GRASP and Grid Computing to Solve the Location Area Problem

Sónia M. Almeida-Luz
Dept. of Informatics Engineering,
Sch. of Tech. and Mgmt, Polytechnic Institute of Leiria
2400 Leiria, Portugal
e-mail: sluz@estg.ipleiria.pt

Manuel M. Rodríguez-Hermoso, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez
Dept. Technologies of Computers and Communications
Escuela Politécnica, University of Extremadura
10071 Caceres. Spain
e-mail: {mmrodri, mavega, jangomez, sanperez}@unex.es

***Abstract*—In this paper we present a new approach based on the GRASP (Greedy Randomized Adaptive Search Procedure) metaheuristic to solve the Location Area (LA) problem over a grid computing environment. All the experiments carried out to complete this study were executed in a real grid environment provided by a virtual organization of the European project EGEE. These experiments were divided into sequential and parallel executions with the intention of analyzing the behavior of the different variants of GRASP when applied to the LA problem. We have used four distinct test networks and also decided to compare the results obtained by this new approach with those achieved through other algorithms from our previous work and also by other authors. The experimental results show that this GRASP based approach is very encouraging because, with the grid computing, the execution time is much more reduced and the results obtained are very similar to those of other techniques proposed in the literature.**

Keywords-location area problem; location management; GRASP algorithm; evolutionary algorithms; grid computing

I. INTRODUCTION

In the last decade, the development of network infrastructures has been growing, principally those directed for personal communication networks (PCN) [1, 2], because they must support the increase of user services and communications. In order to these networks support the mobility of users and be able to find them, also when they change their location, it is essential to consider mobility management and more precisely location management (LM) when the network infrastructures are defined.

In this paper, a GRASP based algorithm is used to solve the Location Area (LA) problem with the objective of determining the best network partitioning, minimizing the mobility management costs involved. Therefore, we present a new approach to this problem using a grid computing environment with different variants of GRASP.

Our study was mainly divided into three parts: the first has the objective of determining the best value for the local search parameter used in the GRASP algorithm; the second corresponds to the sequential executions with the goal of achieving the GRASP variants that are most adequate to the LA problem; and the third includes the parallel experiments executed to define the best configuration of the model master/slave implemented.

The paper is organized as follows. Section II provides an overview of the Location Area (LA) problem and the involved costs. In section III, the GRASP algorithm is described. The grid computing environment is exposed in section IV. In section V, the experiments and results are presented and analyzed, with the intent of defining the most adequate variants of GRASP, as well as comparing the results of sequential and parallel executions. Finally, section VI includes the conclusions and future work.

II. THE LOCATION AREA PROBLEM

Currently, for the systems based on network of cells (e.g. mobile networks) it is important to track the location of users, even when they move around between different cells, without making or receiving calls, to make possible to route incoming calls regardless of their location.

The Location Areas (LA) scheme represents an important strategy of location management, which is used with the objective of reducing traffic on mobile networks, caused by paging messages and location updates in cellular network systems.

In the LA scheme, the network is partitioned into groups of cells and each group corresponds to a LA or region. It is possible to observe in Fig. 1, how is the configuration of a network with four LAs (light gray and dark gray LAs with five cells and black and white LAs with three cells). In this scheme, when a mobile terminal moves to a new LA, its location is updated, which means a location update is performed. When the user receives an incoming call, the

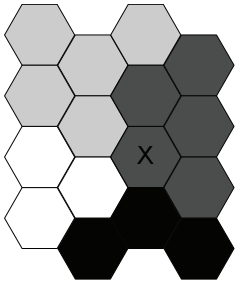


Figure 1. Configuration of a network with LAs.

978-1-4244-5612-3/09/$26.00 ©2009 IEEE

network must page all the cells of the new LA of the user, looking for its mobile terminal.

The LA problem can be defined as the problem of finding an optimal configuration of location areas, minimizing the location management cost. The location management cost normally is divided into two main parts: location update cost and location paging cost [3], [4].

A. Update Cost

The location update (LU) cost corresponds to the cost involved with the location updates performed by mobile terminals in the network, when they change their location to another LA. Because of that, the number of location updates is normally caused by the user movements in the network. This means that, when we calculate the update cost for a certain LA, we must consider the entire network and look for the flow of users.

