Solving the Location Area Problem by Using Differential Evolution

Sónia M. Almeida-Luz, Miguel A.Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez

Abstract: **In mobile networks, one of the hard tasks is to determine the best partitioning in the Location Area problem, but it is also an important strategy to try to reduce all the involved management costs. In this paper we present a new approach to solve the location management problem based on the Location Area partitioning, as a cost optimization problem. We use a Differential Evolution based algorithm to find the best configuration to the Location Areas in a mobile network. We try to find the best values for the Differential Evolution parameters as well as define the scheme that enables us to obtain better results, when compared to classical strategies and to other authors' results. To obtain the best solution we develop four distinct experiments, each one applied to one Differential Evolution parameter. This is a new approach to this problem that has given us good results.**

Index terms**: Differential Evolution, Location Area problem, location management, mobile networks**

I. INTRODUCTION

Personal communication networks (PCN) [1] have a digital communication system that enables any user to make or receive calls from any location and at any time of the day. For that, the system must support the mobility of the users as well as be able to find the users even when they change their location.

Because communication networks must support a big number of users, and their applications, as well as a wide range of data transfers, the task of designing the infrastructure of these networks must consider, as a very important point, the mobility management. Mobility management involves the process of location management that enables the mobile network to find the current location of the mobile terminal in order to make or receive calls, and the process of handoff management that enables the mobile network to locate roaming mobile terminals.

We are principally concerned about the mobility management because their requests normally occur when a mobile terminal changes its location or when the quality of the received signal becomes deteriorated, so this process becomes even more important for the current and future generations of mobile networks.

Location management involves two elementary operations: location update and location inquiry (or terminal paging). The location update corresponds to the notification of current location, performed by mobile terminals when they change their location in the network. The location inquiry is the operation of determining the location of the mobile terminal, which is executed by the network when it tries to direct an incoming call to the user.

Location management strategies may be divided into two main categories: static and dynamic schemes. The static schemes consider the same behaviour of the network for all users, while the dynamic schemes consider different network topologies for different users based on the individual user's call and mobility patterns. Unlike dynamic schemes that are more complex, static schemes are more common in the actual mobile networks, because they require less computational effort. A survey of different dynamic techniques based on users' behaviour such as timer-based, distance-based, movement-based (among others) may be seen in [2]. As static techniques, the most common ones are always-update, never-update, and location area schemes [2], among others.

Always-update and never-update are the two simple location management strategies. In the always-update strategy, each mobile terminal performs a location update every time it enters on a new cell, but no search operation would be required for incoming calls, because it is considered that all cells have different location areas. For the never-update strategy no location update is performed but, when there is an incoming call, a search operation is executed with the objective of finding the corresponding user; because all cells are considered as belonging to the same location area. Normally these two strategies correspond to the extremes of location management strategies and for that, most of existing network systems use a combination of them. One of the most common location management strategies in the existing systems is the Location Area scheme that is presented with more detail in the next section.

There exist several authors working with the location area scheme and applying computationally efficient algorithms like genetic algorithms [3 - 5], simulated annealing [5, 6], taboo search [5] and clustering techniques [7] (among others).

In this paper, a Differential Evolution based algorithm is used to find the best configuration for the location area scheme in a mobile network. Therefore, we present a new

Manuscript received March, 2008 and revised July, 2008.

This Paper was presented as part at the International Conference on Software, telecommunications and Computer Networks (SoftCOM) 2007.

Sónia M. Almeida-Luz is with Polytechnic Institute of Leiria, School of Technology and Management, Leiria, Portugal (e-mail: sluz@estg.ipleiria.pt)

Miguel A.Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez are with University of Extremadura. Dept. Technologies of Computers and Communications Escuela Politécnica, Spain (e-mail: {mavega, jangomez,sanperez}@unex.es)

approach to this problem. Section II provides an overview of the location area problem and the involved costs. In section III, the Differential Evolution based algorithm is described, as well as its parameters and different possible schemes. In section IV, the experimental results of the four specific experiments are presented and an analysis over the obtained results is done with the intent of defining the best Differential Evolution parameters configuration. Finally, section V includes conclusions and future work.

II. Location Area Problem

In cellular network systems it is very important to keep track of the location of the users, even when they move around without making or receiving calls, so as to consequently, be able to route calls to the users regardless of their location.

Location Areas (LA) scheme corresponds to an important strategy of location management, that is used with the objective of reducing signalling traffic 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 region, or more precisely to a LA, as we can see in Fig. 1, where we have a network with four LAs and each with four 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 network must page all the cells of the new LA of the user, looking for its mobile terminal.

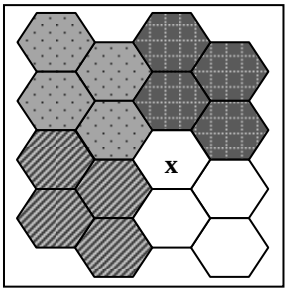


Fig. 1. Network Partitioning into Location Areas

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 in two main parts: location update cost and location paging cost [3, 4].

A. Location 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 white 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 and the calculus is:

$$NLU = 108 + 41 + 42 + 73 + 84 + 63 + 58 = 469 \quad (1)$$

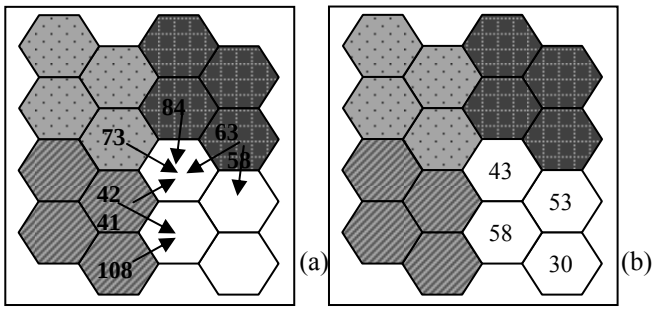


Fig. 2. a) Entering flow of users b) Incoming calls to the white LA

B. Location 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, and 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 white LA shown in Fig. 2b, the calculus of paging cost is:

$$NP = (43 + 53 + 58 + 30) \times 4 = 736 \quad (2)$$

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 [8] is:

$$Cost = \beta \times NLU + NP \quad (3)$$

The total cost of location updates is given by NLU, the total cost of paging transactions is given by NP, 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 white LA referred earlier, and presented in Fig. 2a and 2b, the total cost by (3) would be:

$$Cost = 10 \times 469 + 736 = 5426 \quad (4)$$

To calculate the total cost of the network with the configuration defined, which means with four LAs, would be

necessary to make the calculus for each LA and then sum all the values and get the final total cost.

III. Differential Evolution Algorithm

The Differential Evolution (DE) is a population-based algorithm, created by Ken Price and Rainer Storn [9], whose main objective is functions optimization. It is one strategy based on evolutionary algorithms with some specific characteristics.

The DE algorithm's main strategy is to generate new individuals by calculating vector differences between other randomly-selected individuals of the population. This algorithm uses four important parameters: population size, mutation, crossover and selection operators; there are different variants.

A. Initial Population

Like other Evolutionary Algorithms, DE works with a population of NI individuals (candidate solutions) and this number never changes during the optimization process. Normally the initial population is randomly generated and the population will be improved by the algorithm iteratively, through the mutation, crossover and selection operators (in [10] is possible to see more details about the DE flowchart).

B. Mutation Operator

The mutant operator *F* is a scaling factor that controls the amplitude of the differential variation of those random individuals used in the calculi.

