M-GRASP: A GRASP with Memory for Latency-Aware Partitioning Methods in DVE Systems

Journal:	*Transactions on Systems, Man, and Cybernetics--Part A: Systems and Humans*
Manuscript ID:	SMCA07-11-0353.R1
Manuscript Type:	Regular Paper
Date Submitted by the Author:	20-Oct-2008
Complete List of Authors:	Orduña, Juan; Universidad de Valencia, Departamento de Informática Morillo, Pedro; Universidad de Valencia, Departamento de Informática Duato, Jose; Universidad Politécnica de Valencia, DISCA
Key Words:	Distributed Virtual Environments, Partitioning method, GRASP
Note: The following files were submitted by the author for peer review, but cannot be converted to PDF. You must view these files (e.g. movies) online.	
StandAloneSimulator.tgz	

M-GRASP: A GRASP with Memory for Latency-Aware Partitioning Methods in DVE Systems

Pedro Morillo, Juan M. Orduña, and José Duato, *Member, IEEE*

***Abstract*—A necessary condition for providing quality of service to Distributed Virtual Environments is to provide a system response below a maximum threshold to the client computers. In this sense, latency-aware partitioning methods try to provide response times below the threshold to the maximum number of client computers as possible. These partitioning methods should find the assignment of clients to servers that best optimizes system throughput, system latency and partitioning efficiency.**

In this paper, we present a new algorithm based on GRASP with memory for finding the best solutions as possible to this problem. We take into account several different alternatives in order to design both the constructive phase and the local search phase of this multi-start metaheuristic for combinatorial problems. Additionally, we enhance this basic approach with some intensification strategies that improve the efficiency of the basic search method. Performance evaluation results show that the new algorithm increases the performance provided by other meta-heuristics when applied to solve the latency-aware partitioning problem in DVE systems.

P. Morillo and Juan M. Orduña are with the Departamento de Informática, Universidad de Valencia, Valencia, SPAIN. e-mail: {Pedro.Morillo, Juan.Orduna}@uv.es

José Duato is with DISCA, Universidad Politécnica de Valencia, Valencia, SPAIN. e-mail: jduato@disca.upv.es

Manuscript received April 19, 2005; revised January 11, 2007.

***Index Terms*—Distributed Virtual Environments, Partitioning method, GRASP.**

AUTHORS RESPONSE TO THE REVIEW OF SMCA07-11-0353

Editor

Comment: The reviewers point to several technical issues that need to be further addressed, and I agree with the technical issues/concerns raised. Particularly the authors need to highlight their contributions in this paper, in the light of their previous work.
Answer: We have addressed each one of these issues, as detailed below. We hope that the contribution of the paper is now clear.

Reviewer 1

Comment: The following references are related to this paper and necessary to add them into the references. ...
Answer: The required references have been included in the reviewed manuscript.
Comment: ***MINOR CORRECTIONS*****
Answer: All the minor corrections have been included in the reviewed manuscript
Comment: *****About simulation software******
The attached software ”StandAloneSimulator” could not be complied and linked easily as executable soon. So I could not verify if his performance data is correctly reproduced with the provided C++ program. I suggest authors further provide a FTP address for downloading the executable files for further direct verification of his results.
Answer: Due to technical/political reasons

related to our institution, we cannot provide a public FTP address for downloading the executable file. Therefore, we provide this executable file at the link http://grev.uv.es/grev/files/MGRASP-IEEE-SMC.exe. It is compiled for a PC with a 32 bits Windows O.S..

Reviewer 2

Comment: I would suggest the revision of this paper, by

- avoiding extended repetitions of previous papers
- focusing and explaining better the new results making clear the additional contribution of this paper

Answer: We have modified Section I to highlight the additional contribution of this paper, and we have completely rewritten Section II, completely avoiding the repetition of previous papers.

Comment: 1. Concerning (P) (equation (4)), I would appreciate an answer on the following question: Is RTT independent to the number of avatars, which is a parameter taken into account in the factor (P)? In other words: are (P)and (P) independent the one with the other?

Answer: As shown in reference 11 of the original manuscript, RTT is independent of the number of avatars while the percentage of CPU utilization in all the servers is kept below 100%. This is the reason for defining the first term of the quality function with an inverse exponential behavior: first, we MUST keep the CPU utilization in all the servers below 100%, and then we should try to minimize the RTT for all the clients. We carefully explain this issue in the reviewed manuscript (Section II).

Comments:

2. Page 5, 1st column, line 16: th word ”therm” should change to: ”term” 3. Page 5, 1st column, line 24: th word ”Greedy” should change to: ”Greedy” 4. The acronym GRASP is analysed in page 5. However, it is refered in the first page of the paper. I propose to analysed it in the first section of the paper (I Introduction)

Answer: We have included these minor comments in the reviewed manuscript.

Comment: Page 6: References needed for the techniques RS, RS* and GG presented.

Answer: We have included these references in the reviewed manuscript.

Comment: Page 9. 2nd column, 1st paragraph: It is refered that: ”Table V shows the results for the combinations that provided the best results.” It would be useful to see the results for the rest of cases, i.e. RS+OS RS+1F RS+2S RS*+2S

I would propose the same for both Tables VI and VII (page 10)

Answer: We have added the results for these combinations in both Table V and Table VI. However, we have not included these combinations in Table VII. In this case, we think that the number of possible combinations when adding two possible intensifications results in huge tables with too much information, where only half (more or less) of this information is valuable.

Anyway, if the reviewer thinks that it is crucial to add these combinations also to tables VII and VIII, we will add it in a further revision of the manuscript.

Comments:. Page 10. 2nd column, 1st paragraph, line 12: please place a space among the words "up,down,new, none,random and both)"

8. Page 10. 2nd column, 2nd paragraph, line 20: There is a reference to "Table III-C". However, there is no such a table.

Answer: These changes have been included in the reviewed manuscript.

Reviewer 3

Comment: The first part of the paper (until Section III), which is the same to the paper described in reference 10, should be completely changed. The authors should focus on the elements and concepts of the M-GRASP idea and the exact use of already published work should be avoided.

Answer: Section I has been modified to highlight the differences and contributions of this paper regarding reference 10. Section II has been completely changed, focusing on the detailed description of the quality function and the reasons behind that function. No previously published material at all appears in the reviewed manuscript.

