

# A grid representation for Distributed Virtual Environments

Pedro Morillo, Marcos Fernández, Nuria Pelechano

Instituto de Robótica, Universidad de Valencia. Polígono Coma S/N.

Aptdo. Correos 22085, CP: 46071 Valencia, Spain

{Pedro.Morillo,Marcos.Fernandez,Nuria.Pelechano}@uv.es

**Abstract.** Fast Internet connections and the widespread use of high performance graphic cards are making Distributed Virtual Environments very common nowadays. The architecture and behaviour of these systems are very similar to new grid computing applications where concepts as sharing and high scalability are extremely exploited. However, there are several key issues in these systems that should still be improved in order to design a scalable and cost-effective system. One of these key issues is the partitioning problem. This problem consists of efficiently assigning clients (3-D avatars) to the arbiters (servers) in the system. As an alternative to the adhoc heuristic proposed in the literature, this paper presents a comparison study of two evolutionary heuristics for solving the partitioning problem in DVE systems. Performance evaluation results show that heuristic methods can greatly improve the performance of the partitioning method, particularly for large DVE systems. In this way, efficiency and scalability of DVE systems can be significantly improved.

## 1 Introduction

**Grid Computing** and **Distributed Virtual Environments** (DVE) have evolved together in the development of high performance 3D real-time applications. A grid system ([6]), composed of a network of heterogeneous computers, is the logical framework to simulate a virtual world that is shared by thousands of users.

In a DVE system a collection of users are working on different computers connected to the Internet running an interactive 3D graphic program that simulates a virtual scene. Each user is represented in the virtual environment by a humanoid called *avatar*. Since this system supports visual interactions between multiple users in the shared 3D virtual environment, every change in the simulation has to be propagated to other participants. These participants are located in another computers within the grid system.

In order to handle an important number of avatars in the simulation, traditional architectures for these applications are based on a *central server* model. In this model, detailed in [17], clients communicate with a central server, which manages the entire system and informs clients of any updates and changes in the state of avatars and objects. Clients only communicate with the stand-alone server, which contains the entire scene database and tracks all objects of interest within the system. The server

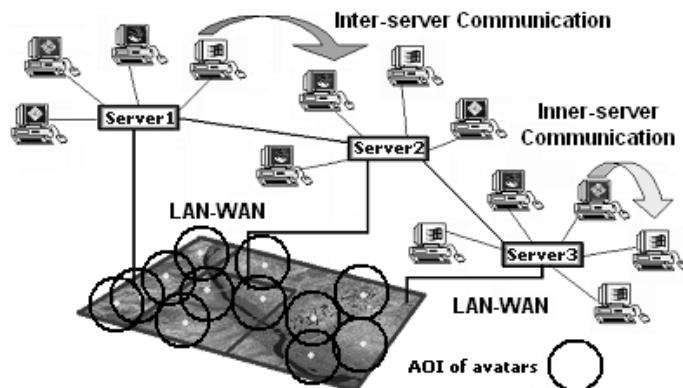
processes all input from each client, and dispatches the changes in the environment to each user. In [5] several driving simulators based on this model are analysed, as example of DVE. Different tests have demonstrated that these systems cannot support more than a hundred of avatars simultaneously. This is not only due to the amount of information managed, but also due to the important number of connections between them.

Described also in [12], *replicated* systems have appeared in order to handle a greater amount of connections. In this systems a set of federated servers shares the whole volume of data within the simulation. If the number of avatars increases significantly this common data volume, then unfeasible simulations appear.

Nowadays, parallel to the appearance of Grid computing applications, the way of developing these systems has changed. Instead of maintaining the whole simulation within a central computer, this is distributed among all the avatars that are currently running. Each of them controls and simulates his surrounding part of the scene and sends data to an arbiter node. This new methodology, named *server-network* architecture, has let to develop DVE systems with a large number of avatars geographically distanced. Offering low latency performance, the proposed architecture allocates three thousand avatars over Internet in real time. With this scheme, every avatar is sharing the same resources in the 3D virtual scene and the bottleneck within the central simulator is avoided.

