Circular Pattern (CCP) [4], HP-All (HPA) [12] and HP-Near (HPN) [26]. CCP considers that all avatars in the virtual world move randomly around the virtual scene following circular trajectories. HPA considers that there exists certain "hot points" where all avatars approach sooner or later. This movement pattern is typical of multiuser games, where users must get resources (as weapons, energy, vehicles, bonus points, etc,) that are located at certain locations in the virtual world. Finally, HPN also considers these hot-points, but only avatars located within a given radius of the hot-points approach these locations. In order to illustrate these movement patterns, figure 1 shows the final distribution of avatars that a 2-D virtual world representing a square would show if these movement patterns were applied to a uniform initial distribution of avatars. For evaluation purposes, we have considered the nine possible combinations of the three initial distributions of avatars in the virtual world and the three movement patterns.

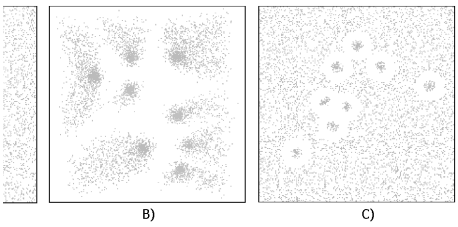

Figure 1: Final distribution of avatars for a) CCP, b) HPN, and c) HPA movement patterns applied to an initial uniform distribution of avatars.

We have performed more than 4000 experiments to study the be-

havior of DVE systems based on a P2P architecture. In any case the system provided a full awareness rate while saturation was not reached by any CPU. We have made tests with different world sizes S (the number of connected clients), although for the sake of shortness we will show a single case with 520 avatars.

Concretely, figure 2 shows a representative example of the experiments performed in order to study the system throughput. On the X-axis this figure shows the iteration number of the simulation performed, and on the Y-axis it shows the average value in seconds for the ASR of all the avatars. Each point in the plots represents the average ASR of ten different executions of the same simulation. In all of them, each cluster node hosted 40 clients.

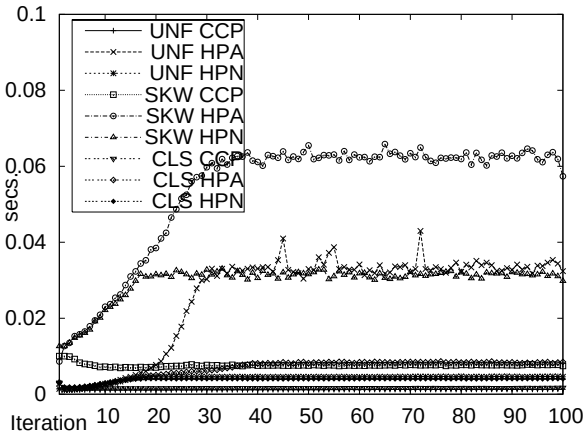

Figure 2: Average ASR values for different configurations of a DVE with 520 avatars

Figure 2 shows that the ASR remains always below the threshold of 250 ms. required to provide QoS to users [14] and also with a full awareness rate. Since in these experiments each cluster node hosted 40 avatars, we can state that there would not be any problem (the CPU utilization would be far from 100%) when executing one avatar in one PC.

Additionally, we have analyzed the system behavior in a transverse way, that is, we have studied the average ASR values for different population sizes. Although whe have performed this analysis for all the combinations of initial distributions and movement patterns, for the sake of shortness we show here a single case. All the cases showed similar results. Concretely, Figure 3 shows the average ASR values provided to avatars when following the Uniform-CCP scheme.

Figure 3 shows three flat lines at a different levels, showing that while the population increases in an order of magnitude (from 100 to 1000 avatars) the average ASR increases by a factor of two, and it is still far from the QoS threshold of 250 ms.. Therefore, these experiments prove that the peer-to-peer scheme is fully scalable, regardless of the number of client computers connected to the system. Although these results could be expected due to the inherently distributed nature of the peer-to-peer scheme, it is worth mention the remarkable differences with the results provided by the networked-server scheme [32].

Also, we have measured the average value of the ASR provided to all avatars for different movement patterns and initial distributions of avatar in the virtual worlds. Since the workload that a given avatar adds to the system depends on both the movement rate of the avatar and also on the number of neighbor avatars in the virtual world [31], we have measured the ASR provided by the system for different values of these two parameters. In these tests we used a wide range of populations sizes. However, for the sake of shortness we will show in this section the results for a system with 101

