Applied Soft Computing 11 (2011) 410–427

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Differential evolution for solving the mobile location management

Sónia M. Almeida-Luz[a,*], Miguel A. Vega-Rodríguez[b], Juan A. Gómez-Púlido[b], Juan M. Sánchez-Pérez[b]

[a] Department of Computer Science - School of Technology and Management - Polytechnic Institute of Leiria, Campus 2 - Morro do Lena - Alto do Vieiro, Apartado 4163, 2411-901 Leiria, Portugal

[b] Department of Technologies of Computers and Communications, University of Extremadura, Escuela Politécnica, 10071 Cáceres, Spain

ARTICLE INFO

Article history:
Received 2 March 2009
Received in revised form
18 November 2009
Accepted 21 November 2009
Available online 1 December 2009

Keywords:
Differential evolution
Location areas problem
Reporting cells problem
Location management
Costs optimization

ABSTRACT

In this work we present two new approaches to solve the location management problem, respectively, based on the location areas and the reporting cells strategies. The location management problem corresponds to the management of the network configuration with the objective of minimizing the costs involved. We use the differential evolution algorithm to find the best configuration for the location areas and the reporting cells strategies, which principally considers the location update and paging costs. With this work we want to define the best values to the differential evolution configuration, using test networks and also realistic networks, as well as compare our results with the ones obtained by other authors. These two new approaches applied to this problem have given us very good results, when compared with those obtained by other authors.

1. Introduction

The use of mobile networks is growing every day and being applied to the most of newly and renovated applications for data transfer, voice and fax services among many other mobile services. Because of this, communication networks [1] must support a big number of users and their respective applications maintaining a good response without losing quality and availability. With the goal that mobile networks keep this quality it is necessary to consider the mobility management when making design of the network infrastructure.

Mobility management is a very important point because it includes the process of hand off management and location management. The process of hand off management enables the mobile network to locate roaming mobile terminals. The process of location management enables the mobile network to find the current location of a mobile terminal in order to make or receive calls, from any location and at any time of the day.

We are principally concerned about the location 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. One of the major objectives of location management is to minimize the involved costs associated to the user movements and their tracing, and this will be also our major goal.

There exist several strategies of location management and we will apply the location area and reporting cell schemes, which are two of the more common ones.

This article proposes two new approaches that use a differential evolution (DE) based algorithm to solve, respectively, the location areas and reporting cells problems. The main goal of each of these problems is to optimize the configuration, for mobile networks, that minimizes the involved costs.

With the objective of testing our approaches we have used test networks and also realistic networks as SUMATRA [2]. The application of the algorithm is described in detail for both approaches and the parameters were studied intensively to define the most adequate values.

In conclusion, our contributions have the objective of introducing the DE algorithm for solving these important location management problems, outperforming the results of the existing works.

The article is organized as follows. In the next section it is explained the location management and in more detail the location areas and reporting cells strategies. In Section 3, the DE based algorithm is described, as well as its parameters and different possible schemes. Section 4 includes the implementation details for both approaches. In Section 5, the experimental results are presented, analyzed and compared with the results of other authors.

* Corresponding author. Tel.: +351 244 820 300; fax: +351 244 820 310.
E-mail addresses: sluz@estg.ipleiria.pt (S.M. Almeida-Luz), mavega@unex.es (M.A. Vega-Rodríguez), jangomez@unex.es (J.A. Gómez-Púlido), sanperez@unex.es (J.M. Sánchez-Pérez).

1568-4946/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2009.11.031

Finally, Section 6 includes the conclusions and future lines of work.

2. Location management problem

In cellular network systems it is very important to keep track 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 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 [3]. As static techniques, the most common ones are always-update, never-update, location area and reporting cell schemes [3], 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.

There exist several authors working with the location area scheme and applying computationally efficient algorithms like genetic algorithms (GAs) [4–6], simulated annealing (SA) [5,7], tabu search (TS) [5] and clustering techniques [8] (among others).

Also, for the reporting cells scheme, several computationally efficient algorithms have been applied, like GAs, TS and ant colony algorithm (AC) [9]. In [10], a combination of the Hopfield Neural Network (HNN) and the authors' Ball Dropping Technique (BDT) is used. Contrary to all of them, we use a DE based algorithm. This is an important contribution of our work, because, to the best of our knowledge, this is the first time that this algorithm is used to solve the mobile location management problem.