With this operator DE generates a mutant individual ($I_{i,\,g+1}$), by adding a weighted difference of two population individuals, to a third individual using the equation (5):

$$I_{i,g+1} = X_{1,g} + F\left(X_{2,g} - X_{3,g}\right) \tag{5}$$

The value of F must be greater than zero and will control the magnitude of the differential variation of ($X_{2,\,g}$ - $X_{3,\,g}$). The individuals X_1, X_2 and X_3 are randomly selected and different among them. The g means the actual generation and g+1 means the next generation. DE uses a weighted difference between individuals to perturb the population in each generation, instead of randomly define the quantity of perturbations in the generation of a new individual as the most of other Evolutionary Algorithms do.

C. Crossover Operator

Crossover operator *Cr* is a value between zero and one, which is used to increase the diversity of mutant individuals. This constant represents the probability of trial individual inherits parameter values from the mutant individual.

Mutant individual and target individual are subjected to crossover to generate the trial individual ($T_{\,i,\,g+1}$), as displayed in the following equation (6):

$$T_{ji,g+1} = \begin{cases} I_{ji,g+1} & \text{if } rnj \le Cr \\ X_{ji,g} & \text{otherwise} \end{cases} \tag{6}$$

where j= 1, 2, … G. G corresponds to the number of genes of an individual and rn corresponds to the random value generated.

D. Selection Operator

Selection has the purpose of comparing the trial individual (offspring) produced by the crossover operator with the target individual (parent) and it determines the one that will be part of next generation. If a trial individual has a smaller cost function value it is copied to the next generation, otherwise it is the target individual that passes to the next generation, as it is possible to see in equation (7):

$$\begin{aligned} &\text{if} && f\left(T_{i,g+1}\right) \le f\left(X_{i,g}\right), \quad \text{set } X_{i,g+1} = T_{i,g+1} \\ &\text{Otherwise} && X_{i,g+1} = X_{i,g} \end{aligned} \tag{7}$$

E. DE Schemes

Price and Storn [9] have suggested 10 different schemes (those are presented in Table I) for DE. These schemes are classified based on notation DE/x/y/z, where x specifies the vector to be mutated, y corresponds to the number of difference vectors used in mutation of x (normally 1 or 2) and z represents the crossover scheme. The vector x may be chosen randomly ('rand') or as the best of current population ('best'), and z may be binomial ('bin') or exponential ('exp') depending of the type of crossover used.

TABLE I
DE Schemes

Nº	Scheme	Mutant vector generation
1	DE/best/1/exp	*xbest + F(xr1 − xr2)*
2	DE/rand/1/exp	*xr3 + F(xr1 − xr2)*
3	DE/randtobest/1/exp	*xr3 + F1(xbest − xr3) + F2(xr1 − xr2)*
4	DE/best/2/exp	*xbest + F(xr1 + xr2 − xr3 − xr4)*
5	DE/rand/2/exp	*xr5 + F(xr1 + xr2 − xr3 − xr4)*
6	DE/best/1/bin	*xbest + F(xr1 − xr2)*
7	DE/rand/1/bin	*xr3 + F(xr1 − xr2)*
8	DE/randtobest/1/bin	*xr3 + F1(xbest − xr3) + F2(xr1 − xr2)*
9	DE/best/2/bin	*xbest + F(xr1 + xr2 − xr3 − xr4)*
10	DE/rand/2/bin	*xr5 + F(xr1 + xr2 − xr3 − xr4)*

IV. Experimental Results

In this section, we detail the source and preparation of the test networks, subsequently we explain the most relevant decisions and choices made in our algorithm implementation,

then we expose our different experiments and finally we present our results.

A. Test Networks Generation

There are several studies about other approaches for the LA problem, but unfortunately, most of them do not present the network data used for their implementation.

TABLE II
TEST NETWORK 5X5 ATTRIBUTES

No	UpP	CAr	Neighbors
0	129	50	(0:1,70) (1:5,46)
1	279	73	(0:0,76) (1:2,41) (2:5,31) (3:6,69) (4:7,55)
2	100	44	(0:1,29) (1:3,35) (2:7,22)
3	265	52	(0:2,31) (1:4,61) (2:7,63) (3:8,73) (4:9,27)
4	120	73	(0:3,63) (1:9,50)
5	202	52	(0:0,42) (1:1,29) (2:6,66) (3:10,59)
6	341	44	(0:1,77) (1:5,60) (2:7,32) (3:10,22) (4:11,63) (5:12,74)
7	284	34	(0:1,66) (1:2,19) (2:3,52) (3:6,38) (4:8,33) (5:12,65)
8	347	46	(0:3,70) (1:7,42) (2:9,60) (3:12,79) (4:13,61) (5:14,25)
9	199	52	(0:3,34) (1:4,44) (2:8,72) (3:14,45)
10	167	69	(0:5,51) (1:6,27) (2:11,29) (3:15,46)
11	327	41	(0:6,54) (1:10,37) (2:12,66) (3:15,26) (4:16,85) (5:17,47)
12	454	84	(0:6,83) (1:7,61) (2:8,71) (3:11,77) (4:13,51) (5:17,101)
13	336	55	(0:8,68) (1:12,65) (2:14,40) (3:17,44) (4:18,76) (5:19,29)
14	151	69	(0:8,20) (1:9,45) (2:13,33) (3:19,34)
15	158	52	(0:10,39) (1:11,32) (2:16,29) (3:20,42)
16	365	92	(0:11,83) (1:15,42) (2:17,83) (3:20,47) (4:21,61) (5:22,43)
17	401	56	(0:11,37) (1:12,96) (2:13,49) (3:16,79) (4:18,76) (5:22,49)
18	364	80	(0:13,98) (1:17,71) (2:19,25) (3:22,46) (4:23,59) (5:24,53)
19	135	51	(0:13,34) (1:14,30) (2:18,21) (3:24,36)
20	124	63	(0:15,34) (1:16,60) (2:21,24)
21	150	82	(0:16,61) (1:20,25) (2:22,57)
22	253	59	(0:16,41) (1:17,46) (2:18,34) (3:21,50) (4:23,68)
23	159	52	(0:18,71) (1:22,49) (2:24,33)
24	138	59	(0:18,72) (1:19,40) (2:23,20)

TABLE III
TEST NETWORK 5X7 ATTRIBUTES

No	UpP	CAr	Neighbors
0	115	43	(0:1,75) (1:7,36)
1	315	46	(0:0,61) (1:2,43) (2:7,33) (3:8,61) (4:9,112)
2	161	59	(0:1,31) (1:3,69) (2:9,50)
3	229	43	(0:2,47) (1:4,29) (2:9,43) (3:10,41) (4:11,57)
4	69	34	(0:3,23) (1:5,15) (2:11,24)
5	115	46	(0:4,15) (1:6,18) (2:11,32) (3:12,20) (4:13,16)
6	35	33	(0:5,22) (1:13,5)
7	213	71	(0:0,41) (1:1,33) (2:8,86) (3:14,44)
8	368	51	(0:1,63) (1:7,73) (2:9,54) (3:14,21) (4:15,71) (5:16,73)
9	475	95	(0:1,96) (1:2,56) (2:3,52) (3:8,52) (4:10,122) (5:16,84)
10	420	54	(0:3,23) (1:9,115) (2:11,37) (3:16,63) (4:17,58) (5:18,109)
11	248	41	(0:3,54) (1:4,18) (2:5,29) (3:10,44) (4:12,45) (5:18,42)
12	218	46	(0:5,24) (1:11,49) (2:13,17) (3:18,37) (4:19,25) (5:20,54)
13	54	35	(0:5,17) (1:6,8) (2:12,9) (3:20,7)
14	142	59	(0:7,47) (1:8,18) (2:15,26) (3:21,34)
15	311	45	(0:8,72) (1:14,29) (2:16,42) (3:21,32) (4:22,81) (5:23,41)
16	403	40	(0:8,76) (1:9,80) (2:10,39) (3:15,49) (4:17,76) (5:23,69)
17	431	53	(0:10,55) (1:16,74) (2:18,71) (3:23,70) (4:24,42) (5:25,105)
18	450	49	(0:10,113) (1:11,38) (2:12,49) (3:17,66) (4:19,118) (5:25,53)
19	461	92	(0:12,31) (1:18,122) (2:20,70) (3:25,109) (4:26,52) (5:27,69)
20	182	69	(0:12,53) (1:13,9) (2:19,73) (3:27,42)
21	133	57	(0:14,38) (1:15,25) (2:22,25) (3:28,34)
22	420	99	(0:15,97) (1:21,23) (2:23,108) (3:28,67) (4:29,57) (5:30,59)
23	410	58	(0:15,34) (1:16,60) (2:17,78) (3:22,111) (4:24,54) (5:30,57)
24	408	90	(0:17,36) (1:23,66) (2:25,94) (3:30,110) (4:31,15) (5:32,58)
25	526	63	(0:17,107) (1:18,58) (2:19,106) (3:24,116) (4:26,98) (5:32,23)
26	374	70	(0:19,52) (1:25,116) (2:27,46) (3:32,55) (4:33,15) (5:34,84)
27	200	46	(0:19,77) (1:20,32) (2:26,57) (3:34,29)
28	136	62	(0:21,37) (1:22,67) (2:29,26)
29	173	87	(0:22,56) (1:28,26) (2:30,82)
30	346	58	(0:22,60) (1:23,46) (2:24,112) (3:29,78) (4:31,41)
31	99	48	(0:24,20) (1:30,27) (2:32,36)
32	214	41	(0:24,48) (1:25,36) (2:26,51) (3:31,33) (4:33,35)
33	84	63	(0:26,12) (1:32,37) (2:34,14)
34	143	83	(0:26,85) (1:27,27) (2:33,24)

