A Method for Providing QoS in Distributed Virtual Environments

Abstract

One of the key issues in the design of scalable and cost-effective Distributed Virtual Environment systems is the partitioning problem. It consists of efficiently assigning clients (3-D avatars) to the servers in the system, and some proposed methods allow to significantly increase system throughput. However, these methods are not focused on satisfying any specific time constraint.

In this paper, we show that the problem of providing quality of service in Distributed Virtual Environment systems can also be addressed by means of the partitioning problem. Also, we propose a partitioning method that not only provides a high system throughput, but it also satisfies (if possible) any time constraint that avatars can require. This method is based on a heuristic search technique that looks for the best trade-off between system latency, system throughput and partitioning efficiency. The evaluation results show that this method allows to greatly increase the number of avatars provided with quality of service while also providing the highest system throughput as possible.

I. Introduction

Distributed Virtual Environment (DVE) systems have experienced a spectacular growth over the last few years. These systems allow multiple users, working on different computers that are interconnected through different networks (and even through the Internet) to interact in a shared virtual world. Each user is represented in the shared virtual environment by an entity called *avatar*, whose state is controlled by the user. DVE systems are currently used in many different applications, such as collaborative design [1], civil and military distributed training [2], e-learning [3] or multi-player games [4], [5].

Architectures based on networked servers are becoming a de-facto standard for DVE systems [6]–[8]. In these architectures, client computers are attached exclusively to one of the servers in the system. When a client computer moves an avatar, it also sends an updating message to its server, that in turn must propagate this message to other servers and client computers. Concepts like areas of influence (AOI) [6], locales [9] or auras [10] have been proposed for limiting the number of updating messages. All these concepts defi ne a neighborhood area for avatars, in such a way that a given client computer c 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.

The *partitioning problem* [11] has been shown as the main issue in the design of efficient DVE systems based on networked servers. It consists of efficiently distributing the workload among the different servers in the system (by assigning avatars to servers). In a previous paper, we proposed a dynamic partitioning method for solving the partitioning problem [12]. This method provides a significant throughput increase to DVE systems. Nevertheless, the most important performance measures in DVE systems (as in any client-server system) are not only throughput but also latency. Latency can be defined as the time interval from the instant when any neighbor of a given avatar i makes a movement until the instant when avatar i is notified of that movement. Latency represents *Quality of Service (QoS)* provided to users by the system, since it determines how fast changes in the virtual world are notified to the proper client computers.

In this paper, we show that the problem of providing QoS to avatars in a DVE system can be addressed by means of the partitioning method. Also, we propose a partitioning method based on a meta-heuristic technique that looks for the best trade-off between system latency, system throughput, and partitioning efficiency. The evaluation results show that this method can maximize system throughput while also increasing the number of avatars provided with QoS.

The paper is organized as follows: Section II details the problems to be solved for achieving QoS in DVE systems. Section III describes the heuristic search method used for providing QoS in DVE systems. Next, Section IV presents the performance evaluation of the proposed method. Finally, Section V presents some concluding remarks and future work to be done.

II. Quality of Service in DVE systems

The problem of providing QoS to clients in a DVE system has been already described, and some strategies have been proposed for solving it ([13], [14]). One of these approaches [14] uses latency compensating methods in order to repair the effects of high network jitter. A different approach [13] consists of modifying the resolution of the 3-D models depending on the connection speed of the client computer. Both strategies try to provide QoS to avatars in exchange for reducing the quality of graphics. However, none of these strategies takes into account the non-linear behavior of DVE systems with the workload assigned to each server, as described in [15]. Therefore, they cannot guarantee that the system will not become saturated, greatly degrading latency.

We can consider that a DVE system provides QoS to a given avatar i if the latency offered to i is below a certain limit. Nevertheless, latency cannot be properly measured in distributed systems, and the round-trip delay for the messages sent by an avatar i is used instead. A recent work shows that if this round-trip delay is not greater than 250 milliseconds, then users perceive that the system responds quickly [14]. We have assumed this threshold value as the condition of providing QoS to an avatar. However, it can be modifi ed as necessary, depending on the application requirements.