Comment: Equations 3, 5, 7 and 8 are not explained. As these relationships are vital for the formation of the approach, the authors should describe them in a more analytical and clear way as in their current form the relations seem arbitrary.

Answer: The entire section has been rewritten, and every equation is explained in the reviewed manuscript.

Reviewer 4

Comment: Some points of the paper that need additional explanation are the following: -It is not clear how the relationship in equation 3 (page 3) comes up. It needs additional explanation. -It is not clear how the relationship in equation 5 (page 4) comes up. It needs additional explanation. -It is not clear how the relationship in equation 6 (page 4) comes up. It needs additional explanation. -It is not clear how the relationship in equation 7 (page 5) comes up. It needs additional explanation. -It is not clear how the relationship in equation 8 (page 5) comes up. It needs additional explanation.

Answer: All these relationships are carefully explained in Section II, since this section has been completely re-written.

Comment: Furthermore, the paper should mainly focus on the main aspects of the new approach M-GRASP so as to make clear the additional contribution when compared to the paper presented in reference [10].

Answer: Section I has been modified to focus on this aspect, and Section II has been completely re-written.

M-GRASP: A GRASP with Memory for Latency-Aware Partitioning Methods in DVE Systems

I. Introduction

Distributed Virtual Environment (DVE) systems simulate a virtual world where multiple remote users share the same 3D synthetic scene. These highly interactive systems allow multiple users, working on different client computers interconnected through different networks, to interact in a shared virtual world. 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. Each user is represented in the shared virtual environment by an entity called *avatar*, whose state is controlled by the user through the client computer. Hundreds and even thousands of client computers can be simultaneously connected to the DVE system through different networks, and even through the Internet. Usually, in a DVE the information on virtual objects, including their locations and shapes, are maintained in the system server(s). When an avatar moves within the virtual environment, the information about the virtual objects located within a visible distance from the avatar is conveyed to the avatar's client machine. The information is processed and rendered by the client computer in a timely fashion. DVE systems are currently used in different applications such as civil and military distributed training [1], collaborative design [2] and e-learning [3]. Nevertheless, the most extended example of DVE systems are commercial multi-player game environments. These systems use the same simulation techniques that DVE systems do [4]. A lot of research has been made on these highly interactive systems [5]–[9].

Architectures based on networked servers have become a de-facto standard for DVE systems [10]–[15]. In these architectures, the control of the simulation relies on several interconnected servers. Client computers are attached to one of the servers in the system. When a client computer modifies the state (usually the position) of an avatar, it also sends an updating message to the server where it is assigned, which in turn must propagate this message to other servers and clients. Servers not only perform positional updates of avatars and transfer control information among different clients, but also decide how these updating messages are managed.

A key issue in the design of DVEs based on networked server is the partitioning method. It consists of finding an assignment of clients to servers that maximizes system throughput, and different approaches have been proposed [11]–[13], [16]. All of these approaches exclusively focus on properly balancing the workload supported by the servers, in order to avoid the system saturation as much as possible. On step ahead consists of designing a partitioning method that it is oriented not only to balance the workload supported by the servers, but also to provide the client computers with a system response below a threshold value. In a previous work, we proposed a new partitioning method that also allows the system to satisfy, if possible, any specific latency requirement that avatars can have [17]. That approach is based on a heuristic method that searches a near-optimal solution in the domain of all the possible assignments of avatars to the existing servers (if the DVE system consists of S servers and N avatars, then S^N possible assignments exist). As a result, the performance of the partitioning method is determined by the performance of the heuristic search method. In this sense, the heuristic search methods proposed in that previous work provide better DVE performance than those approaches not considering the latency provided to avatars, and particularly the Greedy Randomized Adaptive Search Procedure (GRASP) seems to be the one providing the best performance. However, the proposed heuristic methods have some limitations. For example, the GRASP method shown in that work is hardly parallelizable, and it needs the whole iterations to be executed in order to provide a feasible solution. As a result, the time constraints required for the search can be too hard if the search domain (the number of avatars and servers in the system) is very large. Therefore, an optimized heuristic method (specifically adapted to this problem) is required in order to guarantee the performance of the partitioning method.

In this paper, we propose a new design of the GRASP for solving the latency-aware partitioning problem in DVE systems. The idea is to start from the scratch and to design a specific GRASP that can be parallelizable and that can provide a feasible solution for the considered problem at any iteration, in such a way that it can be adapted to any time constraint. Since each iteration in GRASP consists of a constructive phase and a local search phase, we propose different alternatives for each phase, evaluating the performance obtained with each alternative. Additionally, we enhance this basic approach with some intensification strategies, selecting the option with the best performance evaluation results as the proposed final implementation for GRASP. In order to compare the performance of the new design of the GRASP with the performance of the methods shown in our previous work, we have used the same cost function (the function that measures the quality of each assignment considered by the heuristic method) proposed in our previous work. The performance evaluation results show that the quality of the provided solutions are significantly increased, meaning an actual improvement in the partitioning method and also in the performance of these interactive, distributed systems.

The rest of the paper is organized as follows: Section II describes in detail the problem to be solved and the cost function used for measuring the quality of each assignment. Section III discusses the alternative options in the design of GRASP with memory and defines a final design of M-GRASP using the alternatives that show the best performance. Next, section IV shows a comparison study of the method proposed in the previous section with other heuristic methods previously proposed. Finally, section V presents some concluding remarks.

II. Background

In this section, we explain the requirements that the partitioning method should fulfill in order to improve the performance of DVE systems based on networked servers, and how we have modeled these requirements in a cost function. The heuristic method will have to use this cost function for measuring the quality of each partition (assignment of all the avatars to the servers in the system) considered.

DVEs are dynamic systems where avatars (controlled by users through the client computers) can freely move, and even leave or join the system. As the system evolves, the workload supported by the different servers may become unbalanced, and/or the latency provided to the client computers can increase beyond a given threshold. Whenever any of these events occur, the partitioning algorithm is executed in order to keep the system below the saturation point and/or to reduce the latency provided to the client computers as much as possible. Is at this point when the heuristic method should find a near-optimal (or at least feasible) partition within a limited period of time. Otherwise, the current state of the system (the location of avatars) will significantly differ from the state considered by the heuristic method, and therefore it will provide obsolete solutions. These time constraints can be very hard when the population increases.