If we consider the network of Fig. 2a, it is possible to see the total number of users who enter in the black LA. To calculate the location update cost for that LA, we must sum up those numbers of users that enter (from another LA) on each cell of the LA:

$$N_{LU} = 78 + 49 + 56 + 83 + 42 + 55 = 363. \quad (1)$$

B. Paging Cost

The location paging (P) cost is caused by the network when it tries to locate a user's mobile terminal, during the location inquiry. Normally the number of paging transactions is directly related to the number of incoming calls. The task of calculating the paging cost is simpler, because we only need to count the number of incoming calls in the selected LA and then multiply the value by the number of cells in the respective LA. Considering the incoming calls to the black LA shown in Fig. 2b, the calculus of paging cost is:

$$N_P = (47 + 45 + 58) \times 3 = 450. \quad (2)$$

C. Total Cost

The location management cost involves other parameters and components, but those are considered to be equal for all strategies [4]. Therefore, these other parameters do not influence the comparison of different strategies, and we will not consider them for the total cost. In conclusion, the combination of location update cost and location paging cost is sufficient to compare different strategy results.

The formula to calculate the total cost of location management [5, 6] is:

$$\text{Total\_Cost} = \beta \times N_{Lu} + N_P. \quad (3)$$

The total cost of location updates is given by N_{LU}, the total cost of paging transactions is given by N_P, and finally β is a ratio constant used in a location update relatively to a paging transaction in the network. The cost of each location update is considered to be much higher than the cost of each paging transaction, due to the complex process that must be executed for each location update performed, and also because most of the time a mobile user moves without making any call [4]. Due to all of that, the cost of a location update is normally considered to be 10 times greater than the cost of paging, that is, $\beta = 10$ [3].

For the black LA referred earlier, and presented in Fig. 2a and 2b, the total cost by (3) would be:

$$\text{Total\_Cost} = 10 \times 363 + 450 = 4080. \quad (4)$$

To calculate the total cost of the network, with the configuration defined, it would be necessary to make the calculus for each LA and then sum all the values and get the final total cost.

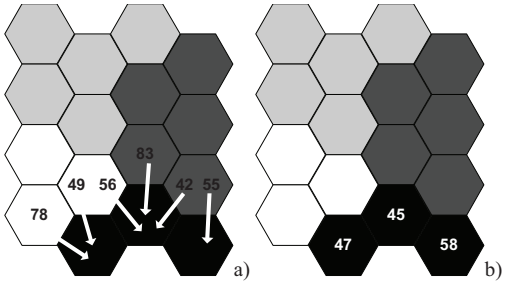


Figure 2. Calculus of: a) Location update cost; b) Paging cost.

III. GRASP Algorithm

The Greedy Randomized Adaptive Search Procedure (GRASP) [7, 8] is an evolutionary algorithm usually applied to combinatorial optimization problems, and it is the one that we will use on the solving of the LA problem.

The GRASP metaheuristic is a multi-start process, where each iteration consists of two phases [7, 8]: construction of a greedy solution and local search over that new solution. In Fig. 3 it is presented the outline of the GRASP algorithm, where i_{max} is a constant that indicates the number of the algorithm iterations, f^* represents the cost of the solution of the problem and x represents the solution. Considering the outline associated to the algorithm implementation, we may see that initially the solution has assigned an infinite cost, which will correspond to the worst possible. After that, the

GRASP Algorithm

```
Require: i_max
        f* ← ∞
        for i ≤ i_max do
                x ← greedyRandomizedMethod();
                x ← localSearchMethod(x);
                if (f(x) < f*) then
                        f* ←f(x);
                        x* ←x;
                end if
        end for
        return x*;
```

Figure 3. Pseudo-code for GRASP algorithm.

initial solution will improve during a predetermined number of iterations where, in each iteration will be included the first phase of creation of a greedy randomized solution, using the respective method, and a second phase of improving that solution with a local search method adapted to the LA problem. After these two steps we compare the cost of the new solution with the best until the moment and if the new one is better (this means the cost is lower), the solution is stored to the next iteration. Finally, concluding all the iterations we will obtain the solution with the best cost, which represents the best configuration of the network, by means of LAs and respective cells.