The round-trip delay for messages sent by a given avatar is closely related to the partitioning problem. If a given avatar i and all its neighbors are assigned to the same server, then the time required for reporting avatar i of the movements of any neighbor avatar will be smaller than if some of that neighbors are assigned to a different server. We have evaluated different DVE systems by simulation (as described in section IV), in order to measure the effect that assigning neighboring avatars to different servers can have on the QoS provided to a given avatar. A signifi cant example of this evaluation is shown in Figure 1. This fi gure shows on the Y-axis the average round-trip delays obtained for messages sent by a given avatar i. The X-axis shows the amount of neighboring avatars of i assigned to the same server where i is assigned to. Each point in the plot represents the average value of the average latencies obtained after 30 different simulations, and the standard deviation of the different simulations was not higher than 25 ms in any case. The presence factor $f_p(i)$ (the number of avatars in whose AOI avatar i appears [16]) in these simulations was equal to 60 for all the avatars.

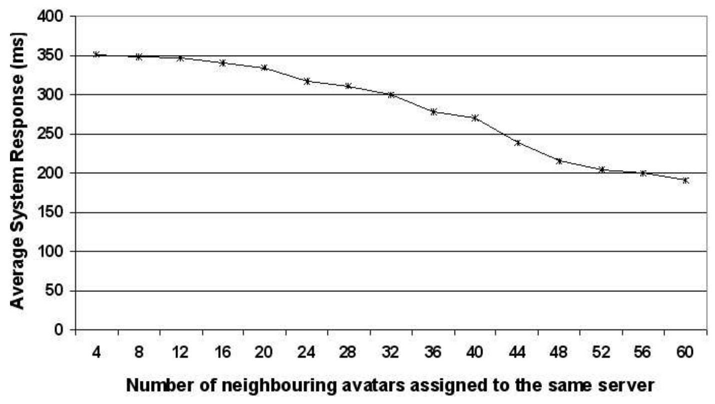

Fig. 1. Effect of assigning avatars to different servers on message latencies.

Figure 1 shows that the average round-trip delay for messages sent by avatar i linearly increases as more neighboring avatars of i are assigned to different servers. Concretely, we can see that the system provides QoS (average round-trip delay under 250ms.) to avatar i if 44 or more neighbor avatars of avatar i (from a total of 60 neighbor avatars) are assigned to the same server where i is assigned to. We repeated the same experiment for several different DVE configurations, and we obtained very similar results. The average round-trip delay for the messages sent by a given avatar increased linearly with the number of neighbor avatars assigned to a different server. Therefore, a feasible way of providing QoS to avatars is to group neighbor avatars as much as possible in the same server.

Nevertheless, grouping some avatars in a given server can unbalance the workload. As shown in [12], [15], this in turn can decrease system performance, making the system not only unable to provide QoS to some avatars, but also greatly decreasing system throughput and greatly increasing latency for all avatars. The partitioning method should guarantee that the percentage of CPU utilization in all servers is below 100%, regardless of any other consideration. Additionally, providing QoS to avatars can have an effect on the number of avatars migrated from one to another server in each execution of the partitioning algorithm, since it can be necessary to migrate a lot of avatars. In this sense, a recent work shows that any efficient partitioning method must not migrate more than 30% of avatars in the DVE system [17]. Therefore, all these three aspects must be taken into account for providing QoS when solving the partitioning problem.

III. Partitioning Method

In order to provide QoS, we propose the use of a meta-heuristic technique. Concretely, we have implemented two different techniques: an evolutionary meta-heuristic technique, called *Simulated Annealing (SA)*, and a constructive meta-heuristic technique, called *Greedy Randomized Adaptive Search (GRASP)* [18]. Since we have obtained better performance in solving this particular problem with the GRASP technique, we propose this technique as a partitioning method for providing QoS in a DVE system.

A. Quality Function

Each partition (assignment of clients to servers) of the DVE system is associated with a target function F, denoted as the *quality function*, that assigns a cost to each particular solution. The meta-heuristic technique must find a near-optimal partition with the lowest value of F as possible.

The target function F must take into account the three factors described in the previous section ion order to providing QoS to avatars: the percentage of CPU utilization in all the servers, the number of avatars that are migrated and the number of avatars that are provided with QoS. We have defined the quality function F as

$$F(P) = \Psi(P) + \Phi(P) + \Gamma(P) \tag{1}$$

where the term $\Psi(P)$ evaluates the estimated percentage of CPU utilization in all the servers for partition P. The estimation of this term is based on the characterization method proposed in [15]. The term $\Phi(P)$ is inversely related to the estimated number of avatars in partition P whose messages would show an average round-trip delay lower than 250 ms. This term is estimated taking into account that the round-trip delay for the messages sent by each avatar is related to the number of neighbor avatars assigned to the same server. Finally, the term $\Gamma(P)$ measures the number of avatars migrated from one to another server by partition P. In order to compute this term, both the current assignment and the assignment defined by partition P are used.