In order to compare results we will use the same test networks of Taheri and Zomaya in [4, 6]. Each of these networks has a set of data for each cell, as presented in Table 2 for the 5x5 network from [6]. The first column represents the cell identification, the second is the number of total updates that each cell may have, the third one means the number of calls received in each cell and the fourth corresponds to the number of updates to be considered by each cell whose neighbours change their LAs to the same one. In this work we use four distinct networks with respective sizes of 5x5 (see Table II), 5x7 (see Table III), 7x7 (see Table IV) and 7x9 (see Table V) cells from [4, 6], with the objective of test the performance of DE approach applied to networks with distinct sizes.

TABLE IV
TEST NETWORK 7X7 ATTRIBUTES

No	UpP	CAr	Neighbors
0	133	54	(0:1,63) (1:7,61)
1	317	86	(0:0,60) (1:2,34) (2:7,27) (3:8,97) (4:9,88)
2	162	50	(0:1,40) (1:3,44) (2:9,70)
3	171	61	(0:2,38) (1:4,30) (2:9,22) (3:10,36) (4:11,28)
4	64	46	(0:3,19) (1:5,14) (2:11,17)
5	90	39	(0:4,13) (1:6,16) (2:11,13) (3:12,24) (4:13,9)
6	33	41	(0:5,15) (1:13,9)
7	248	60	(0:0,56) (1:1,31) (2:8,93) (3:14,63)
8	515	66	(0:1,97) (1:7,79) (2:9,52) (3:14,26) (4:15,135) (5:16,118)
9	455	64	(0:1,74) (1:2,79) (2:3,36) (3:8,55) (4:10,87) (5:16,117)
10	338	61	(0:3,34) (1:9,92) (2:11,38) (3:16,39) (4:17,74) (5:18,46)
11	166	64	(0:3,26) (1:4,12) (2:5,29) (3:10,25) (4:12,32) (5:18,31)
12	145	39	(0:5,18) (1:11,17) (2:13,13) (3:18,38) (4:19,32) (5:20,13)
13	61	43	(0:5,6) (1:6,13) (2:12,17) (3:20,14)
14	200	61	(0:7,71) (1:8,26) (2:15,47) (3:21,43)
15	476	61	(0:8,120) (1:14,52) (2:16,55) (3:21,39) (4:22,111) (5:23,82)
16	544	49	(0:8,103) (1:9,116) (2:10,45) (3:15,48) (4:17,98) (5:23,120)
17	462	51	(0:10,92) (1:16,102) (2:18,43) (3:23,45) (4:24,86) (5:25,79)
18	253	50	(0:10,43) (1:11,45) (2:12,22) (3:17,38) (4:19,34) (5:25,57)
19	198	58	(0:12,29) (1:18,24) (2:20,27) (3:25,38) (4:26,47) (5:27,20)
20	88	50	(0:12,13) (1:13,21) (2:19,22) (3:27,18)
21	183	35	(0:14,50) (1:15,31) (2:22,39) (3:28,51)
22	472	86	(0:15,120) (1:21,41) (2:23,58) (3:28,58) (4:29,106) (5:30,80)
23	502	59	(0:15,84) (1:16,104) (2:17,52) (3:22,61) (4:24,85) (5:30,102)
24	507	54	(0:17,106) (1:23,87) (2:25,56) (3:30,68) (4:31,86) (5:32,87)
25	403	47	(0:17,86) (1:18,62) (2:19,36) (3:24,54) (4:26,70) (5:32,81)
26	306	57	(0:19,47) (1:25,66) (2:27,28) (3:32,33) (4:33,88) (5:34,27)
27	101	56	(0:19,18) (1:20,24) (2:26,25) (3:34,17)
28	232	71	(0:21,51) (1:22,65) (2:29,53) (3:35,54)
29	448	82	(0:22,88) (1:28,51) (2:30,88) (3:35,54) (4:36,70) (5:37,91)
30	550	89	(0:22,91) (1:23,104) (2:24,74) (3:29,71) (4:31,104) (5:37,97)
31	534	53	(0:24,82) (1:30,100) (2:32,65) (3:37,73) (4:38,44) (5:39,156)
32	493	44	(0:24,114) (1:25,93) (2:26,47) (3:31,54) (4:33,99) (5:39,76)
33	449	70	(0:26,84) (1:32,118) (2:34,21) (3:39,64) (4:40,77) (5:41,79)
34	100	28	(0:26,22) (1:27,23) (2:33,19) (3:41,25)
35	170	65	(0:28,53) (1:29,60) (2:36,37) (3:42,14)
36	256	58	(0:29,65) (1:35,38) (2:37,75) (3:42,15) (4:43,15) (5:44,39)
37	482	111	(0:29,75) (1:30,96) (2:31,103) (3:36,78) (4:38,83) (5:44,35)
38	339	35	(0:31,39) (1:37,99) (2:39,58) (3:44,33) (4:45,11) (5:46,90)
39	568	71	(0:31,141) (1:32,100) (2:33,64) (3:38,64) (4:40,122) (5:46,66)
40	416	59	(0:33,78) (1:39,128) (2:41,45) (3:46,42) (4:47,46) (5:48,69)
41	192	49	(0:33,88) (1:34,33) (2:40,30) (3:48,35)
42	36	31	(0:35,15) (1:36,9) (2:43,3)
43	41	42	(0:36,10) (1:42,2) (2:44,18)
44	166	52	(0:36,40) (1:37,30) (2:38,45) (3:43,13) (4:45,25)
45	73	33	(0:38,10) (1:44,31) (2:46,22)
46	311	56	(0:38,87) (1:39,71) (2:40,41) (3:45,30) (4:47,75)
47	152	54	(0:40,54) (1:46,76) (2:48,15)
48	135	73	(0:40,77) (1:41,29) (2:47,19)

B. Parameters Definition

The DE algorithm starts with the definition of an initial population of candidate solutions (individuals). Each individual represents a possible configuration of the network and is composed of N genes, where the N corresponds to the number of cells in the network. Each gene of the individual represents the number of the LA where the cell belongs to.