A. Greedy Randomized Method

There exist several variants of the GRASP algorithm [7] that are responsible for the specification of the method used to generate the greedy randomized solution, and principally, for the construction of the restricted candidate list (RCL).

Basically the RCL can be limited either by the number of elements (cardinality-based), which is the variant C, or their quality (value-based) that corresponds to the variant V. The variant RG is characterized by being, first random and latter greedy. Finally, the variant PG uses a perturbation factor to change the value of determined parameter [7, 8].

Besides these four variants of GRASP, we must also consider the bias function that modifies the mode of selection of the elements that are taken from the RCL. There exist several bias functions [7, 8], however we have used three types in our work, commonly referred as bias=0, bias=1 and bias=2, which respectively correspond to random, linear and exponential functions.

B. Local Search Method

The Local Search (LS) explores in a repetitive way the neighborhood of a solution, searching for a better solution. When we do not find a new solution that can improve the current, we say that this solution is locally optima.

The LS represents an important piece of GRASP because it is good to find locally optima solutions, normally in less time than other algorithms.

In our case, where the LS is applied as part of the GRASP algorithm, it has the objective of improving the initial solution getting a new one with lower cost. That can be achieved reorganizing each location area of the network by adding or subtracting one cell in each step of the improvement.

IV. GRID COMPUTING ENVIRONMENT

The grid environment corresponds to the application of several computers to a single problem, at the same time. It is a system of distributed computing used to share computational resources that are geographically dispersed, to solve problems of large scale, which otherwise would take too long to be solved.

The access to the grid resources was granted through the EELA (E-science grid facility for Europe and Latin America) [9] virtual organization that includes hundreds of working nodes. All the experiments have been executed with a grid provided by the EGEE (Enabling Grids for E-sciencE) European project [10].

The experiments of this work were performed on the grid in sequential and parallel modes. The sequential experiments were directly managed by the grid (launching many jobs by using simple scripts). For the parallel experiments we have followed a master/slave model, which we developed through the implementation of complex shell scripts.

V. EXPERIMENTS AND RESULTS

In this section we will describe all the experiments that have been performed and a respective explanation of the results obtained.

With the objective of test this GRASP based approach we decided to work with for networks of distinct sizes, respectively of 25 (5x5), 35 (5x7), 49 (7x7) and 63 (7x9) cells. These networks are available in [4, 6, 11].

In order to assure the statistical relevance of the results we have performed 30 independent runs for each parameters' combination in each experiment

As we have already said, all the experiments that will be analyzed in the following sections have been performed with the grid provided by the EGEE project [10]. For the execution of all the jobs we use the computing elements that had less workload in their queues, respecting the grid rules. Due to space reasons, we only present the most important results and conclusions of all these experiments.

A. Adjustment of Local Search (LS) Parameter

The goal of the first group of experiments was determining the value of the parameter corresponding to the number of local search executions.

This parameter and the number of iterations are the most important ones, to reach the best solutions with the GRASP based approach. We had predefined the value of one hundred iterations for the GRASP implementation, taking in consideration the computation time (we considered the possibility of a thousand iterations, but it was not fixed because the time of execution was excessive).

The experiments that we will present in the following subsections were executed with a number of iterations equal to one hundred and a bias equal to zero (random bias). For each of the GRASP variants we elected the following configuration: for the variant C and RG the number of elements (k) will be of 3; for the variant V the alpha (α) value will be of 0.5; and for the variant PG the perturbation probability will be of 30α . With the four test networks we tested the following configurations for the number of LS: 1, 2, 4, 6, 8, 10 and 12.

1) Network of 25 (5x5) cells: Using the network of 25 cells and analyzing all the cost results, we concluded that with the value of 12 LS we obtained the best solution, the one with the lowest costs; but also with the value of 10 LS we obtained a very similar cost and with less computational time. Analyzing the results by variant, we noticed that we obtained the lowest cost with the type V.

2) Network of 35 (5x7) cells: Considering the average results obtained with the network of 35 cells, we also

concluded that the ideal number of LS was 12, however the GRASP variant that accomplished the lowest costs was RG.

3) Network of 49 (7x7) cells: With the network of 49 cells and the respective solutions achieved, we noticed that, like in the 5x7 network, the most adequate number of LS was 12 and the variant that reached lower costs was RG. Relatively to the number of LS we also observed that from 8 LS the cost variation of the solutions was very small.