In order to compute $\Psi(P)$, we have defined a function $h_{cpu}(i)$ that evaluates the estimated CPU utilization in each server i for partition P. This function has the exponential behavior shown in Figure 2. If partition P results in an estimated CPU utilization significantly lower than 100% in server i, then the value of $h_{cpu}(i)$ will be very low. This value will significantly increase as server i reaches a CPU utilization close to 90%, and it will shoot up if this CPU utilization goes beyond this threshold value for partition P.

Since the purpose of $\Psi(P)$ is to provide a partition where the estimated CPU utilization is under 100% in *all* the servers, $\Psi(P)$ has been defined as

$$\Psi(P) = \sum_{i=1}^{s} h_{cpu}(i) \tag{2}$$

where s is the number of servers in the DVE system. In this way, if the CPU utilization reaches 100% in any server then $\Psi(P)$ will be very high and the partitioning method will discard partition P.

The behavior of $\Phi(P)$ is inversely related to the number of avatars whose messages show an average round-trip delay lower than 250 ms.. A function $h_{asr}(i)$ that evaluates the average

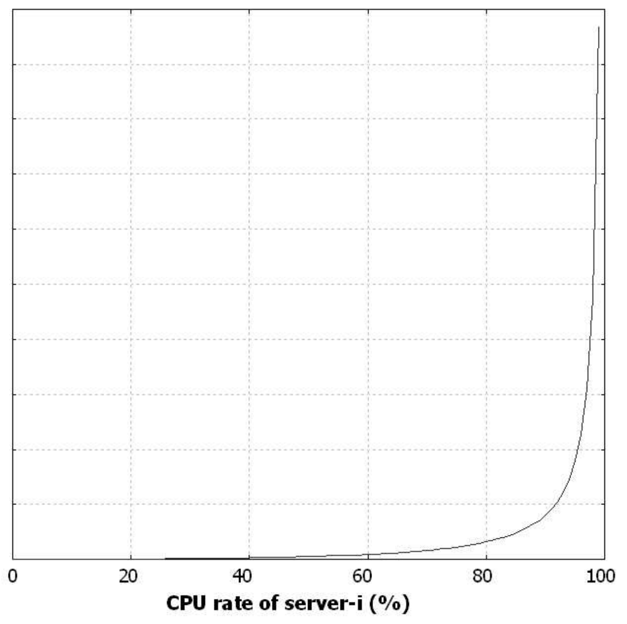

Fig. 2. Evaluation function $h_{cpu}(i)$.

round-trip delay for the messages sent by each avatar i is computed. The current average round-trip delay for each avatar is measured for the current partition, and from these values the average round-trip delay that each avatar would have in partition P is estimated, taking into account that it is linearly related to the number of neighbor avatars assigned to the same server. Then, function $h_{asr}(i)$ assigns a value to partition P for each avatar, depending on that estimated value. Function $h_{asr}(i)$ has the behavior shown in in Figure 3, and it has two sections: An inverse exponential segment from the zero value to 250 ms., and a parabolic segment from 250 ms. up.

Function $\Phi(P)$ is defined as

$$\Phi(P) = \sum_{i=1}^{n} h_{asr}(i) \tag{3}$$

$\Gamma(P)$ is also an evaluation function composed of two sections. In order to be efficient, the partitioning method should not migrate more than one third of the existing avatars [17]. Therefore, the behavior of the evaluation function $\Gamma(P)$ is the one shown in Figure 4. $\Gamma(P)$ has a linear section from the zero value to one third of the existing avatars. From that point up, $\Gamma(P)$ has a parabolic behavior. Thus, the more avatars migrates partition P, the greater the probability of discarding P is.