To define the initial population we assumed, as in other works [4], that there are only two LAs, and one of them is set to each cell with a probability of 50%. After that we have adjusted the parameters value to the ones indicated to each experiment.

TABLE V
TEST NETWORK 7X9 ATTRIBUTES

No	UpP	CAr	Neighbors
0	120	67	(0:1,68) (1:9,43)
1	345	68	(0:0,69) (1:2,43) (2:9,29) (3:10,81) (4:11,114)
2	173	58	(0:1,39) (1:3,75) (2:11,52)
3	307	67	(0:2,84) (1:4,44) (2:11,56) (3:12,45) (4:13,62)
4	111	10	(0:3,47) (1:5,47) (2:13,11)
5	289	42	(0:4,46) (1:6,88) (2:13,66) (3:14,38) (4:15,38)
6	184	39	(0:5,88) (1:7,40) (2:15,46)
7	323	78	(0:6,43) (1:8,69) (2:15,111) (3:16,59) (4:17,29)
8	121	35	(0:7,79) (1:17,35)
9	202	52	(0:0,39) (1:1,31) (2:10,79) (3:18,48)
10	462	64	(0:1,70) (1:9,73) (2:11,68) (3:18,27) (4:19,97) (5:20,118)
11	517	75	(0:1,120) (1:2,38) (2:3,60) (3:10,48) (4:12,121) (5:20,121)
12	426	30	(0:3,53) (1:11,111) (2:13,48) (3:20,45) (4:21,69) (5:22,87)
13	287	51	(0:3,65) (1:4,14) (2:5,62) (3:12,56) (4:14,47) (5:22,30)
14	370	45	(0:5,45) (1:13,59) (2:15,95) (3:22,77) (4:23,60) (5:24,21)
15	401	44	(0:5,38) (1:6,38) (2:7,108) (3:14,98) (4:16,34) (5:24,77)
16	325	67	(0:7,49) (1:15,42) (2:17,68) (3:24,84) (4:25,54) (5:26,12)
17	199	64	(0:7,30) (1:8,36) (2:16,74) (3:26,50)
18	148	61	(0:9,39) (1:10,25) (2:19,37) (3:27,33)
19	335	51	(0:10,92) (1:18,32) (2:20,33) (3:27,35) (4:28,84) (5:29,46)
20	541	65	(0:10,120) (1:11,98) (2:12,39) (3:19,42) (4:21,128) (5:29,99)
21	577	66	(0:12,72) (1:20,137) (2:22,67) (3:29,47) (4:30,95) (5:31,140)
22	433	51	(0:12,85) (1:13,34) (2:14,79) (3:21,69) (4:23,71) (5:31,84)
23	527	89	(0:14,62) (1:22,77) (2:24,107) (3:31,123) (4:32,83) (5:33,67)
24	377	38	(0:14,41) (1:15,58) (2:16,87) (3:23,94) (4:25,26) (5:33,56)
25	207	38	(0:16,42) (1:24,21) (2:26,23) (3:33,42) (4:34,54) (5:35,16)
26	130	30	(0:16,16) (1:17,48) (2:25,26) (3:35,29)
27	143	43	(0:18,33) (1:19,29) (2:28,29) (3:36,40)
28	332	49	(0:19,74) (1:27,38) (2:29,46) (3:36,48) (4:37,75) (5:38,37)
29	381	58	(0:19,46) (1:20,74) (2:21,52) (3:28,47) (4:30,65) (5:38,82)
30	589	106	(0:21,100) (1:29,75) (2:31,121) (3:38,97) (4:39,77) (5:40,108)
31	745	69	(0:21,146) (1:22,87) (2:23,141) (3:30,118) (4:32,137) (5:40,102)
32	602	109	(0:23,92) (1:31,128) (2:33,71) (3:40,102) (4:41,88) (5:42,108)
33	331	64	(0:23,57) (1:24,57) (2:25,37) (3:32,73) (4:34,45) (5:42,50)
34	248	43	(0:25,39) (1:33,42) (2:35,26) (3:42,40) (4:43,54) (5:44,32)
35	110	29	(0:25,18) (1:26,42) (2:34,17) (3:44,25)
36	172	48	(0:27,34) (1:28,48) (2:37,24) (3:45,53)
37	389	45	(0:28,75) (1:36,23) (2:38,77) (3:45,81) (4:46,81) (5:47,40)
38	440	49	(0:28,42) (1:29,52) (2:30,95) (3:37,84) (4:39,58) (5:47,99)
39	505	48	(0:30,82) (1:38,59) (2:40,120) (3:47,120) (4:48,57) (5:49,52)
40	642	82	(0:30,114) (1:31,128) (2:32,123) (3:39,107) (4:41,114) (5:49,48)
41	478	51	(0:32,68) (1:40,129) (2:42,47) (3:49,56) (4:50,51) (5:51,108)
42	395	39	(0:32,104) (1:33,47) (2:34,27) (3:41,44) (4:43,91) (5:51,67)
43	340	55	(0:34,56) (1:42,70) (2:44,24) (3:51,37) (4:52,67) (5:53,73)
44	134	60	(0:34,35) (1:35,34) (2:43,15) (3:53,36)
45	234	83	(0:36,52) (1:37,84) (2:46,46) (3:54,43)
46	445	80	(0:37,64) (1:45,34) (2:47,146) (3:54,74) (4:55,57) (5:56,57)
47	562	64	(0:37,50) (1:38,82) (2:39,143) (3:46,137) (4:48,58) (5:56,75)
48	378	46	(0:39,57) (1:47,65) (2:49,78) (3:56,79) (4:57,24) (5:58,61)
49	345	48	(0:39,55) (1:40,69) (2:41,50) (3:48,69) (4:50,61) (5:58,27)
50	366	33	(0:41,63) (1:49,80) (2:51,39) (3:58,56) (4:59,15) (5:60,99)
51	460	34	(0:41,109) (1:42,75) (2:43,42) (3:50,48) (4:52,105) (5:60,70)
52	379	78	(0:43,68) (1:51,114) (2:53,33) (3:60,50) (4:61,47) (5:62,61)
53	182	57	(0:43,62) (1:44,46) (2:52,36) (3:62,28)
54	153	59	(0:45,44) (1:46,81) (2:55,19)
55	167	60	(0:46,46) (1:54,23) (2:56,90)
56	350	58	(0:46,43) (1:47,82) (2:48,98) (3:55,73) (4:57,48)
57	125	69	(0:48,27) (1:56,33) (2:58,46)
58	244	58	(0:48,59) (1:49,20) (2:50,50) (3:57,38) (4:59,60)
59	126	55	(0:50,21) (1:58,37) (2:60,51)
60	381	63	(0:50,120) (1:51,84) (2:52,45) (3:59,35) (4:61,90)
61	173	65	(0:52,47) (1:60,100) (2:62,19)
62	121	73	(0:52,61) (1:53,28) (2:61,23)

C. Algorithm Implementation

After the initial population is defined, the algorithm proceeds to manipulate the population until a termination condition is reached.

Below we present the outline of the implemented algorithm, with the scheme DE/best/1/exp. In our implementation we will use the ten DE schemes and observe how each of them influence the possibility of obtain the better results, after being defined the best value to each DE parameter. The DE/best/1/exp uses the best individual at the moment, and an exponential crossover:

1. Initialize the population
2. Validate the initial population
3. Evaluate the initial population
4. While termination condition is not satisfied, create next population where each individual (candidate solution) is generated according to:
   a) Randomly select 2 distinct individuals xr1 and xr2 from the population, but different from xbest
   b) Generate a trial individual based on the formula: xtrial = xbest + F(xr1 – xr2)
   c) Use the probability Cr to define the amount of genes changed in trial individual
   d) Validate the trial individual
   e) Evaluate the trial individual

Here the terminal condition will be the number of generations defined by the value 1000, because running the algorithm for unlimited number of generations is not a good choice for DE.