4) Network of 63 (7x9) cells: Relatively to the test network of 63 cells we concluded again that with the value 12 of LS and the variant RG the best solution was obtained. Anyway, the results with 10 LS were similar.

5) Comparison of results: Analyzing and comparing the results obtained, we concluded that with 12 LS we achieved the lowest costs. Nevertheless, we must consider the execution time over each network, and with 12 LS that time usually surpassed the one assigned, per default, by the grid. Because of that we decided to fix the number of LS to 10 for all the following experiments, since with this value the execution time was significantly lower and the results were similar. As for the GRASP variants, we observed that the RG variant accomplished the best costs for almost all the networks.

B. Sequential Executions

The sequential experiments have been executed with the objective of determining what of the GRASP variants could obtain the best results, and the exact configuration of that variant.

We have tried to adjust the executions to the default time of the grid proxy (twelve hours) but, for the bigger networks it was necessary to increase this time to twenty four hours. However, we must say that in this execution time are also included the tasks of submitting the jobs to the grid and collecting the results returned.

The experiments have been done with all the test networks, considering the different GRASP variants and also the three values 0, 1 and 2, of bias function that correspond, respectively, to the random, linear and exponential types. The bias function cannot be applied to the variant PG, just to the variants C, V and RG. To execute the different experiments each variant was configured as follows: in the variants C and RG the number of elements (k) is between 1 and 6; in the variant V the alpha (α) value takes the values 0, 0.25, 0.5, 0.75 and 1; and in the variant PG the perturbation probability takes the values of 10, 20, 30, 40, 50 and 60 percent.

1) Analysis over sequential results: With the sequential experiments we had the objective of determining the 5 best configurations relatively to the GRASP variants, their parameters, and bias function, when applicable. So we decided to elect the best configuration for each GRASP variant and a fifth configuration that would represent the best configuration, in global, for all the networks and variants. Considering this, and knowing that we used these five configurations to proceed with the parallel experiments, presented in the subsection V.C. *Parallel Executions*, the configurations achieved were:

a) Conf_0: configuration associated with the variant C, with 100 iterations, 10 local searches, k=4 and bias=0.

b) Conf_1: configuration associated with the variant V, with 100 iterations, 10 local searches, α=0.5 and bias=0.

c) Conf_2: configuration associated with the variant RG, with 100 iterations, 10 local searches, k=5 and bias=0.

d) Conf_3: configuration associated with the variant PG, with 100 iterations, 10 local searches and probability of perturbation equal to 60α .

e) Conf_4: this is the global configuration, associated with the variant RG, with 100 iterations, 10 local searches, k=4 and bias=1.

2) Comparing our results with other applied algorithms: After obtaining the results of the sequential experiments we decided to compare them with the ones achieved in a previous work using a Differential Evolution based approach [6, 11] and also with the results presented by Taheri and Zomaya in [12], results obtained with very different algorithms as GA (Genetic Algorithm), HNN (Hopfield Neural Network), SA (Simulated Annealing) and GA-HNNx (different combinations of Genetic Algorithm and Hopfield Neural Network, see [12]). Observing Table I we may say that our GRASP approach obtains values very close to those accomplished with the other algorithms.

C. Parallel Executions

In this section we present the results obtained with the parallel experiments.

The executions with the parallel team follow a schema of master/slave, implemented through shell scripts, where each slave will correspond to a different node in the grid. Considering this, the team will be compound with a master node, responsible for creating, launching, receiving and analyzing the jobs, and a set of slaves that will realize the executions with the five configurations obtained in the sequential experiments. To execute the parallel experiments it was necessary to create a grid proxy with forty eight hours.

It is also important to highlight that in each experiment we tested the configuration for the number of synchronizations (between the master and slave nodes) with the values 1, 2, 3, 4 and 5.