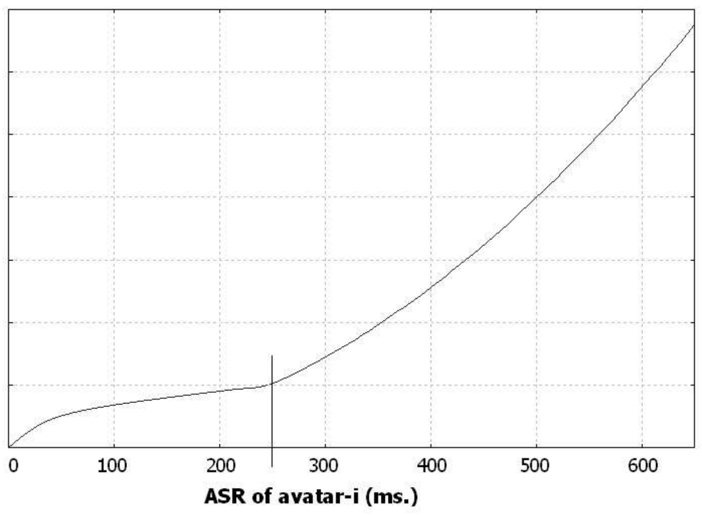

Fig. 3. Evaluation function $h_{asr}(i)$

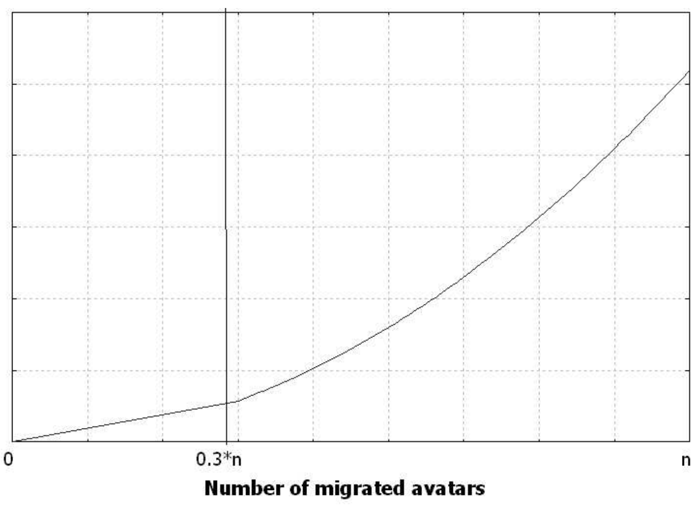

Fig. 4. Evaluation function Γ.

B. Heuristic Search Method

We propose GRASP search as the partitioning method used for providing QoS to avatars. GRASP search has been already evaluated and compared with other meta-heuristic technique in order to be applied to this specifi c problem [18], and it has provided the best results.

GRASP technique starts with an initial partition. This initial partition is provided by the load balancing technique proposed in [12]. Therefore, we ensure that the initial partition is wellbalanced and all servers have a percentage of CPU utilization as low as possible. However, some avatars in this partition may not be provided with QoS. At this point, the GRASP method will be used in order to search for a near optimal partition that provides QoS to the maximum number of avatars possible while migrating the minimum number of avatars.

The first step in our GRASP implementation consists of sorting (in descending order) the avatars whose messages show a round-trip delay higher than 250 ms. (those avatars not provided with QoS) by their presence factor f_p. The idea is to provide QoS to those avatars that require the least system efforts. The avatars with the highest presence factor should exchange updating messages with a lot of avatars. Therefore, migrating these avatars to the proper server can decrease the round trip-delay for the messages sent by the maximum number of avatars as possible (it can provide QoS to a lot of avatars with the least effort). Moreover, if the GRASP method focuses only on this kind of avatars, then it will significantly decrease the Φ function without significantly increasing the Γ function.

The first c elements in the sorted list of avatars (from a population of n avatars) are denoted as *critical avatars*. GRASP method considers that critical avatars are not assigned, and it will try to assign them to a server in such a way that they are provided with QoS. The rest of the n avatars (denoted as the e *easy avatars*, where $n = c + e$) will not be re-assigned, and they will remain assigned to the same server where they were initially assigned to. Each iteration of GRASP method assigns one of the critical avatars. The number of iterations (the number of re-assigned avatars) is the only parameter to be tuned for the GRASP method. If the c value is set too low, then only a few avatars will be provided with QoS. If the c value is set too high (trying to provide too many avatars with QoS), then GRASP method will not be able to find a partition satisfying the requirements for all avatars, and it will take a long time for providing bad partitions.

Each iteration of the GRASP method consists of two steps: construction and local search. The *construction* phase builds a feasible solution choosing one critical avatar by iteration, and the *local search* derives this temporal solution following a neighborhood criterion. The detailed implementation of GRASP search in each iteration is shown in [18]. We have not included this description here due to space limitations. When GRASP method finishes, it provides a partition with a near-optimal value of quality function F.