D. Individuals Validation

When an individual is generated we must consider that an invalid configuration network may be created. This is because with the application of the algorithm it is possible that we have scattered LAs. This means that we may have cells attributed to the same LA in distinct places of the network, as shown in Fig. 3, but in reality that is not possible and we must correct or discard the individual.

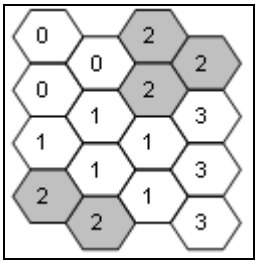


Fig. 3. Scattered LA (LA 2)

To solve this problem we created a method to split these scattered LAs into small ones. Then we applied another method to merge LAs, with the purpose of not having only one cell belonging to a LA, when all their neighbour cells belong to different LAs. Finally, after this, we must renumber the LAs because during all the process some LA numbers may have been deleted.

This process must be repeated for all the individuals that are generated, to assure that the final solution will be a valid one.

E. Fitness Function

In our approach the fitness function corresponds to the calculus of the total cost of location management, which is defined according to the equation (3) presented in section 2.3. This means that for each individual generated (composed of a number of LAs), we will calculate its fitness value, which corresponds to the sum of the total cost of each of those LAs.

F. Simulation Results and Analysis

In order to compare results, the values of always-update and never-update strategies were calculated for all the four test networks.

Then, with the objective of study in more detail the best configuration of DE, we have executed four distinct experiments. For each experiment, and for every combination of parameters, 30 independent runs have been performed in order to assure its statistical relevance. Due to the complexity of the problem, but with the objective of taking the best conclusions, we chose networks from small to medium size to validate our approach.

Like other authors, as Taheri and Zomaya [4, 6], in this study, four distinct test networks are used to ensure the reliability of results. The fact that the results are similar to those test networks (existing networks of different sizes) ensures that the best configuration of parameters can be generalized to any network.

F.1 Experiment 1 – Defining the best NI

The first experiment has the intent of defining the best NI value (which means, define the best population size). So, for that we have fixed the values of F to 0.5, Cr to 0.1, DE strategy as DE/rand/1/bin and the number of generations to 1000, from earlier experiments that we have executed [11, 12]. Then we have initialized the size of NI with 10 and changing it up to 100 with the values 25, 50 and 75.

After this we observed that until now the average of fitness values always presents a positive evolution, so because of that we decided to proceed increasing NI. Considering the results obtained to the best and average fitness values and observing the evolution tendency we have seen that the best value to NI is 250 (as it is possible to see in Table VI) because, although the values between 275 and 400 have been experimented, their results were worse and the evolution of the average fitness became negative.

In order to allow a quick analysis over the best results it was used, in the tables of results, the red colour to mark the best fitness values, one yellow mark to the best average fitness values and one blue mark for the minimum standard deviation values.

With this experiment we have concluded that after a NI value bigger than 250 the positive evolution of the results stop or decrease, in such a way that there are not clearly improvements. We also have to consider that growing the NI value has a direct implication in the increase of execution time.

Due to all of this, we have chosen NI=250, to pass to the second experiment, as an equilibrium point for obtaining good results in small times of execution.

F.2 Experiment 2 – Defining the best Cr

The second experiment has the objective of electing the Cr value that obtains the best results for all, or for the majority, of the test networks.

To proceed with this experiment we initialized and fixed the values of NI to 250 (obtained from experiment 1), F to 0.5, DE scheme as DE/rand/1/bin and the number of generations to 1000 (as defined in the experiment 1).

TABLE VI
EXPERIMENT 1: DEFINING THE BEST NI

Fitness Evaluation													
5x5 Network													
NI	**10**	**25**	**50**	**75**	**100**	**125**	**150**	**175**	**200**	**225**	**250**	**275**	**300**
Best	27216	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990
Average	28992.7	27518.4	27281.4	27292.2	27264.7	27191.0	27148.6	27119.9	27119.5	27149.0	27100.9	27104.7	27062.6
St. Dev.	3358.5	311.5	216.7	172.8	134.3	140.5	147.8	139.1	123.6	110.8	119.6	110.7	103.3
5x7 Network													
Best	41458	40645	40754	40645	40328	40645	40645	40582	40582	40427	40256	40328	40328
Average	44493.7	42415.6	42188.1	42043.6	41638.6	41752.8	41542.3	41393.0	41385.9	41545.6	41313.1	41289.4	41016.0
St. Dev.	3268.9	1025.3	753.4	661.2	615.7	576.6	650.1	499.6	508.5	532.4	517.1	419.4	409.5
7x7 Network													
Best	65331	64362	65153	64879	64879	64674	64161	64732	64477	64433	65458	64043	63958
Average	71501.9	67907.9	67228.8	66830.5	66803.1	66443.1	66252.9	66166.3	66264.2	65996.5	66466.7	65657.9	65873.7
St. Dev.	9670.7	1036.0	775.5	972.1	828.6	984.1	853.0	634.0	762.3	830.9	623.2	851.6	693.7
7x9 Network													
Best	96277	95296	95969	97440	95246	95640	94304	96329	94908	95110	94293	94888	95080
Average	102158.6	100379.7	99699.3	99268.4	98848.2	98567.7	98511.7	98098.4	97800.3	97955.1	97686.8	97413.1	97589.5
St. Dev.	2110.1	1769.0	1288.6	989.0	1442.2	1405.5	1322.7	1183.6	1249.3	1396.3	1260.3	1244.1	1225.7

TABLE VII
EXPERIMENT 2: DEFINING THE BEST CR

Fitness Evaluation										
5x5 Network										
CR	**0.01**	**0.03**	**0.05**	**0.07**	**0.09**	**0.1**	**0.25**	**0.5**	**0.75**	**0.9**
Best	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990
Mean	27249.0	27277.3	27159.5	27145.8	27107.2	27070.6	27038.6	27090.4	27086.5	27136.1
St. Dev.	131.8	112.9	141.5	119.7	127.1	106.2	76.8	111.2	129.7	146.5
5x7 Network										
Best	40672	40645	40645	40645	40525	40301	40466	40301	41465	42219
Mean	41661.7	41552.2	41674.8	41763.8	41405.4	41188.6	41228.7	41398.2	42384.7	42616.6
St. Dev.	627.2	486.6	586.8	449.0	536.6	474.9	416.0	486.0	219.5	278.5
7x7 Network										
Best	64769	65030	64729	63815	63534	63874	64674	64305	67380	67232
Mean	66819.0	66739.1	66368.1	66185.7	66029.1	66057.7	65915.5	67396.8	68937.2	69361.9
St. Dev.	876.1	703.7	638.1	956.5	1041.1	974.6	621.4	1077.1	690.7	1246.8
7x9 Network										
Best	95487	93285	94402	95208	95565	95492	96979	97884	101417	103666
Mean	100178.8	98751.6	98362.5	98015.2	97943.1	97547.0	99332.5	103542.7	105689.5	105707.1
St. Dev.	1313.7	1730.6	1446.9	1064.6	1021.1	1015.4	920.7	2467.3	1779.6	988.0

With these fixed parameters, the experiment was executed initially with Cr equal to 0.1 and follow changing it to the values 0.25, 0.50, 0.75 and 0.9. After obtaining all the results, we could observe that, in the most of the cases, they became worse with the increase of the CR value. Until this moment it was possible to say that the best value was Cr=0.1, but to take more complete conclusions we decided to experiment lower values from 0.01 to 0.09. Finally, looking to all the results (see Table VII), it is possible to conclude that really Cr=0.1 is the best and more stable value to obtain better results.

F.3 Experiment 3 – Defining the best F

In the third experiment we pretend to define the best value of F, that allows us to obtain the best fitness values in the majority of the test networks or, if it is possible, to all the test networks.