Taking into account the time constraints, the heuristic method should provide a partition that fulfills three different (and in some cases conflicting) conditions. The first (main) condition consists of ensuring that the workload generated by all the avatars is properly balanced among all the servers in the system, in such a way that the saturation of any server is avoided. Otherwise, the entire system enters saturation and the system performance greatly decreases [16].

The second condition is that the partition found should reduce the system latency provided to avatars by means of assigning neighbor avatars (assigning the client computers controlling avatars that are close in the virtual world) to the same server while possible (it should be noticed that this condition may conflict with the first one, depending on the location of the avatars in the virtual world). The reason is that in DVEs based on networked servers, the latency of the messages exchanged between client computers assigned to different servers can be much higher than the latency of those messages exchanged between client computers assigned to the same server [17].

Finally, the overhead of migrating many avatars from one server to another one should be taken into account. In order to limit this overhead, the partition found should not differ a lot from the current partition.

The heuristic search method must use a cost function for evaluating the relative quality of each of the partitions (solutions) considered. This cost function should take into account the three conditions explained above. In order to compare the performance of the new design of the GRASP with the performance of the methods shown in our previous work, we have used the same cost function, denoted as f_{QoS}. In order to make this paper self-contained, we briefly explain here the definition of this cost function. Concretely, we have defined f_{QoS} for a given partition P as

$$\begin{aligned} f_{QoS}(P) &= \sum_{i=1}^{s} h_{cpu}(i, est(P)) \; + \\ & \sum_{i=1}^{n} h_{asr}(i, lat(P)) \; + \; \Gamma(P) \end{aligned}$$

where each term measures the quality of the partition with respect to one of the conditions explained above. The first term consists of the aggregated sum of the values provided by a function that is applied to the S servers in the system. For each server i, the function $est(P))$ estimates the percentage of CPU utilization of i for partition P, based on the characterization method proposed in [16]. This estimation uses the movement rate of the avatars and the average number of neighbors to estimate the workload (measured in percentage of CPU utilization) that each avatar represents for a server. According to partition P, the total percentage of CPU utilization of a given server can be computed. Depending on this percentage, function h_{cpu} assigns a different value. If partition P results in an estimated percentage of CPU utilization for server i that is significantly lower than 100%, then the value of $h_{cpu}(i, est(P))$ should be very low. However, this value should greatly increase if server i reaches a CPU utilization close to 90%, and it should shoot up if the percentage of CPU utilization goes beyond this value for partition P. In this way, we ensure that the heuristic search will reject any partition where any of the servers shows an estimated percentage of

CPU utilization greater than 90% (the heuristic method should minimize the cost function in this case). In other words, the purpose of this term is to prevent the selection of an unbalanced partition, avoiding in this way that the system enters saturation. Concretely, the function $h_{cpu}(i, est(P))$ can be defined as

$$h_{cpu}(i, est(P)) = \frac{10000}{100.1 - est(P)} \qquad (1)$$

where i is the considered server and $est(P)$ is the estimated percentage of CPU utilization for that server, ranging from 0 to 100 %.

The second term in f_{QoS} takes into account the second condition explained above. It consists of the aggregated sum of the values provided by a function that is applied to the n client computers in the system. Concretely, $h_{asr}(i, lat(P))$ is a function that evaluates the estimated average round-trip delay $lat(P)$ that the messages sent by client computer i would have in partition P (for the sake of shortness, in the rest of the paper we will use the term avatar also to denote the client computer controlling the avatar). If a given client i is assigned to a server s, then $lat(P)$ can be estimated taking into account the number of avatars in the neighborhood of i that are assigned to other servers distinct from s, as shown in [17]. Depending on the value of $lat(P)$, $h_{asr}(i, lat(P))$ should have different behaviors. If the estimated round-trip delay for the messages sent by avatar i is below 250 ms., then $h_{asr}(i, lat(P))$ should have an inverse exponential behavior, because it is an acceptable value. If that value is greater than 250 ms., then $h_{asr}(i, lat(P))$ should have a parabolic behavior, because as higher this value is, the less this partition should be chosen. In this way, the cost function f_{QoS} will tend to choose those partitions that result in a lower average round-trip delay for the avatars, particularly if this value is below 250 ms. The threshold value of 250 ms. has been chosen because it has been shown as the human limit for considering a virtual environment as interactive [18], [19]. Concretely, $h_{asr}(i, lat(P))$ can be defined as

$$h_{asr}(i, lat(P)) = \frac{100/\sqrt{250}}{\sqrt{lat(P)}} \qquad (2)$$

for the interval $lat(P) \in [0..250]$, and

$$\begin{aligned} h_{asr}(i, lat(P)) \quad = \quad & 0.0016 lat(P)^2 - \\ & 0.04 lat(P) + 10 \end{aligned}$$

for the interval $lat(P) \in [250..\infty)$.

Finally, the last term in function f_{QoS} is a function that takes into account the third condition explained above. Concretely, $\Gamma(P)$ evaluates the partitioning efficiency of partition P, that is, the number of avatars that must be migrated in order to reach partition P from the current partition. A recent work shows that an efficient partitioning method must not migrate more than 30% of avatars (client computers) in the DVE system [20]. Therefore, $\Gamma(P)$ should have different behaviors, depending on the number of avatars that should be migrated to reach partition P. We have constructed $\Gamma(P)$ with a linear behavior section from the zero value to one third of the existing avatars, and a parabolic section from that value up. Concretely, $\Gamma(P)$ can be defined as

$$\Gamma(P) = 22500/n \cdot m, \; m/n \in [0..0.3333] \qquad (3)$$

For the interval $m/n \in [0..0.3333]$, the function would be defined as

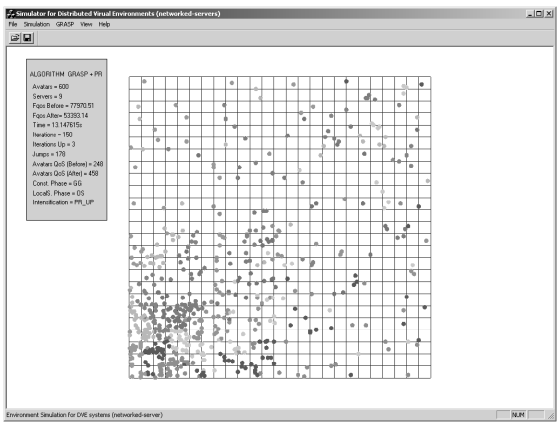
$$\begin{aligned} &\Gamma(P) = 7500+ \\ &+(82500/n)(m/n - n/3)+ \qquad (4) \\ &+(78750/n^2)(m/n - n/3)(m/n - 2n/3) \end{aligned}$$

where n is the number of existing avatars and m the number of avatars in the current partition that have to be assigned to a different server in order to achieve partition P.