IV. Performance Evaluation

We propose the evaluation of generic DVE systems by simulation. The evaluation methodology used is based on the main standards for modeling collaborative virtual environments, such as FIPA [19], DIS [20] and HLA [21]. We have developed a simulation tool that models the behavior of a generic DVE system composed of several interconnected servers, and we have performed experimental studies to evaluate the performance of the proposed technique.

In each simulation, the avatars sharing the same AOI exchange messages in order to notify their position in the 3D virtual world. The message structure is the *Avatar Data Unit (ADU)* specified by DIS [20]. A simulation consists of each avatar performing 40 movements, at a rate of one movement every 2 seconds. Each time an avatar i performs a movement, the client computer controlling i sends a message with a timestamp to the client computers controlling the neighbor avatars of i. These client computers return an ACK message to the client computer controlling i. When simulation finishes, each client computer can compute the average round-trip delay for all the messages sent during the simulation. We denote this average value for a given avatar (client computer) i as asr_i (*average system response* for avatar i). Clock skewing is avoided, since the same client computer controlling i adds both the initial and the ACK timestamps.

For comparison purposes, we have simulated the partitioning method proposed in the previous section (GRASP), and the partitioning method ALB proposed in [12] (the partitioning method that provides the initial partition for GRASP method). We have simulated DVE systems with three different movement patterns of avatars: Changing Circular Pattern (CCP) [22], Hot-Points-ALL (HPA) [23] and also Hot-Point-Near (HPN) [24]. CCP considers that all avatars in the virtual world move circularly, starting and ending at the same location. HPA considers that there exists certain “hot points” where all avatars approach sooner or later. Finally, HPN also considers these hot-points, but only avatars located within a given radius of the hot-point approach these points. An iteration in a given movement pattern consists of all avatars in the system performing a single movement. Additionally, different initial distributions of avatars in the virtual world (uniform, skewed and clustered) have been considered, as in other studies [11], [22]. Figure 5 shows the final distribution of avatars that a 2-D virtual world would show if these movement patterns were applied to an uniform initial distribution of avatars. In this figure, the virtual world is a square and each avatar is represented by a grey dot.

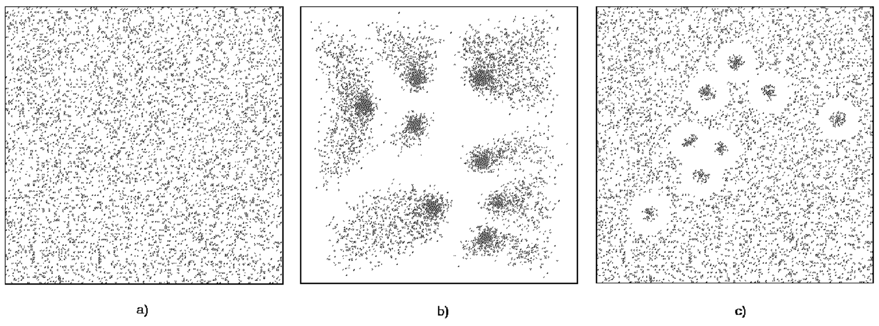

Fig. 5. Final distribution of avatars for a) CCP, b) HPA, and c) HPN movement patterns applied to an initial uniform distribution of avatars.

Both ALB method and GRASP method have been executed each time that an iteration finishes and the CPU utilization in any server reaches 95%. Additionally, the amount of avatars provided with QoS in that iteration is saved. GRASP method is executed after each subsequent iteration if this parameter decreases below the saved value, in order to increase QoS again.

Due to space limitations, we will only show here the results for the most representative combinations of movement patterns and initial distributions of avatars. The rest of the 9 possible combinations of movement patterns and initial distributions of avatars provided very similar results to those shown here for each DVE configuration.

We have tested a great number of different DVE configurations, ranging from small virtual worlds (composed of 3 servers and 250 avatars) to large virtual worlds (composed of 700 avatars and 10 servers). For the sake of shortness, we present in this paper two representative experiments, denoted as MEDIUM1 and MEDIUM2. MEDIUM1 is composed of 250 avatars and 3 servers, and MEDIUM2 is composed of 700 avatars and 10 servers. We obtained very similar results in all the experiments we performed with other DVE configurations. The tuning of the GRASP method for both MEDIUM1 and MEDIUM2 configurations resulted in an optimum number of critical avatars c of 50 and 125 for MEDIUM1 and MEDIUM2 configurations, respectively [18].