In this sense, DVE can be seen as a large data Grid environment because of two main reasons. Firstly the total amount of data within a simulation is segmented among its members, and secondly the reconfiguration methods for failure tolerance that guarantee the correct working of the system

As figure 1 shows, these modern DVE contain a large set of clients and servers (data arbiters), where each client needs to be attached to one of the servers. When a client is updated it has to propagate a message through its server to the rest of clients within a certain area of interest. With this scheme, two kinds of communication messages are defined. Fast inner-server communications when both the sender and receiver are allocated in the same server, and long inter-server otherwise.



**Fig. 1.** Basic DVE communication model

Nowadays DVE systems are used in a wide variety of applications such as collaborative design ([16]), civil and military ([14]) distributed training simulations, distributed and interactive e-learning ([15]) and multiplayer games ([11]).

DVE applications achieve to simulate large-scale virtual worlds through dispersed and heterogeneous computational resources. However, there are several key issues in these systems that should still be improved in order to design a scalable and cost-effective DVE. One of these key issues is the **partitioning problem**. Redefined in [12] for DVE systems, this hot research problem consists in efficiently assigning the clients (3-D avatars) to the server nodes in the grid of computers. Good solutions to this grid task provide low latency and high scalability to the DVE system.

The partitioning problem deals with finding the best way to attach clients to servers in order to reduce the workload of messages within the grid network. The goodness of a client's partition is given by two parameters, one describing the client's workload and the other one describing the inner-server communication cost.

This paper shows the behaviour of DVE in grid systems and proposes two methods to solve the partitioning problem. These methods are based on evolutionary computation where a heuristic search scheme using Genetic Algorithms and Simulated Annealing are used to obtain good clustering solutions.

## 2 Partitioning Problem. State of the art

In spite of the heterogeneity described above, some common features are shared by the modern DVE systems. In the whole set of communication models presented in several papers ([2][17][18]), network-server architectures are becoming in a de-facto standard. Parallel or network-based distributed technologies developed on DVE, like high performance web-server management, are on the way to get the same conclusions ([9]). In these architectures, interconnected servers that perform as data arbiters organize simulation control.

Multiplatform clients are allocated to one of these servers; so that, for each avatar update a client sends one update message to a server, and the server propagates the message to other server and clients. In order to avoid a message outburst when the number of clients increases, areas of influence (AOI in [17], locales in [1] and aura in [9]) are defined for each one of the avatars. In this way, messages are only propagated from one avatar to the avatars that fall into its AOI. Depending on which is the destination avatar, inner-server and inter-server are messages are defined in a DVE scheme (see figure 1).

With this approach the traffic distribution load is moved out of the clients into the servers. For each local update, clients only send one message to a server and receive messages from a server in order to update all avatars' neighbours. Therefore, they must complete very little workload, storage or messaging in order to maintain a consistent state among many avatars in a big DVE.

Lui-Chan have redefined in [12] the partitioning problem as the difficulty of finding the best client-to-server assignment in order to ensure a good frame-rate for the applications and to minimize the network traffic.

To achieve this goal, two factors must be considered in an optimal design of a DVE:

On one hand, every server should share out the client's workload in a balanced way. This will be given by the factor:  $C_P^W$ . Logically, it is not necessary for the servers to be identical, but their pondered hardware features have to support the same number of clients. (It could be possible to have, for example, a 150-avatars DVE, in a network constituted by 10 identical Pentium IV computers PCs, each of them, shout allocate 15 avatar, more or less. Or it could also be possible to have an heterogeneous network formed by 4 Pentium IV PCs, 2 Silicon Graphics Reality Monster and 3 IBM eServer. They could host 10, 15 or 20 avatars respectively).

On the other hand ( $C_P^L$ ), in order to minimize the overall inter-server traffic, neighbouring avatars should be assigned to the same server. Due to topological reasons it is not possible, for the majority of the cases, to satisfy this condition. It is therefore necessary to find a suitable topological grouping.