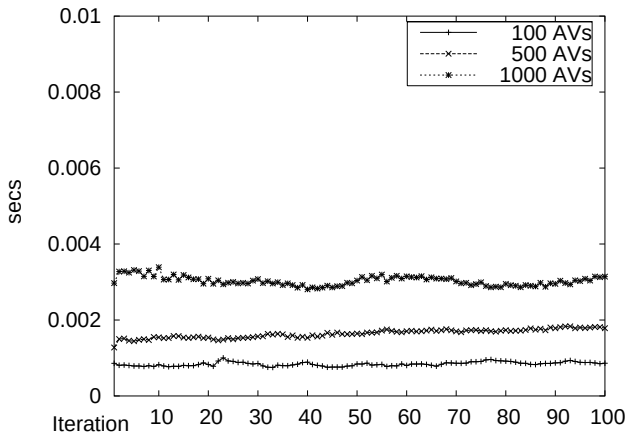

Figure 3: Average ASR values for different populations of avatars under a Uniform-CCP combination

avatars. Concretely, Figure 4 shows the results for all of the distributions with a cycle period T of 0.6 seconds (all avatars make a movement every 0.6 seconds) and an AOI of 10 meters, while Figure 5 shows the results for the same AOI but now with a movement rate of 0.1 seconds. Then, Figure 6 and Figure 7 show the results for an AOI of 20 meters and a movement rate of 0.6 and 0.1 seconds respectively. All these figures show on the X-axis the iteration number (a simulation is composed of 100 iterations), and it shows on the Y-axis the average value of the ASR provided to all the avatars.

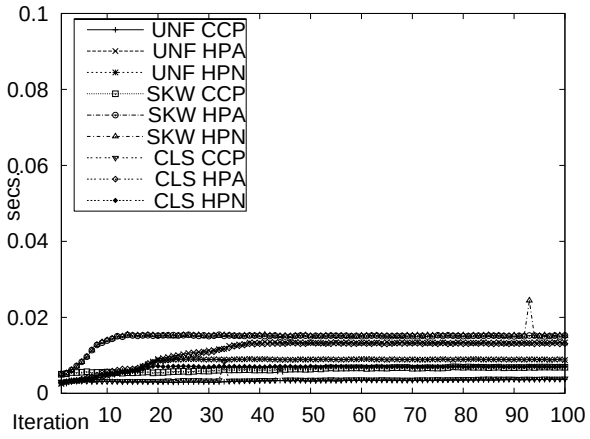

Figure 4: Average ASR values for T=0.6 s.& AOI=10 m.

All the plots in figure 4 are flat lines of different ASR values, showing that the system is below its saturation point and the average ASR does not indefinitely increase. Moreover the plots for all the movement patterns are below 0.02 seconds, far away from the QoS threshold of 0.25 seconds. Figure 5 shows the results for the same number of avatars when the cycle period is reduced to 0.1 seconds (the movement rate of all avatars is increased). In this case, the three plots corresponding to the HPA movement pattern and the plot for the HPN pattern with a skewed initial distribution of avatars show a linear increasing with the number of iterations. That is, for these patterns the client computers are not capable of sending, returning and processing the number of messages generated by avatars when they move in a period cycle. This is due to the fact that in the

HPN and particularly in the HPA pattern, avatars tend to crowd the hot points, increasing the number of neighbors in the AOI in the subsequent iterations. Since they must send more messages in the same cycle period, the client computers become saturated and the average ASR increases.

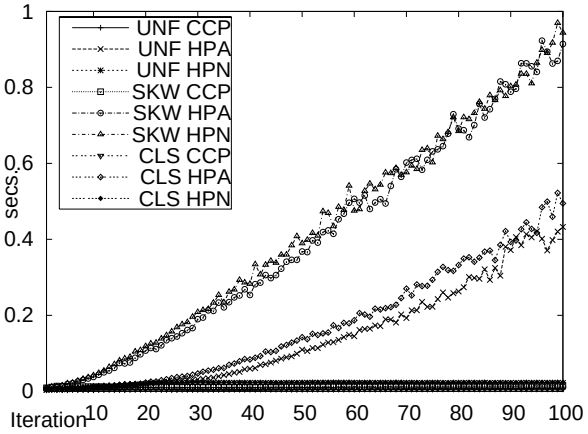

Figure 5: Average ASR values for T=0.1 s.& AOI=10 m.

Figures 6 and 7 show the results for the same population of avatars and the same patterns when the AOI size is doubled. The results shown in these figures are similar to those shown in figures 4 and 5. Figure 6 show all the plots except one with a flat slope. Although the average ASR values are slightly higher than the ones shown in 4, all of them are far away from 0.25 seconds.

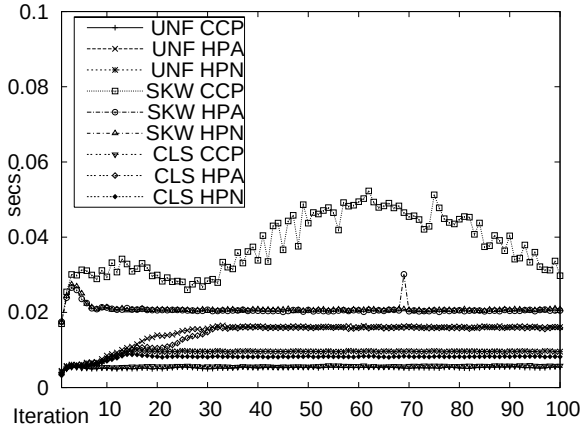

Figure 6: Average ASR values for T=0.6 s. & AOI=20 m.