Although other values and/or different behaviors could have been selected for the three terms in f_{QoS} (and even for the definition itself of f_{QoS}), we have selected these values in order to obtain performance evaluation results that are comparable with the results provided by the techniques proposed in our previous work [17].

III. GRASP Design

GRASP is a multi-start method for combinatorial problems in which each iteration consists of generating a randomized greedy solution and applying local search on it [21], [22]. In this section, we analyze both the construction and the local search phases of GRASP when applied to solve the latency-aware partitioning problem in DVE systems. Concretely, we propose different constructive methods and different local search methods. In the same way, we study the effect of two different intensification techniques in order to improve the quality of the final solutions for this problem. The implementation of all the techniques shown below are in the StandAloneSimulator attachment. This attachment contains all the source code of these implementations, as well an explanation file (EXPLANATION.TXT) that can be used as a user guide. As an example of the use of the provided software, Figure 1 shows a snapshot of the provided software. Concretely, this snapshot was taken when executing the Greatest Group technique with the Oriented Search technique and the Path-Relinking-UP intensification method (see the descriptions of these methods below) in a DVE configuration of nine servers and six hundred avatars. The initial distribution of avatars was the skewed distribuiton. In this Figure, the virtual world is represented as a 2-D square, each avatar is represented by a dot and the assignment for each avatar is encoded as a different grey level for the corresponding dot.


Simulator for Distributed Virual Environments (networked-servers)
File
Simulation
GRASP
View
Help
Environment Simulation for DVE systems (networked-server)
NUM

ALGORITHM GRASP + PR
Avatars = 600
Servers = 9
Fqos Before = 77970.51
Fqos After= 53393.14
Time = 13.147615s
Iterations = 150
Iterations Up = 3
Jumps = 178
Avatars QoS (Before) = 248
Avatars QoS (After) = 458
Const. Phase = GG
LocalS. Phase = OS
Intensification = PR_UP

Fig. 1. Snapshot of the StandAloneSimulator simulator

A. Constructive Phase

The construction phase of each iteration generates a greedy randomized solution. At this phase, let the set of candidate elements be formed by all elements that can be incorporated to the partial solution under construction without destroying feasibility. The selection of the next element to be added is determined by the evaluation of all candidate elements according to a greedy evaluation function. Usually the constructive phase starts with an empty set and the greedy function adds elements to this set until a feasible solution is reached.

However, due to the nature of the latency-aware partitioning problem in DVE systems, the construction phase of the GRASP method must start from accelerated populations (partitions) that keep the system below its saturation point [16]. GRASP will provide a partition that not only keeps the system away from saturation, but also provides the maximum number of avatars with quality of service. Therefore, we have used the partition provided by the FGALB method [23] as the initial starting point for the constructive phase. FGALB method provides a partition ensuring that all of the servers in the system have a percentage of CPU utilization below their saturation point (a partition P minimizing the first term in $f_{QoS}(P)$). Otherwise, the system could be unstable due to the behavior of saturated DVE systems [16].

The constructive phase of GRASP, when applied to the considered problem, is based on the concept of *border avatars*. An avatar i is a border avatar if it is assigned to a server s and at least one of its neighbor avatars (as defined in [17]) is assigned to a server different from s. The *cardinality* of a border avatar i is defined as the number x of different servers where the neighbors of i are assigned to. The neighborhood of an avatar is defined by its AOI.

The assignment of the border avatars is critical, since it allows to obtain partitions with low levels of h_{asr}. Thus, the first step in the constructive phase is to determine the set of border avatars in the DVE system. GRASP will exclusively re-assign these avatars to provide the final partition. We have considered three different options or techniques for the constructive phase. Two of them are techniques that have been widely used in different heuristic methods due to its simplicity. The other one has been designed taking into account the specific features of the latency-aware partitioning problem. For all the considered techniques, an array of n elements containing the server where each avatar is assigned to is constructed. The constructive phase exclusively considers the border avatars, leaving the rest of the avatars assigned as they are in the partition provided by the FGALB method. In this way, we ensure that the system will be kept below the saturation point. Thus, all the border avatars are assigned to server -1 (unassigned) in this array, thus starting the constructive phase from an empty set of assignments. Prior to the application of the constructive phase, an integer number k between 1 and n is randomly generated. This number determines the starting point of the constructive phase, since all the techniques consists of sweeping the whole array starting at element k and finishing at element $k-1$, looking for border avatars (elements of the array containing -1 value). The considered techniques differ in the way that border avatars are assigned:

- *Random sample (RS) [21]:* In this technique, when a border avatar is found it is randomly assigned to any of the servers in the system.
- *Random sample* (RS*) [21]:* This technique is based on constructing a *list of border servers* for each of the border avatars. A server S belongs to the list of border servers for avatar a if exists any avatar b in the AOI of a that is assigned to server S. RS* randomly assigns each border avatar to one of the servers in the list of border servers. Thus, this option restricts the number of potential servers for the border avatars.
- *Greatest Group (GG) [24]:* For each border avatar b, this technique counts the number of neighbors (avatars currently in the AOI of b) assigned to each server. Avatar b is assigned to the same server where the greatest number of b neighbors is already assigned to. The purpose of this greedy assignment of border avatars is to minimize the final h_{asr} value for these avatars while assigning them in the most compact way as possible. The randomization of this technique consists in the generation of k. If the first border avatars found in this process have another border avatars among its neighbors, then these neighbors can be still unassigned. Therefore, the results provided by this technique depends on the element k where it starts sweeping the population. If all the neighbors of a given border avatar i are border avatars still not assigned, then one of these neighbors is randomly chosen and avatar i is assigned to the server to which this neighbor was assigned in the partition provided by the FGALB method.