According to these two parameters a cost evaluation function is offered to estimate the goodness of a clients partition:

$$C_\rho = W_1 C_P^W + W_2 C_P^L \text{ Such as } W_1 + W_2 = 1 \quad (1)$$

$W_1$  and  $W_2$  denote the relative importance of the computational workload and the inter-server communication cost mentioned above. In the general formal case  $W_1$  is equal to  $W_2$  and equal to 0.5. It is evident that, when a DVE is working on a high performance network, the quotient  $W_1/W_2$  may be much greater than one. Quite the opposite for the DVE working in a slow-shared network or Internet, where large messages latencies make the quotient to be close to zero. During the simulation time, the algorithm, that is continuously regrouping clients, must be run several times. The execution rate is published in [13].

In [12] authors demonstrate that the partitioning problem is NP-complete and offer a platform test to scientific community where every approximations to the problem can be checked and compared. Furthermore, a refinement of their initial algorithm is proposed and a parallelization based on the amount of avatar criteria is presented in [13]. This refinement divides the algorithm in three parts: a recursive bisection partitioning that carries out an initial allocation based on a cells criterion, a layering partitioning that maximizes workload balance, and a communication refinement partitioning that maximizes inner-server communication. Depending on DVE size or state, some of them can be repeated in the same execution step.

There are other approaches, with different denominations, to this problem published in [2] and [18]. In [18] it is described an approach that groups the avatars following a regular distribution. To ensure good graphic performances for the clients, this algorithm generates a number of grouped avatars equal to the number of simulation servers.

This solution did not obtain good results when avatars were located in a non-uniform distribution. Other approach ([2]) rejects dynamic concepts associated to avatars like aura, AOI or locale. This technique divides the 3D virtual scene in a regular grid. For each cell, a multicast group is created, so that avatars sharing a cell

are assigned to the same server saving a lot of messages and sharing multicast packets. Despite it was a quick and determinist solution, this static classification performs badly when avatars are distributed in a non-uniform way within the 3D virtual scene. The fact that there may be some servers massively charged and many avatars in the frontiers will increase the global cost.

### 3 A new Approach

Solutions mentioned above for the partitioning problem share the same feature: they are custom solutions that are defined like ad-hoc heuristics to solve the problem.

As other researchers carried out for another kind of problems, the solutions presented in this paper are based on evolutionary computation. In various domains of applications these heuristics search methods imitate principles of the nature in order to solve complex problems. Although simplistic from a biologist's viewpoint, these algorithms are sufficiently complex to provide robust and powerful adaptive search mechanisms. Although a variety of evolutionary algorithms had been proposed, this paper focuses on design and development of a solution for the partitioning problem based on genetic algorithms and simulated annealing.

*Genetic Algorithms (GA)*, using biology concepts, provide a learning method based on the idea of evolution by natural selection. Although this idea dates back to the seventies it is not until the middle eighties when its creators Holland and Goldbert managed to achieve fast and satisfactory solutions for some complex problems.

Detailed in [8], GA start creating a population of pre-solution that will evolve generation by generation until it achieves a good quality population of post-solutions.

To solve the partitioning problem, the method proposed in this paper starts from a population composed by a set of elements, called *genomes* or *chromosomes*. For each iteration, the number of chromosomes is equal to the number of partial solutions that the GA needs. For a given time step, each chromosome will be defined by a descriptor vector giving a possible configuration for the partitioning of the DVE. Therefore, each element of this vector will have assigned a range of values equal to the number of servers within the DVE.

The Genetic solution to the partitioning task follows four steps. Each of them has been adapted to the specifications of the particular problem:

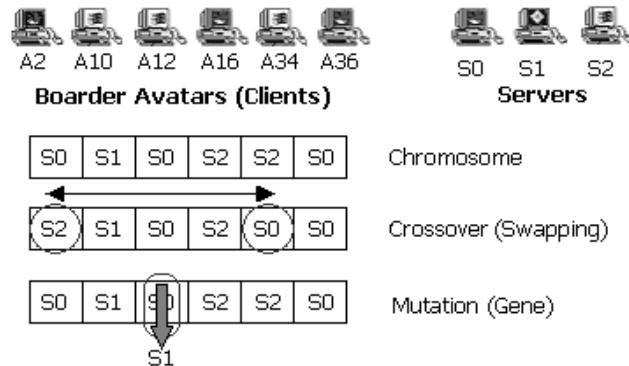
a) *Initial solutions*. To avoid the algorithm spending some of its initial iterations trying unsatisfactory combinations, good initial values need to be found in order to obtain low initial Cp costs. A method from pattern recognition, based on Graph-Theoretic ([4]), has been adapted to achieve fast initial solutions.