Figure 7 shows that for five combinations of movement patterns and initial distributions of avatars the average ASR values linearly increase with the iteration number. This behavior indicates that the client computer cannot process all the messages sent and received at that rate. These results show that increasing the AOI size. When comparing this figure with figure 5 the only significant difference is that in the former one there are five plots with a nonzero slope, while in the latter one there are four plots.

Additionally, figures 6 and 7 show an unexpected behavior. Effectively, it can be seen in both figures that the plot with the highest ASR values is the CCP movement pattern with a skewed initial distribution of avatars in the virtual world (SKW CCP). This is a

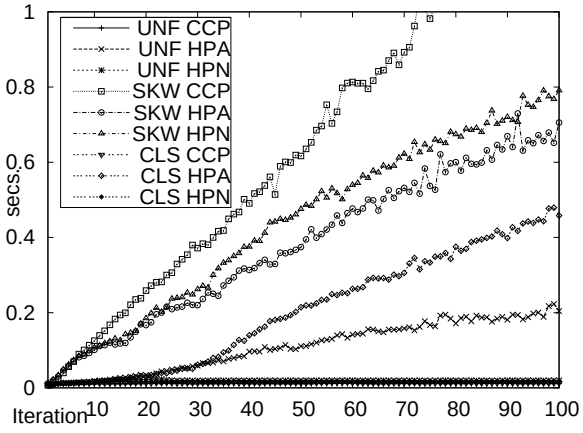

Figure 7: Average ASR values for T=0.1 s. & AOI=20 m.

strange behavior, because the CCP pattern describes a circular pattern and therefore the number of avatars within the AOIs of avatars does not increase as simulation proceeds.

In order to find the reasons for this strange behavior, we have studied the average percentage of messages sent by any avatar in each iteration, in respect with the total number of avatars in the system S. Figures 8 and 9 show the evolution of this percentage as simulations proceed for an AOI of 20 m. and for the two rates of movements (0.6 and 0.1 seconds).

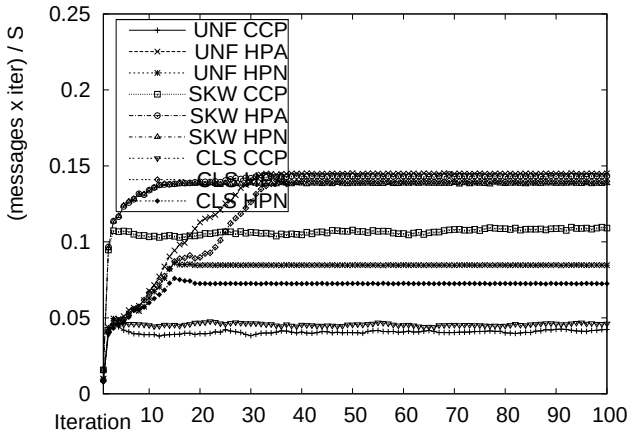

Figure 8: Average percentages of messages sent in each iteration for AOI=20 m. and T=0.6 s.

These figures show that the percentage of messages sent in the combination of the CCP pattern and the skewed initial distribution of avatars is far from being the highest one. Moreover, figure 9 shows that for a cycle period of 0.1 seconds this combination provides one of the lowest percentage. Therefore, these figures prove that the behavior shown in figures 6 and 7 is not due to the amount of new messages generated in each iteration.

After discarding the number of messages as the cause of the strange behavior, we studied the number of neighbors. Thus, Figures 10 and 11 show the average percentage of neighbors of any avatar in each iteration, with respect to the total number of avatars in the system S.

As it could be expected, the plots in figures 10 and 11 show similar results to those shown in figures 8 and 9. That is, the percent-

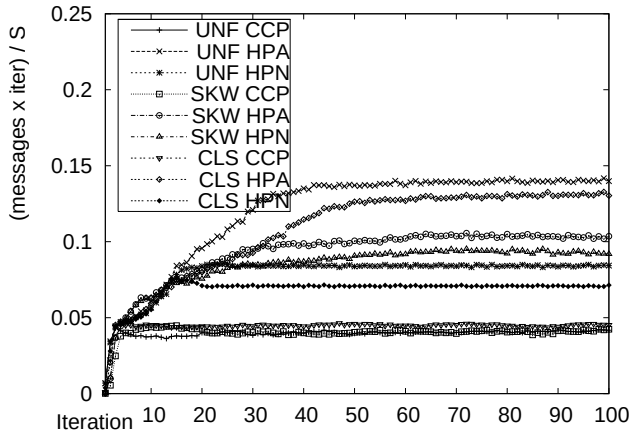

Figure 9: Average percentages of messages sent in each iteration for AOI=20 m. and T=0.1