In the following subsections we will present in more detail the location areas and reporting cells schemes and the respective calculus of location management costs.

2.1. Location areas problem

The location areas (LA) scheme corresponds to an important strategy of location management, which 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.

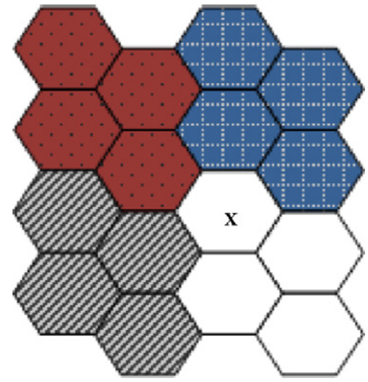

Fig. 1. Network partitioning into location areas.

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, of the location areas scheme, is divided in two main parts: location update cost and location paging cost [4,6].

2.1.1. 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. 2(a), 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 (1):

$$N_{LU} = 108 + 41 + 42 + 73 + 84 + 63 + 58 = 469 \quad (1)$$

2.1.2. 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. 2(b), the calculus of paging cost is (2):

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

2.1.3. Total cost

The location management cost involves other parameters and components, but those are considered to be equal for all strategies [6]. 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 manage-

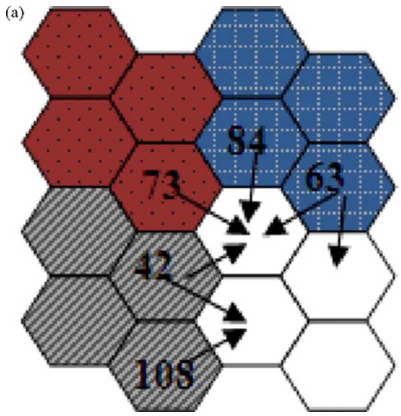

Fig. 2. (a) Entering flow of users to the white LA. (b) Incoming calls to the white LA.

ment [11] is (3):

$$Cost = \beta \times N_{LU} + N_P \tag{3}$$

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 from one cell to another without making any call [6]. 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, $\beta = 10$ [4]. For the white LA, referred earlier, and presented in Fig. 2(a) and (b), the total cost by (3) would be (4):

$$Cost = 10 \times 469 + 736 = 5426 \tag{4}$$

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.

2.2. Reporting cells problem

The reporting cells (RC) planning scheme was proposed by Bar-Noy and Kessler [12] with the objective of minimizing the cost of tracking mobile users.

This strategy is characterized by defining a subset of cells as reporting cells and the others as non-reporting cells (nRC), as it is possible to see in Fig. 3(a) (RC represented with value 1 and in blue/grey colour and nRC represented with value 0 and in white colour). The mobility terminals only perform a new location update when they change their location and move to one reporting cell. If an incoming call must to be routed to the mobile user, the search can be restricted to his last reporting cell known and their respective neighbours which are non-reporting cells.

It is necessary to calculate for each cell the vicinity factor, which represents the maximum number of cells that the user must page when an incoming call occurs.

The vicinity value of a reporting cell corresponds to the number of non-reporting cells that are accessible from this reporting cell, without crossing other reporting cells, and adding the reporting cell itself. For example, considering the calculus of vicinity factor for the cell number 5 (RC) in Fig. 3(a), we must count the number of neighbours that are nRCs (cells 0, 1, 4, 9, 12 and 13) and also include the RC itself, which makes a total of seven neighbours. This total number of neighbours will correspond to the vicinity factor of this RC.

If we are calculating the vicinity value of a non-reporting cell it is necessary to consider the maximum vicinity value among the reporting cells from where this one can be reached. This means that if a non-reporting cell belongs to the neighbourhood of more than one reporting cell, the calculus has to be done for all the reporting cells and then, the maximum number is set as the vicinity factor of the respective non-reporting cell. If we consider the cell number 9 (nRC), in Fig. 3(a), we can observe that it belongs to the neighbourhood of at least two RCs (more precisely four cells: number 5, 8, 10 and 14). Because of that, the calculus of vicinity factor must be done for all those RCs and after this, the maximum number will be considered as the vicinity factor for