TABLE VIII
EXPERIMENT 3: DEFINING THE BEST F

Fitness Evaluation					
5x5 Network					
F	**0.1**	**0.25**	**0.5**	**0.75**	**0.9**
Best	26990	26990	26990	26990	26990
Mean	27080.6	27072.7	27141.8	27134.3	27078.4
St. Dev.	123.7	106.8	125.5	112.8	114.2
5x7 Network					
Best	40473	40466	40328	40496	40328
Mean	41491.0	41221.5	41194.3	41287.8	41266.3
St. Dev.	558.2	536.1	447.2	477.9	475.3
7x7 Network					
Best	64893	64893	64671	64879	64207
Mean	66051.2	66140.0	66192.1	65993.3	65981.9
St. Dev.	554.3	749.8	602.5	679.3	790.2
7x9 Network					
Best	96220	95076	93040	94774	95105
Mean	98116.6	97821.8	97826.5	97884.3	97849.8
St. Dev.	1022.6	1281.1	1402.2	925.5	1213.3

So, in order to execute this experiment we fixed the value of NI to 250 (from experiment 1), Cr to 0.1 (from experiment 2), DE scheme as DE/rand/1/bin and 1000 generations as stop criterion (as defined in the two earlier experiments). The value of F was initialized to a probability of 0.1, and then the algorithm was also evaluated with the values of 0.25, 0.50, 0.75 and 0.9.

Observing the results obtained with this experiment, that are presented in Table VIII, it is possible to verify that, principally, the F values of 0.5 and 0.9 permit obtain better results. But F=0.5 was the elected one because it is the one that performs better when considering also the fitness average evolution.

F.4 Experiment 4 – Defining the best DE scheme

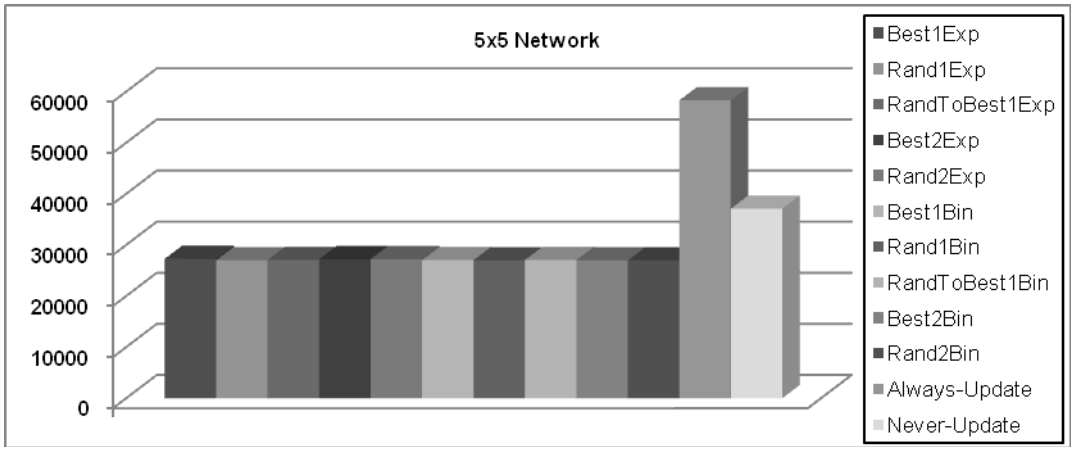
After the three earlier experiments we have obtained and fixed the best values for the DE parameters as NI=250, Cr=0.1 and F=0.5. So in this last one we try to define what is the most appropriate scheme, that is, the DE scheme that permits to obtain the best results. For that, and again for each test network, the algorithm has been executed applying all the ten DE schemes.

Once obtained all the results, we could conclude that the scheme DE/rand/1/bin is the one that performs better (see Table IX), and that permits to obtain the best fitness value in three of the four test networks.

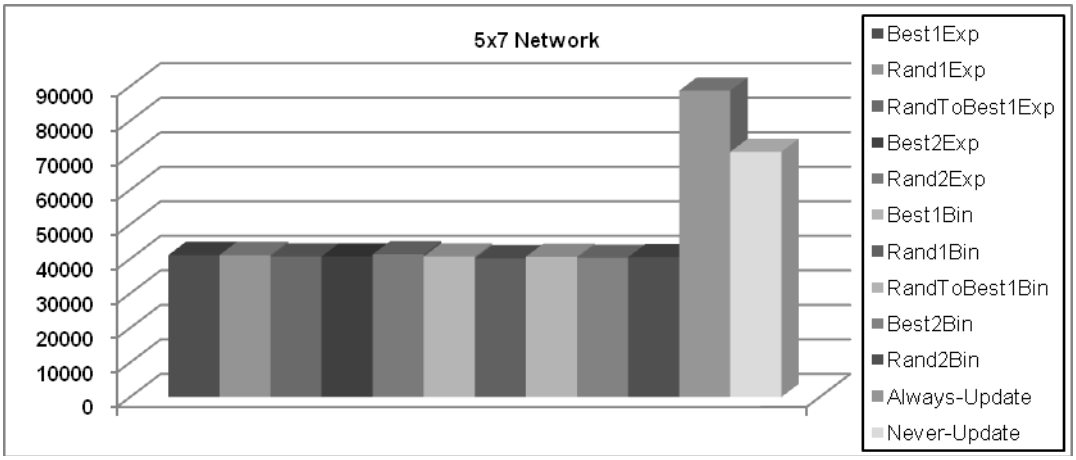
Finishing these four experiments we had defined the best DE configuration, applied to the Location Areas problem, setting the parameters as NI=250, Cr=0.1, F=0.5 and DE scheme as DE/rand/1/bin.

TABLE IX
EXPERIMENT 4: DEFINING THE BEST DE SCHEME

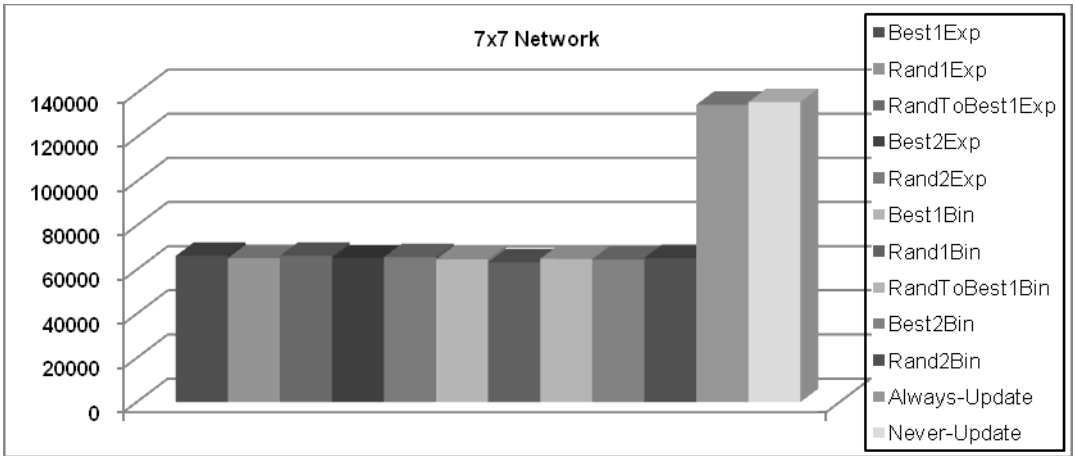
Fitness Evaluation										
5x5 Network										
	Exponential Crossover					**Binomial Crossover**				
Scheme	**Best1**	**Rand1**	**RandToBest1**	**Best2**	**Rand2**	**Best1**	**Rand1**	**RandToBest1**	**Best2**	**Rand2**
Best	27282	26990	27048	27211	27211	27048	26990	27048	26990	26990
Mean	27871.2	27620.4	27946.3	27784.2	27572.8	27304.9	27077.8	27420.6	27291.4	27102.7
St. Dev.	305.0	297.3	395.9	354.6	295.9	132.9	111.4	255.5	169.4	125.6
5x7 Network										
Best	41141	41141	40722	40722	41340	40706	40205	40645	40346	40525
Mean	42772.1	42499.1	42853.6	42497.0	42542.5	41692.0	41261.8	41917.4	41627.5	41351.4
St. Dev.	987.1	1010.5	1136.6	794.4	927.9	400.8	562.2	676.1	600.4	387.2
7x7 Network										
Best	66215	65281	66243	65188	65658	64625	63307	64890	64560	65290
Mean	67709.3	67510.0	68417.0	67708.2	67367.4	66366.3	65737.1	66976.2	66247.6	66273.2
St. Dev.	1093.5	1330.8	1276.8	1399.4	1126.5	828.3	854.2	893.7	790.7	520.6
7x9 Network										
Best	100386	100484	98967	99512	100295	95125	94841	96408	92900	94483
Mean	102580.6	103191.6	103152.4	103199.8	103100.8	97947.5	97895.1	98346.4	97479.8	97598.4
St. Dev.	1166.4	1212.2	1643.1	1751.7	1213.0	996.4	1275.4	945.2	1408.7	1285.8