In order to evaluate each one of these techniques (as well as the rest of the techniques shown in this paper), we have used the evaluation methodology proposed in

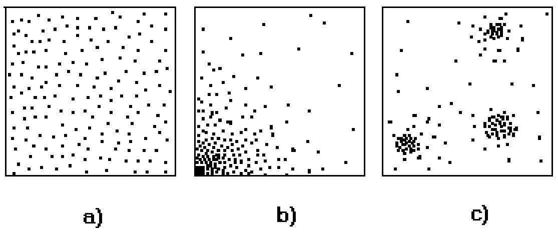


Fig. 2. Distributions of avatars in a 2-D virtual world: a) Uniform b) Skewed c) Clustered

[16]. Concretely, we have simulated the most common configurations in DVE configurations ranging from 250 avatars and 3 servers (denoted as MEDIUM1) to 700 avatars and 10 servers (denoted as MEDIUM2). However, due to space limitations, we present here the result for MEDIUM2 configuration. The results obtained for the rest of configurations were very similar. The values shown in each of the tables below are the average values obtained after 100 executions for different virtual worlds. In all the configurations tested, uniform, skewed and clustered distributions of avatars have been simulated. The reason for using different distributions is that they generate a different workload. Figure 2 shows an example of the considered avatar distributions in a 2-D virtual world. In this figure, the virtual world is a square and avatars are represented as black dots. In order to make this paper reproducible, the precise procedures to generate these populations are described in [11].

Table I shows the performance evaluation results for the constructive phase. Concretely, this table shows the results not for the entire GRASP procedure with different constructive algorithms, but the results of applying exclusively the constructive phase to the initial partition. This table shows in each row the f_{QoS} values provided by each one of the techniques considered in this phase for a MEDIUM2 configuration. For comparison purposes, the first row shows the values provided by FGALB method for the initial partition. In each column, this table shows the results for each one of the distributions of avatars in the virtual world.

TABLE I
f_{QoS} VALUES OBTAINED FOR THE CONSTRUCTIVE PHASE

	Skewed	Clustered	Uniform
FGALB	77050	78494	61461
RS	77050	78494	61461
RS*	70729	75382	55781
GG	65727	72879	53180

TABLE II
EXECUTION TIMES REQUIRED FOR OBTAINING THE PARTITIONS WHOSE f_{QoS} VALUES ARE SHOWN IN TABLE I

	Skewed	Clustered	Uniform
RS	1.65	2.07	0.73
RS*	1.62	2.02	0.72
GG	1.72	2.18	0.74

Table I shows that a random assignment does not improve at all the quality of the result in regard to the initial partition (FGALB). On the contrary, GG technique provides the best results for all the distributions considered, since it provides the lowest values of f_{QoS}.

Table II shows the execution time required for providing the partitions whose f_{QoS} values are shown in Table I. This table shows that for the uniform distribution of avatars, the execution times are shorter than for the non-uniform distributions. Nevertheless, this table does not show significant differences among the three techniques in any of the considered distributions of avatars. Taking into account the results in these two tables, GG technique seems to be the best choice for the constructive phase of GRASP.

B. Local Search Phase

In order to design a truly efficient GRASP for the latency-aware partitioning problem, we have also designed 3 different techniques for the local search phase. These techniques are the following ones:

- *Oriented Search (OS) [17]:* This technique was designed as a first GRASP approach for the considered problem. It sweeps all the border avatars in the current solution, randomly assigning each border avatar to a new server.
- *1-Flip (1F) [25]:* This technique randomly chooses any of the border avatars in the current solution and randomly assigns it to any of the servers in the system.
- *2-Swap (2S) [26]:* This technique randomly chooses two border avatars in the current solution and exchanges the server where they are assigned to.

In any of these methods, if the change performed in each iteration improves the quality of the resulting solution (decreases the f_{QoS} value) then that change is accepted. Otherwise, the new assignment is rejected and the previous assignment for that avatar is kept.

We have tried different finishing alternatives for the 1F and 2S techniques. As described in [27], in these techniques the algorithm tries to flip (swap) every variable (border avatars) before giving up. We tried first this alternative. However, in this problem the starting population is not a random but an accelerated one (the initial population is the one provided by FGALB method, since otherwise the system can be saturated and/or in a unsteady state). Therefore, we also implemented these techniques trying a single iteration before giving up (swapping / flipping a single avatar). Although the comparative results are not shown here for the sake of shortness, the implementation containing a single iteration provided better results (the ones shown below).

In order to study the efficiency of each of the proposed techniques, we have not evaluated them starting from the solution provided by the constructive phase. Instead, we have evaluated them starting directly from the initial solution provided by FGALB method.

Table III shows the evaluation results for the three local search techniques considered (not for a complete GRASP execution). For comparison purposes, it also contains the initial f_{QoS} values provided by FGALB method. This table shows that only OS can provide partitions with slightly better f_{QoS} values than the ones provided by FGALB. It is also clearly shown that 1F and 2S techniques shown practically identical results, with no improvement at all with respect to the initial values.

TABLE III
f_{QoS} VALUES PROVIDED BY THE TECHNIQUES CONSIDERED FOR THE LOCAL PHASE

	Skewed	Clustered	Uniform
FGALB	77050	78494	61461
OS	73106	74197	57268
1F*	76476	77421	60975
2S	76772	77279	60775

TABLE IV
EXECUTION TIMES REQUIRED TO PROVIDE THE PARTITIONS WHOSE f_{QoS} VALUES ARE SHOWN IN TABLE III

	Skewed	Clustered	Uniform
OS	1.62	2.01	0.73
1F*	0.24	0.27	0.19
2S	0.25	0.27	0.19

Table IV shows the execution times required to provide the partitions whose f_{QoS} values are shown in table III. In the case of the uniform distribution, it shows that the execution time required for the OS method is around three times longer than the ones required for any of the other two techniques. In the cases of non-uniform distributions, the execution times required for the OS technique are at least 8 times longer. The reason for these differences is that, unlike the other two techniques, OS changes the assignment for all the border avatars.

Considering the results shown in table IV and table III, we can state that the slight performance improvement achieved with OS technique is done at the cost of greatly increasing the execution time with respect to 1F or 2S techniques. None of the three techniques provides a significant improvement, and in particular 1F and 2S techniques do not provide any improvement at all. Although both techniques are capable to provide good results when applied to other problems [28], [29], they don't obtain significant reductions in the final values of the cost function when they are used for solving the latency-aware partitioning problem in DVE systems.