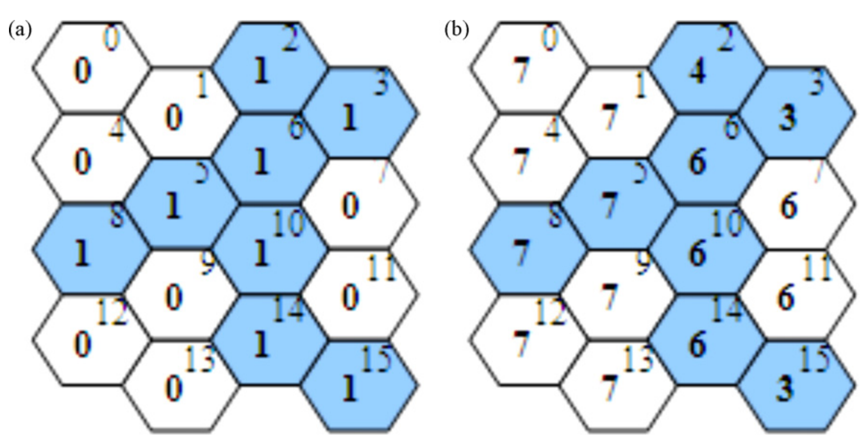

Fig. 3. (a) Reporting cells planning. (b) Reporting cells vicinity values.

cells 5, 8, 10 and 14 is respectively 7, 7, 6 and 6, so the maximum value that represents the vicinity factor of cell number 9 is 7.

Considering the reporting cells planning of Fig. 3(a) and calculating all the vicinity factors, the result will be the one presented in Fig. 3(b).

2.2.1. Location management cost

In this case, the location management cost is principally divided into two fundamental operations: location update and location paging. 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 and must register the new one. 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. Like we have explained in 2.1.3 *Total cost*, the location management cost involves several other parameters and components that are considered to be equal for all strategies and do not make influence when comparing the results obtained by different strategies. Because of that these costs are not considered for the total cost of the reporting cells scheme.

In the reporting cells scheme the location updates only are performed when a mobile user enters in a reporting cell and the vicinity factor of each cell must be considered. Because of that the generic formula given by (3) must be readjusted and it is formulated as [13](5):

$$Cost = \beta \times \sum_{i \in S} N_{LU}(i) + \sum_{i=0}^{N} N_P(i) \times V(i) \tag{5}$$

Here we can see that $N_LU(i)$ corresponds to the number of location updates associated to the reporting cell i, S indicates the subset of cells defined as reporting cells, $N_P(i)$ is the number of incoming calls attributed for cell i, N is the total number of cells that compound the mobile network configuration and $V(i)$ is the vicinity factor attributed for cell i.

We will use this formula with the objective of minimize the location management, for the application of the reporting cells strategy. With the objective of explain these calculus, consider the network, with respective RC configuration, presented in Fig. 3, and the location updates and incoming calls shown in Fig. 4. The location updates are represented in Fig. 4(a) and the respective update cost would be calculated as (6):

$$N_LU = 360 + 548 + 1451 + 816 + 647 + 1105 + 1058 + 434 = 6419 \tag{6}$$

The paging cost is calculated based on the incoming calls, presented in Fig. 4(b) and the vicinity factor of each cell, which were previously calculated and they are shown in Fig. 3(b). Considering this, the calculus would be (7):

$$N_P = 360 \times 7 + 377 \times 7 + 248 \times 4 + 518 \times 3 + 365 \times 7 + 1355 \times 7 + 438 \times 6 + 415 \times 6 + 366 \times 7 + 435 \times 7 + 510 \times 6 + 501 \times 6 + 470 \times 7 + 376 \times 7 + 569 \times 6 + 361 \times 3 = 47,967 \tag{7}$$

After these, and considering that the update cost is considered 10 times greater that the paging cost (as we have explained before), the total cost (considering this reporting cells configuration of the network) would be the following (8):

$$TotalCost = 10 \times 6419 + 47,967 = 112,157 \tag{8}$$

3. Differential evolution algorithm

The differential evolution (DE) is a population-based algorithm, created by Price and Storn [14], 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.

3.1. 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 [15] it is possible to see more details about the DE flowchart).