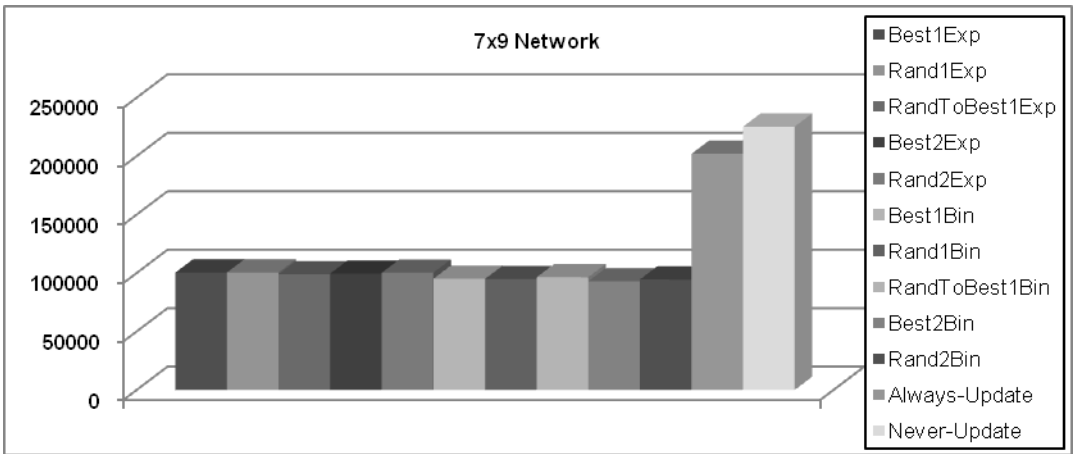
a)



b)



c)



d)

Fig. 4. Comparison Results a) 5x5 Network b) 5x7 Network c) 7x7 Network d) 7x9 Network

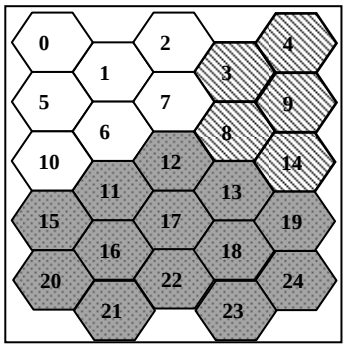
F.5 Comparing our results with other applied algorithms

Now, if we compare our results with the classical strategies always-update and never-update we may say that, for all the used test networks, our approach always obtains better solutions (lower fitness values) as it is possible to see in Fig. 4.

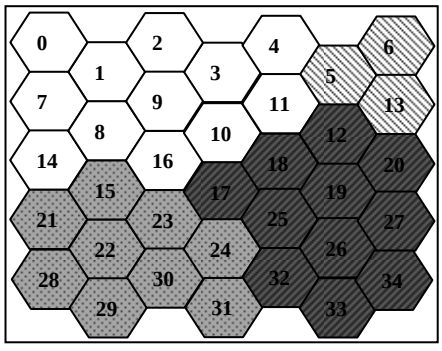
Comparing with studies of other authors, as Taheri and Zomaya [4, 6, 13], that use respectively genetic algorithms, simulated annealing and hopfield neural network approaches, our results are very similar and in some cases even better.

For example, for the 5x5 network, our best fitness solution corresponds to a cost of 26990 and their best result is between 25000 and 30000. Using the 5x7 network, our best fitness solution represents a cost of 40205 and their results are between 40000 and 45000. In the 7x7 network our lower cost is 63307 and their best value is between 60000 and 65000. Finally, for the 7x9 network the best fitness value obtained by our approach is 92900 and their best solution is between 90000 and 95000.

All of these costs were calculated with the network partitioning defined by the DE algorithm and represented in Fig. 5.



a)



b)

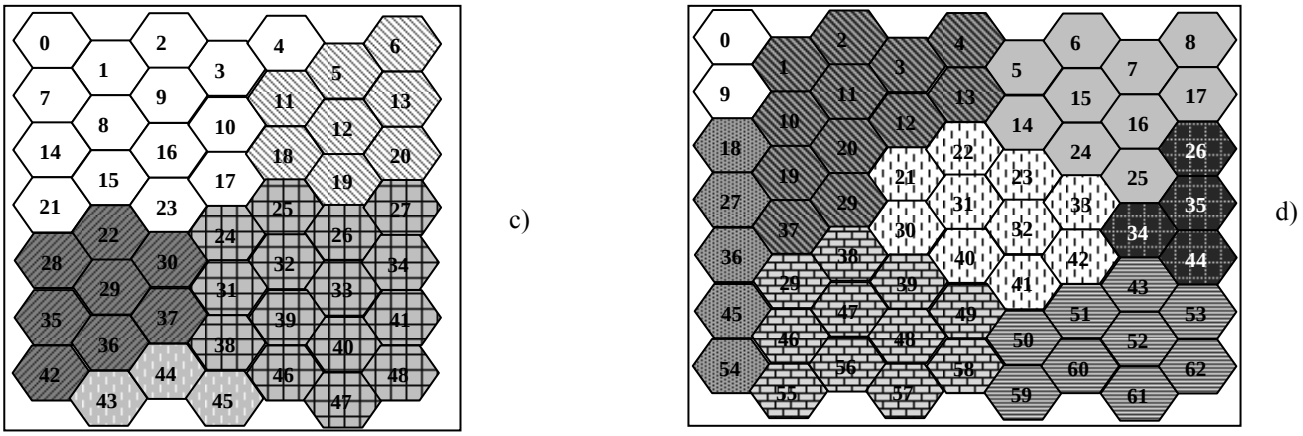


Fig. 5. Best LAs Configuration a) 5x5 Network b) 5x7 Network c) 7x7 Network d) 7x9 Network

With respect to the ideal number of location areas, we observed that, for the 5x5 network, all the best solutions correspond to a network partitioning in 3 Location Areas (see Fig. 5a). When we refer to the 5x7 network, the best solution corresponds to a partitioning in 4 distinct LAs (see Fig. 5b). Moving to the 7x7 network, the ideal partitioning is represented in 5 LAs (see Fig. 5c). For the bigger network, the 7x9, the best configuration corresponds to a partitioning in 8 LAs (see Fig. 5d).

Relatively to the shape of the LAs, the most of them do not have a circular shape, as in the actual GSM systems. Their forms are diverse but principally of triangular or rectangular shape.