C. GRASP: Putting it Together

Taking into account the evaluation results shown in the previous subsections, we have tested (for the final design of GRASP) all the possible combinations of the constructive techniques described above with the techniques considered for the local search phase. Although the evaluation results of all these tests are not shown here for the sake of shortness, Table V shows the results for some of the possible combinations. This table shows the results for a complete GRASP when using each one of the combinations shown in each row. The first five rows show the results for the combinations that provided the best results, and the last four rows show the results for other combinations, just for comparison purposes.

Table V shows that the combination that provides the best values of the cost function f_{QoS} consists of using GG technique for the constructive phase and the OS technique for the local search phase. The differences between the best and the worst value in each column are significant, although they do not show a great percentage decreasing.

Table VI shows the execution times (in seconds) required for providing the partitions whose results are shown in table V. We can see that the execution times required by the first two combinations (GG+OS and RS*+OS) are around three times longer than the execution times required by the rest of the combinations.

TABLE V
f_{QoS} VALUES PROVIDED BY DIFFERENT COMBINATION OF CONSTRUCTIVE AND LOCAL SEARCH TECHNIQUES

	Skewed	Clustered	Uniform
GG+OS	63920	70132	50647
RS*+OS	68035	73980	54550
GG+1F	65352	72009	53301
GG+2S	65579	72124	53167
RS*+1F	71587	74856	55098
RS+OS	72015	75254	55264
RS+1F	73426	76284	56528
RS+2S	74043	76001	57717
RS*+2S	71984	75025	55102

TABLE VI
EXECUTION TIMES REQUIRED FOR OBTAINING THE RESULTS SHOWN IN TABLE V

	Skewed	Clustered	Uniform
GG+OS	9.65	12.31	3.14
RS*+OS	9.81	12.19	3.16
GG+1F	2.80	4.16	1.36
GG+2S	2.77	4.16	1.33
RS*+1F	2.55	3.99	1.35
RS+OS	9.78	12.55	3.21
RS+1F	2.75	4.21	1.29
RS+2S	2.83	4.19	1.40
RS*+2S	2.99	4.07	1.32

Concretely, the combination providing the lowest f_{QoS} value also requires one of the longest execution times. Taking into account the results shown in these two tables, it is not so clear which combination is the best one in order to solve the QoS problem.

D. Intensification

By definition, iterations in GRASP are highly independent, in such a way that the current iteration does not use the information provided by the previous iterations [21], [22], [30]. Although this feature makes GRASP an easily parallelizable method, it also limits the quality of the provided solutions. In order to improve the quality of the solutions, intensification strategies like Path-Relinking [27], [31] can be added.

In order to improve the quality of the solutions to the latency-aware partitioning problem in DVE systems, we have considered two intensification mechanisms. One of them, denoted as *M-GRASP (M stands for memory)*, consists of the same combinations of techniques used for the construction and local search phases but re-computing the set of border avatars after each iteration. As a consequence of both the constructive and the local search phases, the assignment of border avatars may have changed. Therefore, may be that some border avatars at the beginning of the previous iteration are not border avatars any more. Also, the same assignments of the previous iteration can make that some avatars become border avatars. In M-GRASP, at the beginning of each iteration the avatar population is swept again and a new set of border avatars is computed. In this way, the actions performed in the subsequent iterations are based on the actions taken in the current iteration (long-term memory is added to GRASP, and so the name of this strategy). The other intensification mechanism considered has been Path-Relinking. This mechanism was first applied within the GRASP framework to solve the problem of 2-layer straight line crossing minimization [32], and since then it has been widely used for other kind of problems [21], [27], [31]. Path-Relinking uses a pool of *elite solutions* made up of high-quality solutions found during the GRASP execution until the current iteration. When the current iteration finishes, Path-Relinking combines the solution provided in this iteration with a solution in the pool of elite solutions, and after that the pool is re-computed. The pool of elite solutions used in Path-Relinking provides the long-term memory needed by GRASP to improve the quality of the solution to the latency-aware partitioning problem. Although the results are not shown here for the sake of shortness, we have tested five different types of path-relinking found in the literature (up, down, new, none, random and both) [27], obtaining the best results with the ”up” type of path relinking.

TABLE VII
f_{QoS} VALUES PROVIDED BY THE INTENSIFICATION STRATEGIES

	Skewed	Clustered	Uniform
GG+OS+Path-R	60691	63210	44908
RS*+OS+Path-R	65210	66002	49187
GG+1F+Path-R	62781	63434	46952
GG+2S+Path-R	63215	63558	47040
GG+OS+M-GRASP	56326	60238	45372
RS*+OS+M-GRASP	65772	66755	47533
GG+1F+M-GRASP	62911	63189	45404
GG+2S+M-GRASP	63732	54316	45436

Table VII shows the performance evaluation results for the two intensification strategies considered, in terms of the cost function f_{QoS} values. This table shows the results for the intensification techniques described in this subsection when they are applied to some of the techniques shown in Table III.

Table VII shows that the combination of GG+OS+M-GRASP method provides the best f_{QoS} values. Although it provides similar values to the ones provided by the GG+2S+M-GRASP combination for the cases of a uniform and a clustered distribution of avatars, it provides better values for the case of a skewed distribution of avatars. It is worth mention that GG+OS+M-GRASP combination decreases the values provided by the path-relinking intensification method, particularly for non-uniform distributions of avatars. Comparing the results for the GG+OS+M-GRASP combination in this table with the results of the GG+OS combination in table V, we can see that M-GRASP is able to decrease the f_{QoS} values around a 15% in the case of a clustered distribution. That is, the M-GRASP intensification manages to provide even better solutions than the ones provided by the path-relinking intensification.

Table VIII shows the execution times required for

TABLE VIII
EXECUTION TIMES REQUIRED BY THE INTENSIFICATION STRATEGIES

	Skewed	Clustered	Uniform
GG+OS+Path-R	17.78	17.27	2.99
RS*+OS+Path-R	16.56	19.35	3.12
GG+1F+Path-R	8.35	8.64	2.58
GG+2S+Path-R	8.63	8.94	2.80
GG+OS+M-GRASP	8.73	10.81	1.13
RS*+OS+M-GRASP	9.91	12.26	1.42
GG+1F+M-GRASP	6.74	7.10	1.24
GG+2S+M-GRASP	6.72	7.11	1.11

providing the results shown in table VII. Unlike in table VI, the results in this table shows that GG+OS+M-GRASP combination requires execution times slightly longer than the ones required by GG+1F+M-GRASP or GG+2S+M-GRASP combinations, that are the combinations with the shortest execution times. Since path-relinking is a post-optimization method, the execution times shown in the upper rows of table VIII are longer than the ones shown in the lower rows, particularly for those combination involving OS local search phase.