3.2. 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 Eq. (9):

$$I_{i,g+1} = X_{1,g} + F(X_{2,g} - X_{3,g}) \tag{9}$$

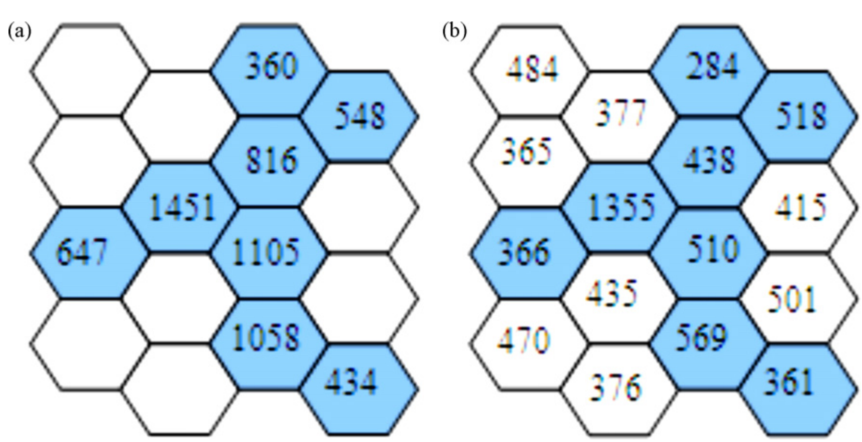

Fig. 4. (a) Reporting cells location updates. (b) Reporting cells incoming calls.

The value of F must be greater than zero and will control the magnitude of the differential variation of two individuals ($X_{2,g} - X_{3,g}$). The individuals X_1, X_2 and X_3 are randomly selected and different among them. The g means the current 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.

3.3. 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 Eq. (10):

$$T_{ji,g+1} = \begin{cases} I_{ji,g+1} & \text{if} \quad rnj \le Cr \\ X_{ji,g} & \text{otherwise} \end{cases} \tag{10}$$

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

3.4. 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 Eq. (11):

$$\begin{array}{lll} \text{If} \quad f(T_{i,g+1}) \le f(X_{i,g}), & \text{set} & X_{i,g+1} = T_{i,g+1} \\ \text{Otherwise} & & X_{i,g+1} = X_{i,g} \end{array} \tag{11}$$

3.5. DE schemes

Price and Storn [14] have suggested 10 different schemes (those are presented in Table 1) 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.

3.6. DE algorithm

The pseudo-code of the DE algorithm, using the $DE/best/1/exp$ is presented in Table 2. It starts by defining and evaluating the initial population through calculating the fitness value for each individual.

Table 1
Differential evolution schemes.

No.	Scheme	Mutant vector generation
1	DE/best/1/exp	$xtrial = xbest + F(xr1 - xr2)$
2	DE/rand/1/exp	$xtrial = xr1 + F(xr2 - xr3)$
3	DE/randtobest/1/exp	$xtrial = xr1 + F1(xbest - xr1) + F2(xr2 - xr3)$
4	DE/best/2/exp	$xtrial = xbest + F(xr1 + xr2 - xr3 - xr4)$
5	DE/rand/2/exp	$xtrial = xr1 + F(xr2 + xr3 - xr4 - xr5)$
6	DE/best/1/bin	$xtrial = xbest + F(xr1 - xr2)$
7	DE/rand/1/bin	$xtrial = xr1 + F(xr2 - xr3)$
8	DE/randtobest/1/bin	$xtrial = Xr1 + F1(xbest - xr1) + F2(xr2 - xr3)$
9	DE/best/2/bin	$xtrial = xbest + F(xr1 + xr2 - xr3 - xr4)$
10	DE/rand/2/bin	$xtrial = xr1 + F(xr2 + xr3 - xr4 - xr5)$

Table 2
DE algorithm pseudo-code with scheme DE/best/1/exp.

```
Initialize the population
Evaluate the initial population
While (termination condition not satisfied) {
    Randomly select ind. xr1 ≠ xbest
    Randomly select ind. xr2 ≠ xr1 and ≠ xbest
    Generate trial ind.: xtrial = xbest + F(xr1 - xr2)
    Use Cr to define the amount of genes changed
       in trial individual
    Evaluate the trial individual
    Deterministic selection
}
```

After that, until the termination condition is not reached, the necessary individuals are picked and a new one is produced according to the selected DE scheme and respective rules. This new individual is evaluated and compared with the old one. Just the one with the best fitness value will be chosen and pass for population of the next generation.

4. Implementation details

In this section we intent to explain the considerations that must be taken before the implementation of experiments.

We will divide this section in two major subsections that will include, respectively, the implementation details for the location areas and the reporting cells schemes.