F.6 The importance of the number of generations

The DE algorithm is a population-based algorithm that improves its results generation by generation. Considering this, we may say that obtaining the best results depends on the number of generations defined as stop criterion. In this work, we always have used 1000 generations, because increasing it corresponds to increase the execution time. However, there are several works [4, 14] that present the results obtained with an "infinite" or very high number of iterations (generations). With the objective of compare our results with those ones, we decided to execute our approach for all the four tests networks using 5000 generations as stop criterion.

In Table X is shown the evolution of results (best fitness value/lower cost for each test network) over the algorithm execution during the 5000 generations. It is possible to conclude that having more generations, permits to obtain better results.

TABLE X
EVOLUTION OF RESULTS OVER 5000 GENERATIONS

Test Network	Generations				
	1000	**2000**	**3000**	**4000**	**5000**
5x5	26990	26990	26990	26990	26990
5x7	40205	40117	40085	40085	39859
7x7	63307	62720	61951	61567	61037
7x9	92900	91104	90687	90437	89973

Now, in Table XI we compare the new results with the ones presented by Taheri and Zomaya in [14], where "infinite" or very high number of iterations is used. We can observe that DE (with only 5000 iterations) always performs better than GA (Genetic Algorithm). If we compare with HNN (Hopfield Neural Network), SA (Simulated Annealing) or with the GA-HNNx (different combinations of Genetic Algorithm and Hopfield Neural Network, see [14]), in the most of the cases the results are similar or even better.

TABLE XI
COMPARISON OF NETWORK COSTS WITH DIFFERENT ALGORITHMS

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

Considering these results we may say that if our algorithm runs using endless generations, it would probably overcome the remaining results obtained by the other methods.

V. CONCLUSIONS AND FUTURE WORK

This paper presents a new approach based on DE algorithm with the objective of finding the best configuration for the LAs in a mobile network. It also intends to understand the influence of DE parameters and schemes. One of the principal characteristics of using DE algorithm is the fact that we always obtain (until the optimal solution is found) an equal or better individual in each generation.

We have shown that our approach improves the results obtained with other classical location management strategies as always-update and never-update.

When our implementation results are compared with the ones of other authors, it is possible to conclude that they are considered interesting because they are equal or better, when applied to the same test networks. We have studied in detail the best configuration of DE, and the best parameters, after a big number of experiments with four distinct networks, are NI of 250, Cr of 0.1, F of 0.5 and DE/rand/1/bin as the best scheme. It is also possible to conclude that in general the binomial schemes perform better than the exponential ones.

Each execution with the 5x5 test network took about 15 seconds, with the 5x7 test network took about 25 seconds and with the 7x7 and 7x9 test networks took 35 and 40 seconds respectively. Considering the execution time for each test network, we may say that it is proportional to the network size, but also that those times are low and good to be used in industry. In total, to perform all the experiments (more than 5000 independent runs), for this paper, around 90 hours were needed.

As future work we have the intention of test our approach working with bigger test networks seeing if it works well or if the performance decreases.

We have also planned to use real data (like SUMATRA [15]) as input for generating the test networks and then to apply our approach.

The application of other evolutionary strategies to the LA problem and the comparison of their results with the ones accomplished by the DE algorithm are also a matter of future work.

Finally, the formulation of the LA problem as a multiobjective optimization problem will be investigated as well.

ACKNOWLEDGMENTS

This work has been developed in part thanks to the project OPLINK (TIN2005-08818-C04-03). Thanks also to the Polytechnic Institute of Leiria, for the economic support offered to Sónia Almeida-Luz to make this research.

REFERENCES

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

[2] V.W.S. Wong, V.C.M. Leung: *Location Management for Next-Generation Personal Communications Networks*. IEEE Network, October 2000, vol. 14, no. 5, pp. 18-24

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

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

[5] P. Demestichas, N. Georgantas, E. Tzifa, V. Demesticha, M. Striki, M. Kilanioti, M. Theologou: *Computationally Efficient Algorithms for Location Area Planning in Future Cellular Systems*. Computer Communications, vol. 23, no. 13, July 2000, pp. 1263-1280.

[6] J. Taheri, A.Y. Zomaya: *A Simulated Annealing Approach for Mobile Location Management*, in 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS), April 2005, pp. 194-201.

[7] J. Taheri, A.Y. Zomaya: *Clustering Techniques for Dynamic Mobility Management*, in the 4th ACM Workshop on Mobility Management and Wireless Access (Part of MSWiM'06), October 2006, pp. 10–17.

[8] R. Subrata, A.Y. Zomaya: *Evolving Cellular Automata for Location Management in Mobile Computing Networks*. IEEE Transactions on Parallel and Distributed Systems, vol. 14, no. 1, January 2003, pp. 13-26.

[9] K. Price, R. Storn: Web Site of Differential Evolution (on March 2008): http://www.icsi.berkeley.edu/~storn/code.html.

[10] R. Storn, K. Price: *Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces*. In Journal of Global Optimization, December 1997, vol. 11, no. 4, pp. 341-359.

[11] S.M. Almeida-Luz, M.A. Vega-Rodríguez, J.A. Gómez-Pulido, J.M. Sánchez-Pérez: *A Differential Evolution Algorithm for Location Area Problem in Mobile Networks*. In SoftCOM 2007 - 15th International Conference on Software, Telecommunications and Computer Networks - co-sponsored by the IEEE Communications Society (COMSOC), ISBN: 953-6114-95-X, pp. 1-5, September 2007.

[12] 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*. In EPIA 2007 – 13th Portuguese Conference on Artificial Intelligence, Guimarães (Portugal); Published in "New Trends in Artificial Intelligence", (Eds). APPIA, Associação Portuguesa para a Inteligência Artificial, December 2007. ISBN: 978-98-995-6180-9; pp. 219-230.

[13] J. Taheri, A.Y. Zomaya: *The Use of a Hopfield Neural Network in Solving the Mobility Management Problem*. IEEE/ACS International Conference on Pervasive Services (ICPS'04), 2004, pp. 141-150.

[14] J. Taheri, A.Y. Zomaya: *A Combined Genetic-Neural Algorithm for Mobility Management*. Journal of Mathematical Modelling and Algorithms, Springer Netherlands, Volume 6, Number 3, September 2007, pp. 481-507.

[15] Stanford University Mobile Activity TRAces (SUMATRA) (on March 2008): http://infolab.stanford.edu/sumatra/.

Sónia M. Almeida-Luz is a professor of Programming and Information Systems in the Dept. of Computer Engineering, School of Technology and Management, Polytechnic Institute of Leiria, Leiria, Portugal. She is currently making investigation to her PhD in evolutionary computing and optimization. Her main research interests are information systems, applications of artificial intelligence and evolutionary computing.

Miguel A. Vega-Rodríguez is a professor of Computer Architecture in the Dept. Technologies of Computers and Communications, University of Extremadura, Spain. He received a PhD degree in Computer Science from the University of Extremadura. Dr. Vega-Rodríguez has authored or co-authored more than 200 publications including journal papers, book chapters and peer-reviewed conference proceedings. Furthermore, he is editor and reviewer of several international journals. Dr. Vega-Rodriguez's main research interests are parallel and reconfigurable computing, and also evolutionary computing.

Juan A. Gómez-Pulido is a professor of Computer Architecture in the Dept. Technologies of Computers and Communications, University of Extremadura, Spain. He received a PhD degree in Computer Science from the Complutense University of Madrid in 1993. His main research interests are artificial intelligence and applications of reconfigurable hardware to different fields of signal processing.

Juan M. Sánchez-Pérez is Professor of Computer Architecture in the Dept. Technologies of Computers and Communications, University of Extremadura, Spain. He received a PhD degree in Physics from the Complutense University of Madrid in 1976. His research interests are artificial intelligence, logic design and modern computer architectures.