We must take in consideration that the assignment of the configurations must be proportional to the number of slave nodes that compound the team. For example, if we had 5 slaves, each slave contained a different configuration file, if

TABLE I. COMPARISON OF NETWORK COSTS ACHIEVED BY DIFFERENT ALGORITHMS

Test Net.	Algorithm							
	DE	***GA***	***HNN***	***SA***	***GA-HNN1***	***GA-HNN2***	***GA-HNN3***	***GRASP***
5x5	26990	28299	27249	26990	26990	26990	26990	27056
5x7	39859	40085	39832	42750	40117	39832	39832	40611
7x7	61037	61938	63516	60694	62916	62253	60696	62900
7x9	89973	90318	92493	90506	92659	91916	91819	92312

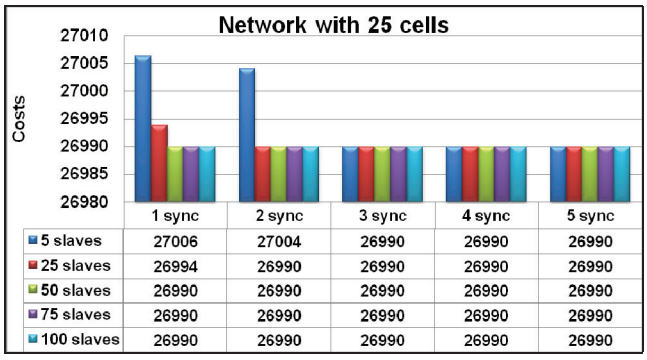
we had 25 slaves, we would have 5 slaves with equal configurations (but, of course, managing different solutions), if we had 50 slaves we would have 10 slaves with equal configurations, and so on.

1) Executions with 5x5 network: In Fig. 4 we present the average of the results achieved, when we used the network of 25 cells and respectively the configurations with 5, 25, 50, 75 and 100 slaves. Observing the results we concluded that, for this network, we obtained the lowest costs with the majority of configurations and that from 50 slaves, and for all the number of synchronizations we always obtained the best solution.

2) Executions with 5x7 network: Using the network of 35 cells we observed (see Fig. 5) that it was with the configuration of 50 slaves and 3 synchronizations, with the master node, that we achieved the solution with the lowest cost.

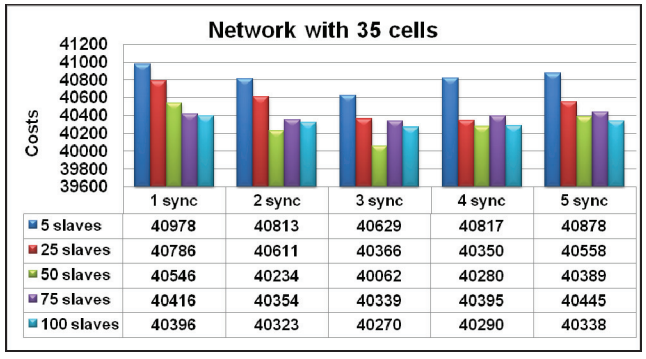
3) Executions with 7x7 network: The average results obtained with the network of 49 cells are presented in Fig. 6. Here we concluded that it was with 75 slaves and 4 synchronizations, with the master, that the best solutions were found.

4) Executions with 7x9 network: Considering the average results shown in Fig. 7, for the network of 63 cells, once again (like for the 7x7 network) we observed that was with the configuration of 75 slaves and 4 synchronizations, that the best solutions (the lowest costs) were achieved.



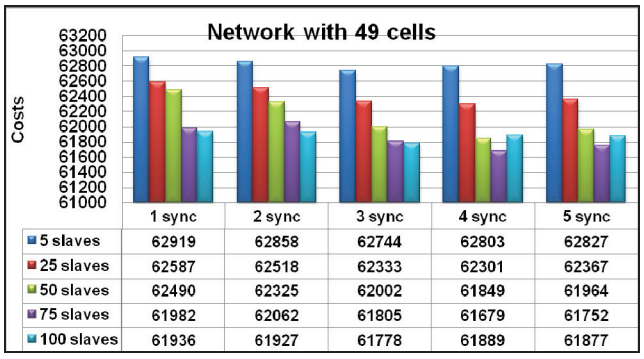
	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	27006	27004	26990	26990	26990
25 slaves	26994	26990	26990	26990	26990
50 slaves	26990	26990	26990	26990	26990
75 slaves	26990	26990	26990	26990	26990
100 slaves	26990	26990	26990	26990	26990

Figure 4. Parallel executions with 5x5 network.