4.1. Location areas scheme

For the implementation of our experiments, using the location area scheme, we will start detailing the major considerations and decisions. It will start with the detail of source, definition and preparation of the test networks; subsequently we explain the decisions about the total cost calculus; then we expose the original definition of parameters and finally, we explain the most significant decisions and adjustments to the specific problem.

4.1.1. Networks used

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.

In order to compare results we will use the same test networks of Taheri and Zomaya in [6,7]. Each of these networks has a set of data for each cell, as presented in Table 3 for the 5×5 network from [7]. 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 5×5 (see Table 3), 5×7, 7×7 and 7×9 cells from [6,7], with the objective of test the performance of DE approach applied to networks with distinct sizes.

Beyond the use of these test networks, we decided to test our approach using more realistic data. These data were obtained from SUMATRA [2,16]. SUMATRA traces are based on real user network behaviour and are well validated against real world data.

The SUMATRA traces are compound by four distinct traces, each one representing a different situation in a mobile network. From these four traces, we use the BALI-2, because it includes the 24 h call and movement trace for the San Francisco Bay Area cellular network [2]. This test network is compound by 90 cells and 66,550 mobile users.

Table 3
Test network 5 × 5 attributes.

No	UpP	CAr	Neighbours
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)

4.1.2. Fitness function

For the location areas based approach, the fitness function corresponds to the calculus of the total cost of location management, which is defined according to Eq. (4) presented in Section 2.1.3. 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.

However, for the experiments using realistic data from SUMATRA [2,16] we will apply this fitness function with a two-step paging proposed by Subrata and Zomaya in [17] and already applied in [18]. In the first step it is done the paging to the last known location of the user (it is considered that the initial location of each user is known). If the user is not found in the first step, it is applied the second step that consists in making a network wide search. But, considering that we are applying the LA strategy, this search is just conducted over the other cells (except the one already paged in the first step) that compound the respective LA. With this process we try to obtain a compromise between the rapid location of the user and the required level of Quality of Service (QoS).

4.1.3. 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 it is compound by 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 [6,19], 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 values to the ones indicated for each experiment. For the initial experiment we set the number of individuals (NI) to 10, the crossover value (Cr) to 0.1 and the mutation factor (F) to 0.5. For the DE scheme, we choose $DE/rand/1/bin$. Finally, the number of generations represents the terminal condition when the algorithm is executed and it is set to 1000.

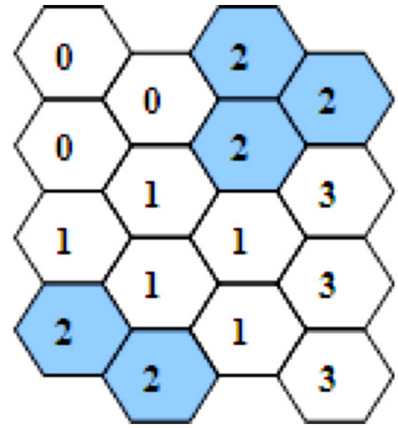

Fig. 5. Scattered location area (LA 2).

4.1.4. Individuals validation

Each individual that is generated on each population represents a network configuration. However we must take in consideration that it can represents an invalid configuration, because the application of the algorithm can produce individuals with scattered LAs. This means that we may have cells attributed to the same LA in distinct places of the network and that do not communicate between them, as shown in Fig. 5, but in reality that is not possible and we must correct or discard the individual. 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 on each generation, to assure that the final solution will be a valid one.

4.2. Reporting cells scheme

In this section we will start detailing the major considerations and decisions for the implementation of our experiments, using the reporting cells scheme. We present the test networks used, explain the fitness function implemented to evaluate the solutions obtained and expose the original definition of parameters.

4.2.1. Networks used

Also for the reporting cells strategy, most of other authors do not present the test networks used, in their studies, so it is not possible to compare our approach with them. However, in [13] it is presented a set of 12 networks, representing 4 groups defined by size, that have been generated, based on realistic data and patterns, and are available in [20] as benchmark. In this work we used these 12 networks with the objective of compare final results. In Table 4 it is shown, as an example, the test network 1 that represents a 4 × 4 cells configuration. The first column indicates the cell identification, the second column corresponds to the number of location updates NLU and the third represents the number of incoming calls NP.

Table 4
Test network 1 with 4 × 4 attributes.