Table I shows the performance evaluation results obtained for the MEDIUM1 configuration when the partitions provided by different methods are simulated under different combinations of movement patterns and initial distributions of avatars in the virtual world. For each combination of movement pattern and initial distribution of avatars, table I contains two columns, one for each of the considered partitioning methods. The first three rows, labeled with Sx, show the maximum percentage of CPU utilization reached in each DVE server during the simulation with each partitioning method. The last but two row shows the average ASR value (in milliseconds) for the messages sent by all avatars during the simulation, and it shows the quality of service provided to avatars on the average. The next to last row shows the number of avatars whose messages showed an average round trip-delay lower than 250ms. That is, this row shows the number of avatars provided with QoS by each partitioning method. Finally, the last row shows the number of migrated avatars by each partitioning method during the simulation.

TABLE I

RESULTS FOR MEDIUM1 DVE CONFIGURATION

	HPA Uniform		**HPN Clustered**	
	ALB	GRASP	ALB	GRASP
S0 (max. %)	94	96	87	91
S1 (max. %)	97	91	99	98
S2 (max. %)	86	97	94	89
Av. ASR	219.2	167.6	247.5	189.4
QoS	168	238	170	235
Migrations	21	81	48	121

Table I shows that for the MEDIUM1 configuration both methods avoid the saturation of the DVE system, since none of the servers reaches 100% of CPU utilization at any moment. However, the average ASR values obtained with GRASP method are significantly lower than the ones obtained with ALB method for both movement patterns. In the case of the HPN pattern, the average ASR value provided by ALB method is near the limit of 250 ms., and the one provided by the GRASP method is much lower. In terms of the amount of avatars provided with QoS, the proposed method greatly increases this performance measure. The ALB method only provides QoS to around 68% of the population in both HPN/Uniform and HPA/Clustered patterns, while the proposed method is able to provide QoS to around 95% of the population of avatars for the same patterns. Finally, although the proposed approach migrates more avatars than the ALB method, it does not migrate more than 30% of the population (250 avatars) during the whole simulation. Therefore, the GRASP approach is far from migrating more than 30% of the population in any single execution of the search method.

Table II show the results obtained for a MEDIUM2 configuration. This table has the same format as Table I, except that it shows the results for ten servers. The results shown in this table correspond to the HPN movement pattern with a uniform initial distribution of avatars and also to the HPA pattern with a skewed initial distribution of avatars.

TABLE II

RESULTS FOR MEDIUM2 DVE CONFIGURATION

	HPN Uniform		**HPA Skewed**	
	ALB	GRASP	ALB	GRASP
S0 (max. %)	97	98	100	99
S1 (max. %)	98	97	98	97
S2 (max. %)	98	96	97	100
S3 (max. %)	96	97	100	100
S4 (max. %)	98	94	98	100
S5 (max. %)	97	98	100	98
S6 (max. %)	98	96	97	100
S7 (max. %)	94	98	100	98
S8 (max. %)	95	97	100	100
S9 (max. %)	96	97	98	99
Av. ASR	278.7	233.9	411.8	259.2
QoS	214	404	101	240
Migrations	56	139	103	271

Table II shows that in the case of the HPN pattern the system worked below its saturation point, and thus the maximum CPU utilization was not greater that 98% in any server. However, for the case of the HPA pattern the system worked in deep saturation, and only 26 iterations were performed before the system blocked due to the huge workload generated by avatars. For this movement pattern, all the average values are measured for these 26 iterations. As it could be expected, for this movement pattern several servers reached 100% of CPU utilization with both methods. In this way, we could measure the performance of the proposed method under different load levels.

Table II shows the greatest performance differences between the two considered partitioning methods. Effectively, the average ASR values provided by both methods signifi cantly differs for the HPN pattern, and the GRASP methods decreases this value to half the value provided by the ALB method in the case of HPA pattern. These results shows that the greater workload the system supports, the greater performance differences exist between the two considered methods. In terms of the number of avatars provided with QoS, the GRASP method also shows a signifi cant increase. The GRASP method practically doubles the number of avatars provided with QoS for both HPN and HPA patterns. These results show that the proposed method is able to increase the number of avatars provided with QoS not only when the systems supports a low load, but also when the system works in deep saturation. Finally, the number of migrations shows that none of the methods migrates more than 30% of the population. These results validate GRASP method as a partitioning method capable of providing QoS to a high number of avatars.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a partitioning method for DVE systems based on a meta-heuristic technique that looks for the best trade-off between system latency, system throughput, and effi ciency when solving the partitioning problem.