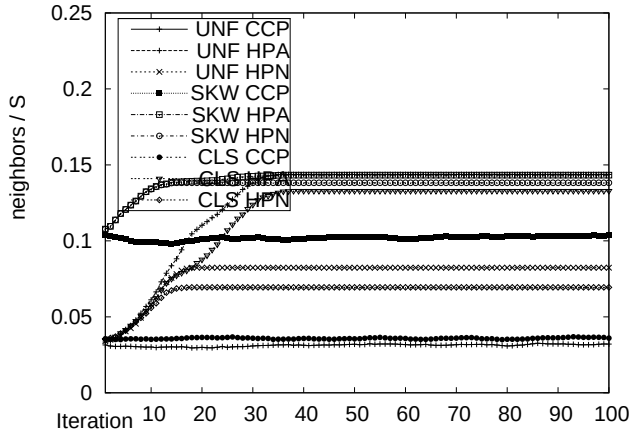

Figure 10: Average percentages of neighbors for AOI=20 m. and T=0.6 s.

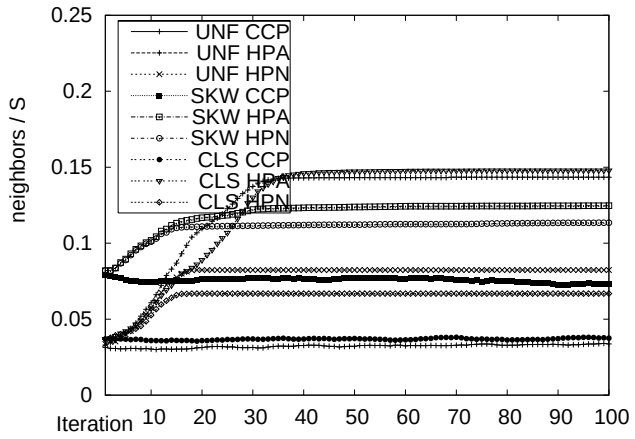

Figure 11: Average percentages of neighbors for AOI=20 m. and T=0.1 s.

ages of neighbors in each iteration for the SKW CCP combination are not the higher ones, and are clearly overcome by other combinations like SKW HPA. Therefore, the behavior shown in figures 6 and 7 is neither related to the percentage of neighbors that avatars have in each iteration.

Finally, we studied the number of new connections started by avatars in each iteration. Figures 12 and 13 show the average percentage of connections started by avatars (in an accumulative measurement), with respect to the total number of avatars in the system *S*. Figure 12 shows that the percentage of connections made in the SKW CCP combination are the higher ones. The shape of this plot has three segments, the first one with a flat slope, then a linear segment with a high slope, and finally a third linear segment with a low slope. This shape could explain the shape of the SKW CCP plot shown in figure 6.

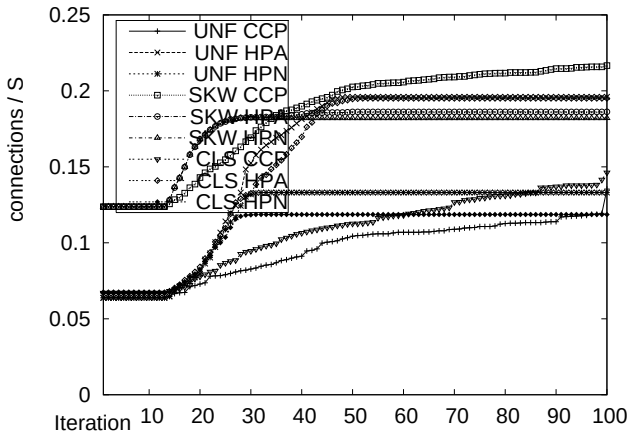

Figure 12: Average fraction of new connections for AOI=20m and T=0.6

Figure 13 shows that the SKW CCP plot is one of the combinations with the highest number of new connections per iteration, and therefore these results can explain the behavior shown in figure 6 and 7. Effectively, the CCP is a circular movement pattern. When this pattern is applied on a skewed distribution of avatars, in each iteration there are a lot of avatars that enter and leave the AOI of each avatar. This fact happens for a lot of avatars, forcing the corresponding client computers to establish new connections and also to finish some of the current ones. The overhead required to perform these actions makes the client computers to enter saturation, the connections cannot be established immediately and therefore the average ASR increases.

As a result, we can state that not only the number of current neighbors, but also the number of new connections made along time by avatars affects the ASR provided to other avatars, particularly if the client computers are close to saturation.

Additionally, we have studied the effects that a slow client computer or network link(s) can have on the rest of client computers. For this study, we have tested the 9 different combinations of initial distributions of avatars and movement patterns on a real DVE system based on P2P with 101 PCs. In these experiments, we have forced a given avatar (controlled by a single client computer) to move faster and faster, until the number of movements per second is so high that the client computer cannot send all the messages to the corresponding neighbors within the cycle period. Under these conditions, we have measured measured the latency (the ASR) provided to different avatars.