Cell	NLU	NP
0	452	484
1	767	377
2	360	284
3	548	518
4	591	365
5	1451	1355
6	816	438
7	574	415
8	647	366
9	989	435
10	1105	510
11	736	501
12	529	470
13	423	376
14	1058	569
15	434	361

4.2.2. *Fitness function*

In the study of the reporting cells problem the fitness function is used for measuring the total location management cost of each potential solution, which is defined according to Eq. (5). This means that for each potential solution generated, it is calculated the fitness value, which corresponds to the network configuration by means of reporting cells and non-reporting cells.

4.2.3. *Parameters definition*

The initial definition of parameters is an important step because it represents the basis for the algorithm evolution. First it is defined the initial population of candidate solutions that corresponds to the individuals.

Each individual is compound by *N* genes, where the *N* value is the number of cells in the network and each gene represents the information about the cell type, which can be a reporting cell or a non-reporting cell.

To define the initial population we have set, with a probability of 50%, the type of each cell as RC or nRC.

Initially it is also necessary to set the DE algorithm parameters and that has been done with a number of individuals *NI* equal to 10, the crossover value *Cr* defined as 0.1 and the mutation factor *F* set to 0.5. For the DE scheme, the $DE/rand/1/bin$ has been selected. The number of generations, that is, the terminal condition, is set to 1000.

Throughout the different experiments, the parameters values have been adjusted with the specific objective of obtaining the best results.

5. Experimental results and analysis

In this section we expose the different experiments performed, the results obtained and the respective analysis and conclusions taken.

We will divide this section in two major subsections that will include, respectively, the results and analysis for the location areas and the reporting cells schemes.

5.1. *Location areas scheme*

In order to compare results, the values of always-update and never-update strategies were calculated for all the 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 [6,7], 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. After that, we applied the best configuration of DE to a realistic network based on SUMATRA data [2,16] with the objective of test our approach with major and realistic networks.

5.1.1. *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 [19,21]. 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 analyzed all the fitness values, including the best, worst, average, median and standard deviation results. We observed that until now the average of fitness values always presents a positive evolution, so because of that we decided to continue 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 5) 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 defined and used five levels of grey in the tables of results. The most dark grey (level 1) colour was used to mark the best fitness values. The dark grey (level 2) mark was used to point the best of the maximum (worst) fitness values. Then a grey mark (level 3) was used to show the best average fitness values and one light grey mark (level 4) was used to point the best median fitness value. Finally the most light grey (level 5) mark was used to highlight 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 stops or decreases, 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.

5.1.2. *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).

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 6), it is possible to conclude that really $Cr = 0.1$ is the best and more stable value to obtain better results.

Table 5
Experiment 1: defining the best *NI*.

Fitness evaluation													
NI	10	25	50	75	100	125	150	175	200	225	250	275	300
5 × 5 Network													
Best	27,216	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990
Worst	41,152	28,104	27,992	27,637	27,536	27,464	27,522	27,464	27,336	27,336	27,336	27,282	27,336
Average	28,992.7	27,518.4	27,281.4	27,292.2	27,264.7	27,191.0	27,148.6	27,119.9	27,119.5	27,149.0	27,100.9	27,104.7	27,062.6
Median	27,945.5	27,425.5	27,282	27,336	27,216	27,216	27,182	27,048	27,153	27,211	27,048	27,048	26,990
S.D.	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
5 × 7 Network													
Best	41,458	40,645	40,754	40,645	40,328	40,645	40,645	40,582	40,582	40,427	40,256	40,328	40,328
Worst	57,123	44,005	43,120	43,424	42,600	42,919	43,033	42,483	42,635	42,690	42,236	42,227	41,852
Average	44,493.7	42,415.6	42,188.1	42,043.6	41,638.6	41,752.8	41,542.3	41,393.0	41,385.9	41,545.6	41,313.1	41,289.4	41,016.0
Median	42,925.5	42,079	41,893.5	41,662.5	41,346	41,488	41,484.5	41,217	41,340	41,465	41,340	41,303	41,080.5
S.D.	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
7 × 7 Network													
Best	65,331	64,362	65,153	64,879	64,879	64,674	64,161	64,732	64,477	64,433	65,458	64,043	63,958
Worst	109,549	69,760	68,273	68,280	68,628	68,348	67,525	67,606	67,869	67,414	67,624	67,359	66,986
Average	71,501.9	67,907.9	67,228.8	66,830.5	66,803.1	66,443.1	66,252.9	66,166.3	66,264.2	65,996.5	66,466.7	65,657.9	65,873.7
Median	69,052.5	68,036.5	67,401	67,033	66,914	66,523	66,347.5	66,172.5	66,404	66,041.5	66,506.5	65,566	65,963.5
S.D.	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
7 × 9 Network													
Best	96,277	95,296	95,969	97,440	95,246	95,640	94,304	96,329	94,908	95,110	94,293	94,888	95,080
Worst	106,179	102,941	101,975	101,204	100,589	101,035	100,794	100,774	100,014	99,855	99,829	99,475	99,375
Average	102,158.6	100,379.7	99,699.3	99,268.4	98,848.2	98,567.7	98,511.7	98,098.4	97,800.3	97,955.1	97,686.8	97,413.1	97,589.5
Median	102,791	100,588	99,841	99,466.5	99,463	98,701	98,768.5	98,094.5	97,934	98,280	97,647.5	97,568	97,861.5
S.D.	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