b) *Number of generations*. It is necessary to set a maximum value for the number of generation in order to reduce the cost of the algorithm without increasing the computational time needed to obtain the solutions.

c) *Inheritance mechanism based on derivation*. Starting with the initial solution new generations will be found by iteratively applying crossover between elements of the population. This crossover is represented by a derivative or inheritance scheme.

Therefore, during each iteration a new child will be created for each of the N elements of the current population. Among the  $2N$  chromosomes of the new population, the N elements with lower Cp will be chosen to be used for the next iteration. In order to create a child solution from a parent, two boarder avatars (BA) assigned to different servers will be randomly chosen from the current N chromosomes. BA is an avatar that although it lies within the AOI of another avatar, it is not assigned to the same server. These kinds of avatars will be the ones offering a higher probability of giving a successful permutation since they minimize inter-server communication.

d) *The mutation process.* Mutation is needed to escape from local minima. Once the child-vector has been obtained, mutation involves changing at random the server assigned to one of the elements of the population. Some other mutations such as the modification of several elements or the crossover between the characteristics of pairs of chromosomes have also been tested. But after several tests it has been found that the mutation of just one “bit” is the one offering the best results. It is important to mention that the mutation is a random process controlled by a parameter which needs to be carefully chosen for each specific experiment in order to achieve solutions with low Cp System Cost.



**Fig. 2.** Iteration breeding

In the above figure it is represented how six BA, obtained from the full set of avatars, are selected and defined as a chromosome. Then a crossover and a mutation are applied in order to obtain the breeding for the next generation.

The second solution is based on *Simulated Annealing* (SA). SA is a Monte Carlo approach for minimizing multivariate functions. Described in [11], this heuristic search method is based on a thermodynamic theory establishing that the minimum energy state in a system can be found if the temperature decreasing is performed slowly enough. Simulated Annealing (SA) is a heuristic search method that always accepts better solutions than the current solutions, and also accepts worse solutions according to a probability system based on the system temperature.

SA starts with a high system temperature (a high probability of accepting a worsening movement), and in each iteration system temperature is decreased. Thus, SA can leave local minima by accepting worsening movements at intermediate stages.

The search method ends when the system temperature is so low that worsening movements are practically impossible. Since the method cannot leave local minima, it cannot find better solutions, neither (the algorithm ends when certain amount of iterations  $N$  are performed without finding better solutions).

Each iteration consists of randomly select two different critical avatars assigned to different servers. Then, the servers that two critical avatars are assigned to are exchanged.

If the resulting value of the quality function  $C_p$  is higher than the previous one plus a threshold  $T$ , that change is rejected. Otherwise, it is accepted (the search method must decrease the value of the quality function  $C_p$  associated with each assignment). The threshold  $T$  used in each iteration of the search depends on the rate of temperature decreasing  $R$ , and it is defined as

$$T = R - \left( \frac{R \times i}{N} \right) \quad (2)$$

where  $N$  determines the finishing condition of the search. When  $N$  iterations are performed without decreasing the quality function  $C_p$ , then the search finishes.

As literature shows [11], the two key issues for properly tuning this heuristic search method are the number of iterations  $N$  and the temperature decreasing rate  $R$ . Although they are not shown here for the sake of shortness, we obtained the best results for SA method with  $N=3000$  iterations and  $R=1.25$ . This procedure allows the system to move consistently towards lower energy states, yet still avoids local minima due to the probabilistic acceptance of some upward moves. Number of iterations and cooling schedule were characterized in the process to obtain high-quality clustering solutions.