As it could be expected, most of the combinations shown in table VIII increase the required execution times with regard to the corresponding values shown in table VI, since an intensification method is applied after executing the constructive and local search phases. However, GG+OS+M-GRASP is the only combination of techniques that is able to reduce the execution time with regard to the combination GG+OS. The reason for that behavior is that computing the new set of border avatars after each iteration makes the search to be tuned iteration by iteration. As a result, the number of border avatars as well as the cardinality of each border avatar are decreased in each iteration. Therefore, the execution time required for GG and OS by subsequent iterations decreases.

Taking into account the results shown in Tables VIII and VII, we can conclude that the best GRASP design for solving the considered problem seems to be the combination of GG+OS techniques with the M-GRASP intensification. Although this combination requires execution times higher than the ones required by other combinations, it provides much better values of the cost function than any other combination. Therefore, from this point up we will denote GG+OS+M-GRASP combination as M-GRASP technique.

IV. PERFORMANCE EVALUATION

In this section, we present the performance evaluation of M-GRASP when compared to other meta-heuristics that have been already applied to solve the latency-aware partitioning problem in DVE systems, like Simulated Annealing [28], [33] or Genetic Algorithms [34], [35]. Concretely, we have considered an implementation of Simulated Annealing (SA) [36], an implementation of Genetic Algorithms (GA) [37], and also an initial approximation of GRASP to the latency-aware partitioning problem [36]. Simulated Annealing (SA) is a stochastic meta-heuristic based on a thermodynamic theory. This theory establishes that the minimum energy state in a system can be found if the temperature decreasing is performed slowly enough [28]. SA is a heuristic search technique that always accepts better solutions than the current one, and also accepts worse solutions according to a probability system based on the system temperature. Genetic Algorithms (GA) consists of a search method based on the concept of evolution by natural selection [34]. GA starts from an initial population, made of P *chromosomes*, that evolves following certain rules, until reaching a convergence condition that maximizes a fitness function. Each iteration of the algorithm consists of generating a new population from the existing one by recombining or even mutating chromosomes. A chromosome can contain a *genotype* or particular solution of the problem and also a *phenotype* or additional information for tuning the behavior of the

TABLE IX
EVALUATION RESULTS PROVIDED BY DIFFERENT META-HEURISTICS

	Skewed	Clustered	Uniform
SA	65929	69857	54051
GA	62312	62306	52008
GRASP	64838	66256	56221
M-GRASP	54585	55396	44424

TABLE X
EXECUTION TIMES REQUIRED BY THE META-HEURISTICS SHOWN IN TABLE IX

	Skewed	Clustered	Uniform
SA	84.08	48.89	7.68
GA	18.43	23.40	9.07
GRASP	20.07	43.66	7.31
M-GRASP	8.73	10.81	1.13

algorithm. The main difference of M-GRASP with the initial GRASP approximation shown in [36] is that in the latter one all the iterations must be executed in order to obtain a solution. Additionally, in the initial GRASP iterations are not independent from each other, and therefore it cannot be parallelized, either.

Although for comparison purposes we have evaluated these heuristics methods in both MEDIUM1 and MEDIUM2 configurations, only the results obtained for MEDIUM2 configuration are shown in this section for the sake of shortness. Table IX shows the f_{QoS} values provided by each heuristic method for this DVE configuration.

Table IX shows that M-GRASP significantly increases the quality of the provided solutions. For example, the value provided by M-GRASP is around a 18% lower than the one provided by the SA method for the case of a skewed distribution of avatars. For the cases of clustered and uniform distributions, the percentage of decrease achieved by M-GRASP with respect to SA is around 20% and 18%, respectively.

Table X shows the execution times required for providing the values shown in table IX. The execution times required by M-GRASP method are far and away the shortest ones. They are even several times lower than the times required by other methods.

These results show that M-GRASP represents an efficient heuristic method for solving the latency-aware partitioning problem in DVE systems. On the one hand, it is capable of providing the best solutions, and on the other hand it requires only a fraction of the execution times required by other methods.

V. CONCLUSIONS

In this paper, we have proposed a GRASP design for solving the latency-aware partitioning problem in DVE systems. For both the construction and the local search phase, we have evaluated three different techniques. GG provides better results than other techniques when starting from an initial partition, requiring similar execution times. Similarly, the OS technique provides slightly better results than the other techniques considered for the local search phase when they start from the initial partition. However, OS techniques requires longer execution times than the other techniques considered. As a consequence, when these two techniques are combined, they provide relatively good solutions compared to the solutions provided by other combinations, but they require execution times several times longer than the times required by other combinations.

The addition of long-term memory to the combination of GG+OS, denoted as M-GRASP, greatly improves the behavior of this combination. The computing of the set of border avatars in each iteration greatly improves the performance of the search while even reducing the required execution time. As a result, even a post-optimization technique like path relinking is not able to increase the quality of the solutions provided by M-GRASP.

When compared to other meta-heuristics that have been also designed for solving the considered problem, M-GRASP provides the results with the highest quality, while it requires execution times that are only a fraction of the execution times required by the other methods.

Unlike other GRASP implementations, M-GRASP does not require the execution of the whole iterations in order to provide a solution. Therefore, it can be adjusted to fulfill any time constraint (obviously, at the cost of decreasing the quality of the provided solution). Thus, we can conclude that M-GRASP represents an efficient heuristic method for solving the latency-aware partitioning problem in DVE systems.

ACKNOWLEDGMENT

This work has been jointly supported by the Spanish MEC and European Commission FEDER funds under grants Consolider Ingenio-2010 CSD2006-00046 and TIN2006-15516-C04-04.

REFERENCES

[1] D. Miller and J. Thorpe, "Simnet: The advent of simulator networking," *Proceedings of the IEEE*, vol. 83, no. 8, pp. 1114–1123, 1995.