	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	40978	40813	40629	40817	40878
25 slaves	40786	40611	40366	40350	40558
50 slaves	40546	40234	40062	40280	40389
75 slaves	40416	40354	40339	40395	40445
100 slaves	40396	40323	40270	40290	40338

Figure 5. Parallel executions with 5x7 network.

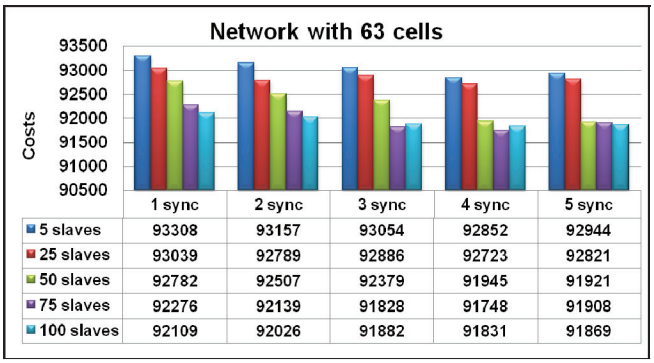


	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	62919	62858	62744	62803	62827
25 slaves	62587	62518	62333	62301	62367
50 slaves	62490	62325	62002	61849	61964
75 slaves	61982	62062	61805	61679	61752
100 slaves	61936	61927	61778	61889	61877

Figure 6. Parallel executions with 7x7 network.

5) Analysis and conclusions over the parallel executions: After analyzing all the results obtained and the partial conclusions, relatively to each test network, we concluded that the best configuration for the parallel team is 75 for the number of slave nodes and 4 for the number of synchronizations. We took this decision because it was with 75 slave nodes that we achieved the majority of the best results. If we consider the values obtained with 100 slaves, they were generally equal or even worse. Relatively to the number of synchronizations we selected 4, because for the bigger networks, this is the most adequate.

6) Participation of the configurations in the parallel team: In this section we present the participation of each GRASP configuration in the parallel team. For this study we used the results obtained with the best configuration of the parallel team, i.e. 75 slave nodes and 4 synchronizations. Fig. 8 shows, respectively for the networks of 25, 35, 49 and 63 cells, the participation percentage of each configuration in the parallel team. Analyzing the results we noticed that the participation of the variant RG in the team, and for all the test networks, is clearly superior. This is reasonable, because it is the variant with the best results in all our experiments. Anyway, we also concluded that the five configurations elected with the sequential executions were very useful in the parallel team. In fact, all of them participate in the results (as shown in Fig. 8) and help to improve the final results, as we will see in the next subsection.



	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	93308	93157	93054	92852	92944
25 slaves	93039	92789	92886	92723	92821
50 slaves	92782	92507	92379	91945	91921
75 slaves	92276	92139	91828	91748	91908
100 slaves	92109	92026	91882	91831	91869

Figure 7. Parallel executions with 7x9 network.

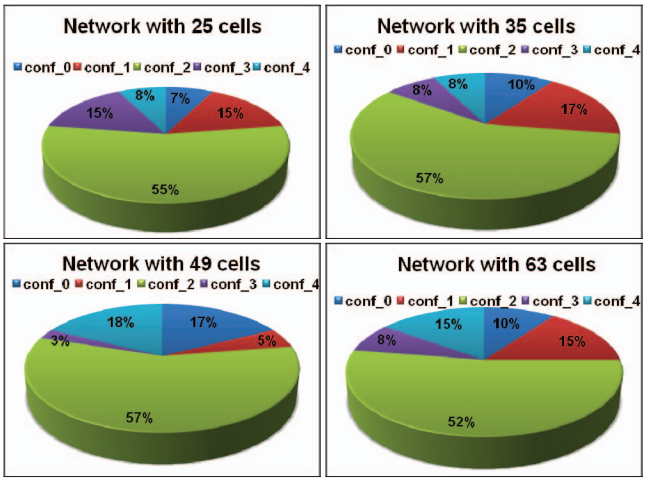


Figure 8. Participation of GRASP configurations in the parallel team.

D. *Comparison between Sequential and Parallel Results*

Finally we decided to compare the results obtained with our parallel implementation of GRASP relatively to its best sequential version. Observing Table II, we noticed that the results of the parallel GRASP always surpass those accomplished with sequential GRASP, showing the importance of the parallel team.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we present a novel approach based on the GRASP algorithm for solving the LA problem. The implementation of this approach took in consideration that all the experiments would be executed using grid computing.