## 4 Performance results

Following the evaluation methodology formulated in [13], it has been empirically tested adapted evolutionary heuristics in two examples of a DVE system: a SMALL world, composed by 13 avatars and 3 servers, and a LARGE world, composed by 2500 avatars and 8 servers. Two parameters has been considered: the value of the quality function for the obtained partition and also the computational cost, in terms of execution time, required by the search method in order to provide that partition.

Since the performance of the heuristic search methods may heavily depend on the location of the avatars, it has been considered three different distributions of avatars: uniform, skewed, and clustered distribution.

Next tables show the values corresponding to the final partitions calculated by each heuristic search method for both worlds, as well as the execution times required for each heuristic search method to obtain that final partition.

**Table 1.** Results in virtual Worlds (SMALL)

(a) Avatars in a Uniform Distribution

	Computation t(s)	System Cost Cp
Exhaustive Alg.	3.411	6.54
Lui-Chan Alg.	0.0009	6.56
<b>Genetic Alg.</b>	0.002	6.54
<b>S.A. Alg.</b>	0.004	6.54

(b) Avatars in a Skewed Distribution

	Computation t(s)	System Cost Cp
Exhaustive Alg.	3.843	7.04
Lui-Chan Alg.	0.0010	8.47
<b>Genetic Alg.</b>	0.003	7.04
<b>S.A. Alg.</b>	0.005	7.46

(b) Avatars in a Clustered Distribution

	Computation t(s)	System Cost Cp
Exhaustive Alg.	4.783	7.91
Lui-Chan Alg.	0.0011	8.76
<b>Genetic Alg.</b>	0.005	7.92
<b>S.A. Alg.</b>	0.004	7.91

**Table 2.** Results in virtual Worlds (LARGE)

(a) Avatars in a Uniform Distribution

	Computation t(s)	System Cost Cp
Lui-Chan Alg.	30.939	1637.04
<b>Genetic Alg.</b>	6.598	1779.76
<b>S.A. Alg.</b>	6.35	1707.62

(b) Avatars in a Skewed Distribution

	Computation t(s)	System Cost Cp
Lui-Chan Alg.	32.176	3460.52
<b>Genetic Alg.</b>	14.593	2825.64
<b>S.A. Alg.</b>	13.789	2628.46

(c) Avatars in a Clustered Distribution

	Computation t(s)	System Cost Cp
Lui-Chan Alg.	43.314	5903.80
<b>Genetic Alg.</b>	29.198	4905.93
<b>S.A. Alg.</b>	29.620	4697.61

For the case of SMALL worlds, since a full solution tree scanning requires only to explore  $3^{13}$  (1.594.323) different solutions, it is also possible to apply an exhaustive algorithm in order to compare the different computational times obtained. However it is not feasible to apply an exhaustive algorithm for the case of LARGE worlds where the solution tree has a dimension of  $8^{2500}$  different results.

From obtained results, it can be seen that both proposed heuristics provide better (lower) Cp values than the Lui-Chan search method for skewed and clustered distribution of avatars. Also for all the distributions in SMALL worlds, these achieved Cp values are very close to the obtained by an exhaustive search.

The goodness of proposed algorithms becomes important when the virtual world size increases to LARGE worlds. Execution times are significantly reduced for all the distributions of avatars and obtained solutions provide solutions where the load balancing and the amount of inter-server messages are improved.

These results have been obtained using a grid system composed of a network of PC-based computers. Nowadays, this grid system is installed at the University of Valencia. Although having a heterogeneous network, all the nodes satisfy some basic graphics requirements which are being Pentium IV processors with an nVidia graphic card.

## 5 Conclusions and future work

In this paper, a correlation between modern DVE architectures, based on network-server models, and the behaviour of the last grid computing applications has been established. Because of information segmentation among the members of a large DVE and all developed mechanisms to handle it, modern distributed virtual environments perform as any other grid applications with a specific set of problems.

In this paper, a new method for solving the problem of efficient client allocation to simulation servers is presented. This aspect is a key issue that allows to design scalable and efficient DVE systems. Instead of developing an ad-hoc technique for this NP-complete problem, as other research groups, two methods based on genetic programming and simulated annealing have been adapted to the problem specifications.