The consistency algorithm used in [30] distinguishes between different kinds of neighbors of a given avatar *i*. First, the first level neighbors (L1) are those avatars located inside the AOI of avatar *i*.

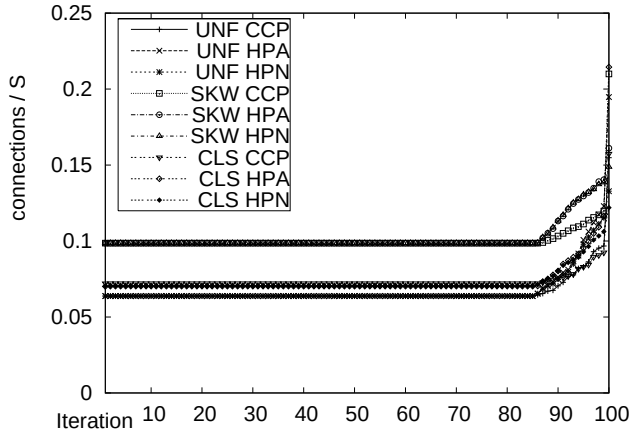

Figure 13: Average fraction of new connections for AOI=20m and T=0.1

Next, the second level neighbors (L2) are those avatars that are L1 neighbors of the L1 neighbors of *i* and are not inside AOI of avatar *i*. that is, the second level neighbors are the neighbors of the neighbors of *i*. Therefore, we have computed the average ASR value provided to *i* , the average ASR value provided to its L1 neighbors , the average ASR value provided to its L2 neighbors and the the average ASR value provided to rest of the avatars in the system.

For the sake of shortness, we will show the results for DVEs with 101 avatars and AOI of 10 meters. Figure 14 shows the results for the Uniform distribution with a movement rate of 2.1 seconds following the CCP movement pattern. In this and the next figures, we have labeled the ASR value provided to avatar *i* as AV0. The average ASR values provided to the L1 neighbors of *i* have been labeled as L1N, the average ASR values provided to the L2 neighbors of *i* have been labeled as L2N, and the average ASR values provided to the rest of avatars have been labeled as NTN.

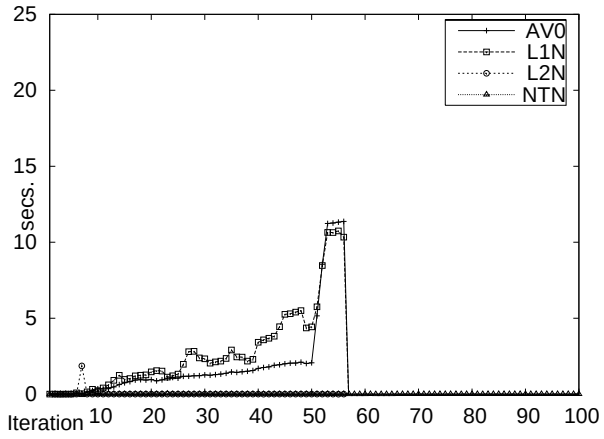

Figure 14: Effects of a client saturation on different kinds of avatars under a Uniform-CCP combination

Figure 14 shows that for the combination of uniform distribution of avatars and a CCP movement pattern the saturation of AV0 mainly has an effect on its L1 neighbors, and it has no significant effects neither on its L2 neighbors nor on the rest of avatars.

Figure 15 shows the results for the combination of a uniform distribution of avatars and an HPAll movement pattern. In this case, while the plot for AV0 has a parabolic shape, the shape of the plot for L1 neighbors is linear with a low slope. The plot for the L2 neighbors is similar to the plot for L1 neighbors, except that from the 70th iteration up it becomes negligible. This can indicate that this behavior The plot for the rest of avatars is a flat line with a negligible value. These plots indicate that the effects of the saturation of AV0 (the client computer controlling AV0) are one order of magnitude lower in its L1 and L2 neighbors, while they are negligible in the rest of avatars.

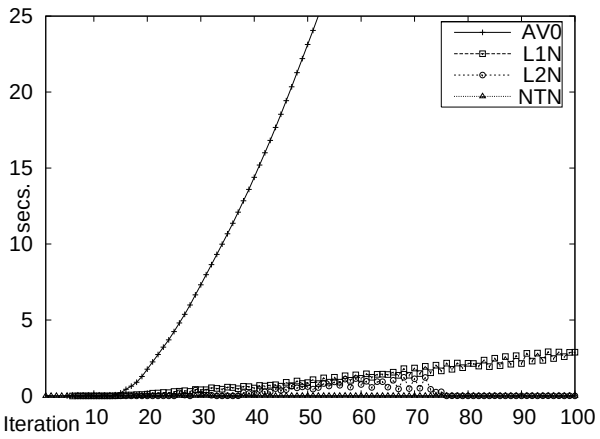

Figure 15: Effects of a client saturation on different kinds of avatars under a Uniform-HPALL combination

Figure 16 shows the results for a uniform initial distribution of avatars following an HPNear movement pattern. In this case, the shape of the plot for AV0 is linear with a high slope, while the shape of the plot for L1 neighbors is linear with a low slope and the plots for the L2 neighbors and the rest of the avatars are flat lines. That is, the difference between the HPA and HPN movement patterns makes that in the latter case the L2 neighbors are not affected by the saturation of a given avatar.