5.1.3. *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.

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 7, 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.

5.1.4. *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

Table 6
Experiment 2: defining the best *Cr*.

Fitness evaluation										
Cr	0.01	0.03	0.05	0.07	0.09	0.10	0.25	0.50	0.75	0.90
5 × 5 Network										
Best	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990	26,990
Worst	27,499	27,512	27,398	27,336	27,464	27,336	27,216	27,336	27,398	27,398
Average	27,249.0	27,277.3	27,159.5	27,145.8	27,107.2	27,070.6	27,038.6	27,090.4	27,086.5	27,136.1
Median	27,213.5	27,215.5	27,211	27,211	27,048	26,990	26,990	27,048	27,048	27,048
S.D.	131.8	112.9	141.5	119 7	127.1	106.2	76.8	111.2	129.7	146.5
5 × 7 Network										
Best	40,672	40,645	40,645	40,645	40,525	40,301	40,466	40,301	41,465	42,219
Worst	43,359	43,114	42,765	42,424	42,433	42,086	42,056	42,318	42,919	43,318
Average	41,661.7	41,552.2	41,674.8	41,763.8	41,405.4	41,188.6	41,228.7	41,398.2	42,384.7	42,616.6
Median	41,610	41,475.5	41,721.5	41,868.5	41,354	41,138	41,227.5	41,474	42,403	42,571
S.D.	627.2	486.6	586.8	449.0	536.6	474.9	416.0	486.0	219.5	278.5
7 × 7 Network										
Best	64,769	65,030	64,729	63,815	63,534	63,874	64,674	64,305	67,380	67,232
Worst	68,829	68,010	67,382	67,819	67,926	67,567	67,055	68,852	70,191	72,662
Average	66,819.0	66,739.1	66,368.1	66,185.7	66,029.1	66,057.7	65,915.5	67,396.8	68,937.2	69,361.9
Median	66,642	66,817.5	66,512	66,208	66,027	66,009	65,950.5	67,836.5	69,097	69,091
S.D.	876.1	703.7	638.1	956.5	1041.1	974.6	621.4	1077.1	690.7	1246.8
7 × 9 Network										
Best	95,487	93,285	94,402	95,208	95,565	95,492	96,979	97,884	101,417	103,666
Worst	102,145	101,299	100,284	99,687	100,181	99,373	101,678	107,827	109,540	107,819
Average	100,178.8	98,751.6	98,362.5	98,015.2	97,943.1	97,547.0	99,332.5	103,542.7	105,689.5	105,707.1
Median	100,498.5	98,883	98,762	97,967.5	98,096	97,441	99,477.5	104,316.5	105,741	105,755.5
S.D.	1313.7	1730.6	1446.9	1064.6	1021.1	1015.4	920.7	2467.3	1779.6	988.0

Table 7
Experiment 3: defining the best F.