[2] J. S. Dias, R. Galli, A. C. Almeida, C. A. C. Belo, and J. M. Rebordao, "mworld: A multiuser 3d virtual environment," *IEEE Computer Graphics and Applications*, vol. 17, no. 2, pp. 55–65, 1997.

[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)*, November 1998.

[4] 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.

[5] J. H. P. Chim, M. Green, R. W. H. Lau, H. V. Leong, and A. Si, "On caching and prefetching of virtual objects in distributed virtual environments," in *MULTIMEDIA '98: Proceedings of the sixth ACM international conference on Multimedia*. New York, NY, USA: ACM, 1998, pp. 171–180.

[6] A. Chan, R. Lau, and A. Si, "A motion prediction method for mouse based navigation," in *Proceedings of Computer Graphics International (CGI'01)*, 2001, pp. 139–146.

[7] A. Chan, R. Lau, and B. Ng, "A hybrid motion prediction method for caching and prefetching in distributed virtual environments," in *Proceedings of ACM VRST 2001*, 2001, pp. 135–142.

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

[9] F. Li, R. Lau, and D. Andkilis, "Gameod: An internet based game on demand framework," in *Proceedings of ACM VRST 2004*, 2004, pp. 129–136.

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

[11] J. C. Lui and M. Chan, "An efficient partitioning algorithm for distributed virtual environment systems," *IEEE Trans. Parallel and Distributed Systems*, vol. 13, no. 3, pp. 193–211, 2002.

[12] R. W. H. Lau, B. Ng, A. Si, and F. Li, "Adaptive partitioning for multi-server distributed virtual environments," in *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*. New York, NY, USA: ACM, 2002, pp. 271–274.

[13] B. Ng, R. Lau, A. Si, and F. Li, "Multiserver support for large-scale distributed virtual environments," *Multimedia, IEEE Transactions on*, vol. 7, no. 6, pp. 1054–1065, Dec. 2005.

[14] J. Chim, R. Lau, H. Leong, and A. Si, "Cyberwalk: A web-based distributed virtual walkthrough environment," *IEEE Transactions on Multimedia*, vol. 5, no. 4, pp. 503–515, 2003.

[15] C. Greenhalgh, A. Bullock, E. Fr¿on, D. Llyod, and A. Steed, "Making networked virtual environments work," *Presence: Teleoperators and Virtual Environments*, vol. 10, no. 2, pp. 142–159, 2001.

[16] P. Morillo, J. M. Ordu¿, M. Fern¿dez, and J. Duato, "Improving the performance of distributed virtual environment systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 7, pp. 637–649, 2005.

[17] S. Rueda, P. Morillo, J. M. Ordu¿, and J. Duato, "A latency-aware partitioning method for distributed virtual environment systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 9, pp. 1215–1226, 2007.

[18] 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.

[19] M. Claypool, "The effect of latency on user performance in real-time strategy games," *Computer Networks*, vol. 49, no. 1, pp. 52–70, September 2005.

[20] 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.

[21] M. Resende and C. Ribeiro, "Greedy randomized adaptive

search procedures," in *Handbook of Metaheuristics*, ser. International Series in Operations Research & Management Science, F. Glover and G. Kochenberger, Eds. Springer-Kluwer Academic Publishers, 2003, vol. 57, ch. 8.

[22] P. Festa and M. Resende, "Grasp: An annotated bibliography," in *Essays and Surveys on Metaheuristics*, P. Hansen and C. Ribeiro, Eds. Kluwer Academic Publishers, 2002, pp. 325–367.

[23] P. Morillo, J. M. Ordu¿, M. Fern¿dez, and J. Duato, "A fine-grain method for solving the partitioning problem in distributed virtual environment systems," in *Proc. of Intl. Conf. on Parallel and Distributed Computing and Systems (PDCS'04)*, IASTED. ACTA Press, 2004, pp. 292–297, best paper award in the area of load balancing.

[24] P. Galinier and A. Hertz, "A survey of local search methods for graph coloring," *Comput. Oper. Res.*, vol. 33, no. 9, pp. 2547–2562, 2006.

[25] M. Resende and T. A. Feo, *A GRASP for Satisfiability*, ser. DIMACS Series on Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, 1996, vol. 26, pp. 499–520.

[26] G. Nemhauser and L. Wolsey, *Integer and Combinatorial Optimization*. Wiley-Interscience, 1999.

[27] M. G. C. Resende and R. F. Werneck, "A hybrid heuristic for the p-median problem," *Journal of Heuristics*, vol. 10, pp. 59–88, 2004.

[28] P. J. Laarhoven and E. Aarts, *Simulated Annealing: Theory and Applications*, ser. Mathematics and its Applications. Springer-Verlag, 1987, vol. 37.

[29] M. Yagiura and T. Ibaraki, "Efficient 2 and 3-flip neighborhood search algorithms for the max sat: Experimental evaluation," *Journal of Heuristics*, vol. 7, no. 5, pp. 423–442, 2001.

[30] F. A. Feo and M. G. Resende, "Greedy randomized adaptive search procedures," *Journal of Global Optimization*, vol. 6, pp. 109–133, 1995.

[31] M. Resende and C. Ribeiro, "A grasp with path-relinking for private virtual circuit routing," *Networks*, vol. 41, pp. 104–114, 2003.

[32] M. Laguna and R. Marti, "Grasp and path-relinking for 2-layer straight line crossing minimization," *INFORMS Journal on Computing*, vol. 11, pp. 44–52, 1999.

[33] C. Oliveira, D. Paolini, and P. Pardalos, "A randomized algorithm for minimizing user disturbance due to changes in cellular technology," in *Proc. of Int. Conf. on Computer, Communication and Control Technologies (CCCT'03)*, 2003, pp. 45–50.

[34] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. Ed. Willey, 1997.

[35] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, 1994.

[36] P. Morillo, J. M. Ordu¿, M. Fernandez, 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.

[37] S. Rueda, P. Morillo, J. M. Ordu¿, and J. Duato, "A sexual elitist genetic algorithm for providing qos in distributed virtual environment systems," in *Proc. of 2005 Int. Parallel and Distributed Processing Symposium Workshops (IPDPS' 2005)*. IEEE Computer Society, 2005.

PLACE
PHOTO
HERE

Michael Shell Biography text here.

John Doe Biography text here.

Jane Doe Biography text here.