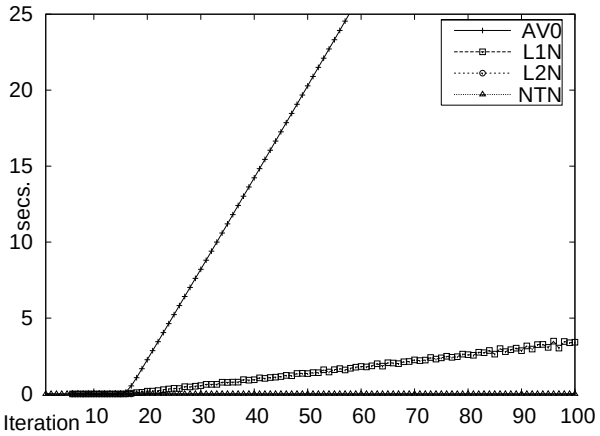

Figure 16: Effects of a client saturation on different kinds of avatars under a Uniform-HPNear combination

Figures 17 and 18 shows the results for the skewed initial distribution of avatars and CCP and HPAll movement patterns, respectively. The results for the HPNear movement pattern were very similar to the ones for HPAll movement pattern, and we are not including them for the shake of shortness. The results in these two figures are similar to those in figures 15 and 16, respectively. That is, the saturation of a given client has a significant effect on the L1

neighbors, and this effect is one order of magnitude lower than the ASR provided to the saturated avatar if an HPN movement pattern is used. On the contrary, if a HPAll movement pattern is used then the L2 neighbors are also affected by the saturation.

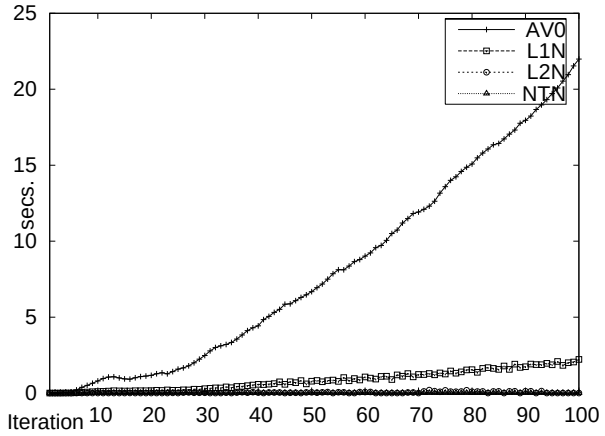

Figure 17: Effects of a client saturation on different kinds of avatars under a Skewed-CCP combination

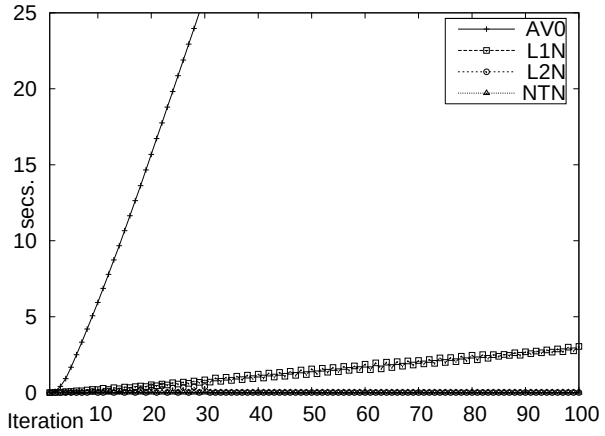

Figure 18: Effects of a client saturation on different kinds of avatars under a Skewed-HPAll combination

Finally, figures 19 and 20 shows the results for the Clustered distribution and CCP and HPAll movement patterns, respectively. Again, the results for HPNear movement were very similar to the ones for HPAll movement and we are not including them for the shake of shortness. Figure 19 is similar to figure 14, in the sense that the average ASR values for the L1 neighbors are greater than the values for AV0 itself. The values for both the L2 neighbors and the rest of avatars remain unaffected. On other hand, the results shown in figure 20 are very similar to those in figure 15. That is, the gradual grouping of all avatars in the HPN and HPAll patterns makes that the saturation of a given avatar reaches the L2 neighbors, regardless of the initial distribution of avatars.

Therefore, with all these results we can state that the effects of the saturation of a given avatar are limited to L1 and L2 neighbors. The rest of avatars are not affected, as we can expect. The reason for the behavior shown in figures 14 and 19 is the circular movement of CCP pattern, that propagates and increases the high ASR values of the L1 neighbors to other L1 neighbors as the simulation proceeds.