Relatively to the grid technology, we may say that the results obtained are very satisfactory, because with it we can solve complex problems which, in other way would be conditioned by the time of execution. If we sum all the execution time of all the GRASP experiments we get a total of 13727 hours, or more specifically 572 days (about 1 year and 7 months). This leads us with the real notion of the importance of the grid and parallel computing.

Considering the results obtained, we defined the best configuration of the GRASP based approach when applied to the LA problem. The results achieved with the configuration of sequential GRASP are very similar of those obtained by other authors. The parallel version of GRASP always surpasses the sequential results and the best configuration is compound by 75 slave nodes and 4 synchronizations with the master node.

Finally, taking in consideration the distribution of the five distinct configurations of GRASP, inside the parallel team, we concluded that the most adequate variant is RG.

As future work we want to develop the same study with other types of evolutionary algorithms. The possibility of using these evolutionary algorithms to develop parallel applications and analyzing the collaboration among them is also matter of study.

TABLE II. COMPARISON OF SEQUENTIAL AND PARALLEL RESULTS

Test Network	GRASP Results	
	Sequential	*Parallel*
5x5	27056	26990
5x7	40611	40062
7x7	62900	61679
7x9	92312	91748

ACKNOWLEDGMENT

This work was partially funded by the Spanish Ministry of Science and Innovation and FEDER under the contract TIN2008-06491-C04-04 (the M* project). Thanks also to the CIIC and the Polytechnic Institute of Leiria, for the economic support offered to Sónia M. Almeida-Luz to make this research.

REFERENCES

[1] K. Pahlavan, A.H. Levesque: Wireless Information Networks. John Wiley & Sons, 1995.

[2] GSM World, http://www.gsmworld.com/newsroom/index.htm (2009)

[3] P.R.L. Gondim, "Genetic Algorithms and the Location Area Partitioning Problem in Cellular Networks", Proc. 46th IEEE Vehicular Technology Conf. Mobile Technology for the Human Race, 1996, vol. 3, pp. 1835—1838.

[4] J. Taheri, A. Y. Zomaya, "A Genetic Algorithm for Finding Optimal Location Area Configurations for Mobility Management", Proc. 30th Anniversary of the IEEE Conference on Local Computer Networks (LCN), Nov. 2005, pp. 568-577.

[5] R Subrata, A.Y. Zomaya, "Evolving Cellular Automata for Location Management in Mobile Computing Networks," IEEE Transactions on Parallel and Distributed Systems, vol. 14, Jan. 2003, pp. 13-26, doi:10.1109/TPDS.2003.1167367.

[6] S.M. Almeida-Luz, M.A. Vega-Rodríguez, J.A. Gómez-Pulido, J.M. Sánchez-Pérez, "Solving the Location Area Problem by Using Differential Evolution", Journal of Communications Software and Systems, vol. 4 (2), Jun. 2008, pp. 131-141.

[7] M. Resende, C. Ribeiro, "Greedy Randomized Adaptive Search Procedures", AT&T Labs Research Tech. Report, 2001, pp. 1–27.

[8] L. S. Pitsoulis, M. G.C. Resende, "Greedy Randomized Adaptive Search Procedures", Handbook of Applied Optimization, Oxford University Press, 2002, pp. 168-182.

[9] EELA Web, http://www.eu-eela.eu (2009)

[10] EGEE Web, http://www.eu-egee.org (2009)

[11] S.M. Almeida-Luz, M.A. Vega-Rodríguez, J.A. Gómez-Pulido, J.M. Sánchez-Pérez, "Defining the Best Parameters in a Differential Evolution Algorithm for Location Area Problem in Mobile Networks", Proc. EPIA 2007 – 13th Portuguese Conference on New Trends in Artificial Intelligence; (Eds). APPIA, Dec. 2007, ISBN: 978-98-995-6180-9; pp. 219-230.

[12] J. Taheri, A.Y. Zomaya, "A Combined Genetic-Neural Algorithm for Mobility Management". Journal of Mathematical Modelling and Algorithms, vol. 6 (3), Mar. 2007, pp. 481-507, doi: 10.1007/s10852-007-9066-5.