Performance evaluation results show that the proposed method can keep the system below the saturation point (if possible) and at the same time it can increase the number of avatars provided with QoS. The grouping of neighbor avatars in the same server allows to reduce the computational power required for maintaining a consistent view of the virtual world. As a result, system can accomplish faster this task, and the amount of avatars provided with QoS is signifi cantly increased.

As a future work to be done, we plan to adapt the proposed method in order to allow different time constraints for different avatars. In this way, we can provide distributed virtual environments that can support different categories of avatars. This feature can be very useful for DVE systems, particularly for some networked games.

REFERENCES

[1] J. S. Dias, R. Galli, and A. C. A. et al., "mworld: A multiuser 3d virtual environment," *IEEE Computer Graphics*, vol. 17, no. 2, 1997.

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

[3] 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)*, 1998.

[4] M. Abrash, “Quakeś game engine: The big picture,” *Dr. Dobb’s Journal*, 1997.

[5] J. Smed, T. Kaukoranta, and H. Hakonen, “A review on networking and multiplayer computer games,” Turku Centre for Computer Science. Tech Report 454., Tech. Rep., 2002.

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

[7] J. C. Lui, M. Chan, and K. Oldfi eld, “Dynamic partitioning for a distributed virtual environment,” Department of Computer Science. Chinese University of Hong Kong, Tech. Rep., 1998.

[8] P. Tam, “Communication cost optimization and analysis in distributed virtual environment,” Department of Computer Science. Chinese University of Hong Kong, Tech. Rep., 1998.

[9] D. Anderson, J. Barrus, and J. Howard, “Building multi-user interactive multimedia environments at merl,” *IEEE Multimedia*, vol. 2, no. 4, 1995.

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

[11] J. C. Lui and M. Chan, “An effi cient partitioning algorithm for distributed virtual environment systems,” *IEEE Trans. Parallel and Distributed Systems*, vol. 13, 2002.

[12] P. Morillo, J. M. Orduña, M. Fernández, and J. Duato, “An adaptive load balancing technique for distributed virtual environment systems,” in *Proc. of Intl. Conf. on Parallel and Distributed Computing and Systems (PDCS’03)*, IASTED. ACTA Press, 2003, pp. 256–261.

[13] Z. Choukair, D. Retailleau, and M. Hellstrom, “Environment for performing collaborative distributed virtual environments with qos,” in *Proceedings of the International Conference on Parallel and Distributed Systems (ICPADS’00)*. IEEE Computer Society, 2000, pp. 111–118.

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

[15] P. Morillo, J. M. Orduna, M. Fernández, and J. Duato, “On the characterization of distributed virtual environment systems,” in *Euro-Par’ 2003 - Lecture Notes in Computer Science 2790*, ACM. Springer-Verlag, 2003, pp. 1190–1198.

[16] P. Morillo, J. M. Orduña, M. Fernández, and J. Duato, “On the characterization of avatars in distributed virtual worlds,” in *EUROGRAPHICS’ 2003*. The Eurographics Association, 2003, pp. 215–220.

[17] K. Lee and D. Lee, “A scalable dynamic load distribution scheme for multi-server distributed virtual environment systems with highly-skewed user distribution,” in *Proceedings of the 10th ACM Symposium on Virtual Reality Software and Technology (VRST 2003)*. ACM, 2003, pp. 160–168.

[18] P. Morillo, J. M. Orduña, M. Fernández, and J. Duato, “A comparison study of metaheuristic techniques for providing qos to avatars in dve systems,” in *ICCSA’ 2004 - Lecture Notes in Computer Science 3044*. Springer-Verlag, 2004, pp. 661–670.

[19] FIPA, *FIPA Agent Management Specification*, 2000.

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

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

[22] N. Beatrice, S. Antonio, L. Rynson, and L. Frederick, “A multiserver architecture for distributed virtual walkthrough,” in *Proceedings of ACM VRST’02*, 2002.

[23] 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.

[24] 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*, vol. 1, no. 1, pp. 11–26, 1999.