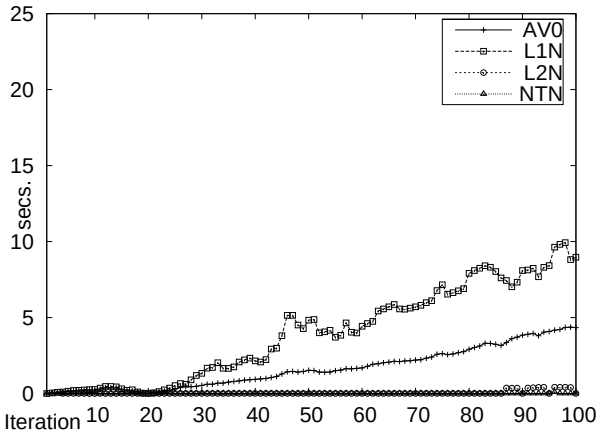

Figure 19: Effects of a client saturation on different kinds of avatars under a Clustered-CCP combination

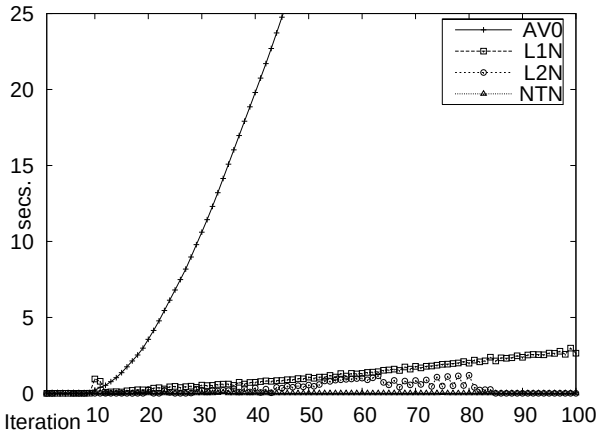

Figure 20: Effects of a client saturation on different kinds of avatars under a Clustered-HPAll combination

4 CONCLUSIONS

In this paper, we have proposed the experimental characterization of peer-to-peer distributed virtual environments, in order to set up the basis for an efficient design of this kind of DVE systems. The characterization results show that system saturation is inherently avoided due to the peer-to-peer scheme, as it could be expected. Also, characterization results show that the saturation of a given client exclusively has an effect on the surrounding clients in the virtual world, having no effects at all on the rest of avatars. This result can be crucial for designing an awareness technique or to achieve an acceptable time-space consistency.

Finally, characterization results show that the response time offered to client computers greatly depends on the number of new connections that these clients have to make when new neighbors appears in their areas of interest. That is, the saturation of an avatar (of the client computer controlling that avatar) can be produced by the sudden appearance (or disappearance) of several neighbors in its AOI. These results can be used as the basis for an efficient design of peer-to-peer DVE systems.

REFERENCES

[1] T. Alexander. *Massively Multiplayer Game Development II*. Charles River Media, 2005.

[2] Anarchy Online: : http://www.anarchy-online.com.

[3] D. Anderson, J. Barrus, and J. Howard. Building multi-user interactive multimedia environments at merl. *IEEE Multimedia*, 2(4), 1995.

[4] N. Beatrice, S. Antonio, L. Rynson, and L. Frederick. A multiserver architecture for distributed virtual walkthrough. In *Proceedings of ACM VRST'02*, pages 163–170, 2002.

[5] C. Bouras, D. Fotakis, and A. Philopoulos. A distributed virtual learning centre in cyberspace. In *Proc. of Int. Conf. on Virtual Systems and Multimedia (VSMM'98)*, November 1998.

[6] J. Duato, S. Yalamanchili, and L. Ni. *Interconnection Networks: An Engineering Approach*. IEEE Computer Society Press, 1997.

[7] Everquest: http://everquest.station.sony.com/.

[8] FIPA. Fipa agent management specification, 2000. Available at http://www.fipa.org/specs/fipa00023/.

[9] R. M. Fujimoto and R. Weatherly. Time management in the dod high level architecture. In *Proceedings tenth Workshop on Parallel and Distributed Simulation*, pages 60–67, 1996.

[10] L. Gautier and C. Diot. Design and evaluation of mimaze, a multiplayer game on the internet. In *Proceedings of IEEE Multimedia Systems Conference*, 1998.

[11] C. Greenhalgh, A. Bullock, E. Frïon, D. Llyod, and A. Steed. Making networked virtual environments work. *Presence: Teleoperators and Virtual Environments*, 10(2):142–159, 2001.

[12] F. C. Greenhalgh. Analysing movement and world transitions in virtual reality tele-conferencing. In *Proceedings of 5th European Conference on Computer Supported Cooperative Work (ECSCW'97)*, 1997.

[13] F. C. Greenhlagh. Awareness-based communication management in massive systems. *Distributed Systems Engineering*, 5, 1998.

[14] T. Henderson and S. Bhatti. Networked games: a qos-sensitive application for qos-insensitive users? In *Proceedings of the ACM SIGCOMM 2003*, pages 141–147. ACM Press / ACM SIGCOMM, 2003.