The experiments performed in the proof-of-concept system demonstrate that the proposed approaches obtain better quality solutions than the Lui-Chan algorithm for the same context tests. Moreover it also achieves better scalability in large-scale DVE systems since it presents lower execution time as the virtual world sized is increased.

As future work, the development of an efficient parallelization of the proposed algorithm based on a master-slave scheme is planned. The obtained results in current research are currently being applied in the development of a collaborative driving simulator. In this system, future drivers will share a 3D virtual town where hundreds of drivers will be connected through a low bandwidth network such as the Internet.

## References

1. D.B.Anderson, J.W.Barrus, J.H.Howard, "Building multi-user interactive multimedia environments at MERL", in *IEEE Multimedia*, 2(4), pp.77-82, Winter 1995.
2. P.Barham, T.Paul, "Exploiting Reality with Multicast Groups", in *IEEE Computer Graphics & Applications*, pp.38-45, September 1995.
3. H.Delmaire, J.A. Díaz, E. Fernández, and M. Ortega, "Comparing new heuristics for the pure integer capacitated plant location problem", Technical Report DR97/10, Department of Statistics and Operations Research, Universitat Politècnica de Catalunya, Spain, 1997.
4. R.Duda, P.Hart, D.Stork, "Pattern Classification", *Ed. Wiley Interscience*, 2000, pp. 567-580.
5. M.Fernández, I. Coma, G.Martín and S.Bayarri, "An Architecture for Optimal Management of the Traffic Simulation Complexity in a Driving Simulator", *Lecture Notes in Control and Information Sciences*, Springer-Verlag, Vol. 243, 1999. ISBN 1-85233-123-2.
6. I.Foster, C.Kesselman, eds. "The Grid: Blueprint for a New Computing Infrastructure". Morgan Kaufmann, San Francisco, California. 1999.
7. I.Foster, C.Kesselman, S.Tuecke. "The Anatomy of the Grid: Enabling Scalable Virtual Organizations" International J. Supercomputer Applications, Vol. 15(3), 2001.
8. Randy L. Haupt, Sue Ellen Haupt, "Practical Genetic Algorithms", *Ed. Willey*, 1997.
9. J.C.Hu, I.Pyarali, D.C.Schmidt, "Measuring the Impact of Event Dispatching and Concurrency Models on Web Server Performance Over High-Speed Networks", *Proc. of the 2nd. IEEE Global Internet Conference*, November.1997.
10. P.V. Laarhoven and E. Aarts, "Simulated annealing: Theory and applications", *Reidel Pub.*, Dordrecht, Holland, 1987.
11. Michael Lewis and Jeffrey Jacobson, "Game Engines in Scientific Research", in *Communications of the ACM*, Vol 45. No.1, January 2002.
12. John C.S. Lui, M.F.Chan, Oldfield K.Y, "Dynamic Partitioning for a Distributed Virtual Environment", *Department of Computer Science*, Chinese University of Hong Kong, 1998.
13. Jonh C.S. Lui, M.F. Chan, "An Efficient Partitioning Algorithm for Distributed Virtual Environment Systems", *IEEE Trans. Parallel and Distributed Systems*, Vol. 13, March 2002
14. D.C.Miller, J.A. Thorpe, "SIMNET: The advent of simulator networking", in *Proceedings of the IEEE*, 83(8), pp. 1114-1123. August, 1995.
15. Tohei Nitta, Kazuhiro Fujita, Sachio Cono, "An Application Of Distributed Virtual Environment To Foreign Language", in *IEEE Education Society*, October 2000.
16. J.M.Salles Dias, Ricardo Galli, A. C. Almeida et al. "mWorld: A Multiuser 3D Virtual Environment", in *IEEE Computer Graphics*, Vol. 17, No. 2, March-April 1997.
17. S.Singhal, and M.Zyda, "Networked Virtual Environments", *ACM Press, New York*, 1999.
18. P.T.Tam, "Communication Cost Optimization and Analysis in Distributed Virtual Environment", *M. Phil second term paper, Technical report RM1026-TR98-0412*. Department of Computer Science. The Chinese University of Hong Kong. 1998.