[15] S. Y. Hu and G. M. Liao. Scalable peer-to-peer networked virtual environment. In *Proceedings ACM SIGCOMM 2004 workshops on NetGames '04*, pages 129–133, 2004.

[16] IEEE. *1278.1 IEEE Standard for Distributed Interactive Simulation-Application Protocols (ANSI)*, 1997.

[17] Y. Kawahara, T. Aoyama, and H. Morikawa. A peer-to-peer message exchange scheme for large scale networked virtual environments. *Telecommunication Systems*, 25(3):353–370, 2004.

[18] J. Keller and G. Simon. Solipsis: A massively multi-participant virtual world. In *Proceedings of Parallel and Distributed Processing Techniques and Applications (PDPTA)*, pages 262–268, Las Vegas, USA, 2003.

[19] B. Knutsson, H. Lu, W. Xu, and B. Hopkins. Peer-to-peer support for massively multiplayer games. In *Proceedings of IEEE InfoCom'04*, 2004.

[20] F. Kuhl, R. Weatherly, and J. Dahmann. *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Prentice-Hall PTR, 1999.

[21] Lineage: http://www.lineage.com.

[22] J. C. Lui and M. Chan. An efficient partitioning algorithm for distributed virtual environment systems. *IEEE Trans. Parallel and Distributed Systems*, 13, 2002.

[23] J. C. Lui, M. Chan, and K. Oldfield. Dynamic partitioning for a distributed virtual environment. Technical report, Department of Computer Science. Chinese University of Hong Kong, 1998.

[24] M. R. Macedonia. A taxonomy for networked virtual environments. *IEEE Multimedia*, 4(1):48–56, 1997.

[25] M. R. Macedonia, M. Zyda, D. R. Pratt, D. P. Brutzman, and P. T. Barham. Exploiting reality with multicast groups: A network architecture for large-scale virtual environments. In *Proceedings of the 1995 IEEE Virtual Reality Annual Symposium*, pages 2–10, 1995.

[26] M. Matijasevic, K. P. Valavanis, D. Gracanin, and I. Lovrek. Application of a multi-user distributed virtual environment framework to mobile robot teleoperation over the internet. *Machine Intelligence & Robotic Control*, 1(1):11–26, 1999.

[27] D. Miller and J. Thorpe. Simnet: The advent of simulator networking. *IEEE TPDS*, 13, 2002.

[28] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. Technical report, Technical Report HPL-2002-57, HP Laboratories, Palo Alto, 2002.

[29] S. Mooney and B. Games. *Battlezone: Official Strategy Guide*. BradyGame Publisher, 1998.

[30] P. Morillo, W. Moncho, J. M. Orduña, and J. Duato. Providing full awareness to Çdsitributed virtual environments based on peer-to-peer architectures. *Lecture Notes on Computer Science*, 4035:336–347, 2006.

[31] P. Morillo, J. M. Orduña, M. Fernández, and J. Duato. On the characterization of avatars in distributed virtual worlds. In *EUROGRAPHICS' 2003 Workshops*, pages 215–220. The Eurographics Association, 2003.

[32] P. Morillo, J. M. Orduña, M. Fernández, and J. Duato. Improving the performance of distributed virtual environment systems. *IEEE Transactions on Parallel and Distributed Systems*, 16(7):637–649, 2005.

[33] Quake: http://www.idsoftware.com/games/quake.

[34] D. Roberts and R. Wolff. Controlling consistency within collaborative virtual environments. In *Proceedings of IEEE Symposium on Distributed Simulation and Real-Time Applications (DSRT'04)*, pages 46–52, 2004.

[35] J. Salles, R. Galli, and A. C. A. et al. mworld: A multiuser 3d virtual environment. *IEEE Computer Graphics*, 17(2), 1997.

[36] S. Singhal and M. Zyda. *Networked Virtual Environments*. ACM Press, 1999.

[37] J. Smed, T. Kaukoranta, and H. Hakonen. A review on networking and multiplayer computer games. Technical report, Turku Centre for Computer Science. Tech Report 454., 2002.

[38] R. B. Smith, R. Hixon, and B. Horan. *Collaborative Virtual Environments*, chapter Supporting Flexible Roles in a Shared Space. Springer-Verlag, 2001.

[39] Startcraft: http://www.blizzard.com/starcraft.

[40] A. Steed and C. Angus. Supporting scalable peer to peer virtual environments using frontier sets. In *Proceedings of 2005 IEEE Virtual Reality*. IEEE Computer Society, 2005.

[41] P. Tam. Communication cost optimization and analysis in distributed virtual environment. Technical report, Department of Computer Science. Chinese University of Hong Kong, 1998.

[42] S. Zhou, W. Cai, B. Lee, and S. J. Turner. Time-space consistency in large-scale distributed virtual environments. *ACM Transactions on Modeling and Computer Simulation*, 14(1):31–47, 2004.