Fitness evaluation					
F	0.01	0.25	0.50	0.75	0.90
5 × 5 Network					
Best	26,990	26,990	26,990	26,990	26,990
Worst	27,512	27,336	27,398	27,336	27,336
Average	27,080.6	27,072.7	27,141.8	27,134.3	27,078.4
Median	27,048	27,019	27,211	27,153	27,019
S.D.	123.7	106.8	125.5	112.8	114.2
5 × 7 Network					
Best	40,473	40,466	40,328	40,496	40,328
Worst	42,801	42,600	42,045	42,440	42,403
Average	41,491.0	41,221.5	41,194.3	41,287.8	41,266.3
Median	41,340	41,138	41,183.5	41,318.5	41,261.5
S.D.	558.2	536.1	447.2	477.9	475.3
7 × 7 Network					
Best	64,893	64,893	64,671	64,879	64,207
Worst	67,141	67,796	67,338	67,448	67,111
Average	66,051.2	66,140.0	66,192.1	65,993.3	65,981.9
Median	66,020	66,036.5	66,066	65,971	66,003.5
S.D.	554.3	749.8	602.5	679.3	790.2
7 × 9 Network					
Best	96,220	95,076	93,040	94,774	95,105
Worst	100,336	99,779	100,211	99,355	100,118
Average	98,116.6	97,821.8	97,826.5	97,884.3	97,849.8
Median	98,129	98,051.5	97,952	97,910.5	97,966.5
S.D.	1022.6	1281.1	1402.2	925.5	1213.3

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 10 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 8), 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 $DE/rand/1/bin$.

Furthermore, a statistical analysis using the ANOVA test has been performed. We consider here a confidence level of 95% (i.e., significance level of 5% or p-value under 0.05), which means that the differences are unlikely to have occurred by chance with a probability of 95%. Using this test we have obtained the results included in Table 9, where we can see that the fitness differences when we use distinct values for each DE parameter have been found as significant in almost all the cases.

Table 8
Experiment 4: defining the best DE scheme.

Fitness evaluation										
Scheme	Exponential crossover					Binomial crossover				
	Best1	Rand1	RandToBest1	Best2	Rand2	Best1	Rand1	RandToBest1	Best2	Rand2
5 × 5 Network										
Best	27,282	26,990	27,048	27,211	27,211	27,048	26,990	27,048	26,990	26,990
Worst	28,247	28,144	28,485	28,242	28,242	27,536	27,336	28,036	27,637	27,398
Average	27,871.2	27,620.4	27,946.3	27,784.2	27,572.8	27,304.9	27,077.8	27,420.6	27,291.4	27,102.7
Median	28,008	27,586.5	28,052	27,824	27,524	27,279	27,048	27,431	27,282	27,019
S.D.	305.0	297.3	395.9	354.6	295.9	132.9	111.4	255.5	169.4	125.6
5 × 7 Network										
Best	41,141	41,141	40,722	40,722	41,340	40,706	40,205	40,645	40,346	40,525
Worst	45,077	45,372	44,623	43,972	44,683	42,720	42,600	43,560	43,108	42,305
Average	42,772.1	42,499.1	42,853.6	42,497.0	42,542.5	41,692.0	41,261.8	41,917.4	41,627.5	41,351.4
Median	42,610.5	42,094	42,784	42,154	42,124	41,720	41,141	41,843	41,661	41,327.5
S.D.	987.1	1010.5	1136.6	794.4	927.9	400.8	562.2	676.1	600.4	387.2
7 × 7 Network										
Best	66,215	65,281	66,243	65,188	65,658	64,625	63,307	64,890	64,560	65,290
Worst	71,240	70,336	70,282	71,856	70,149	68,140	67,398	68,623	67,676	67,200
Average	67,709.3	67,510.0	68,417.0	67,708.2	67,367.4	66,366.3	65,737.1	66,976.2	66,247.6	66,273.2
Median	67,568	67,101.5	68,827	67,520	66,668	66,344.5	65,711	66,924	66,345	66,217.5
S.D.	1093.5	1330.8	1276.8	1399.4	1126.5	828.3	854.2	893.7	790.7	520.6
7 × 9 Network										
Best	100,386	100,484	98,967	99,512	100,295	95,125	94,841	96,408	92,900	94,483
Worst	105,524	106,960	106,126	106,188	105,434	99,473	99,735	100,101	100,587	99,635
Average	102,580.6	103,191.6	103,152.4	103,199.8	103,100.8	97,947.5	97,895.1	98,346.4	97,479.8	97,598.4
Median	102,479.5	103,116.5	102,942.5	102,884	103,179.5	97,896.5	97,874.5	98,238.5	97,730	97,745.5
S.D.	1166.4	1212.2	1643.1	1751.7	1213.0	996.4	1275.4	945.2	1408.7	1285.8

Table 9
ANOVA analysis over DE parameters in the LAs problem.

NI parameter				
Network	5 × 5	5 × 7	7 × 7	7 × 9
p-Value	2.10E−14	<1E−15	2.82E−14	<1E−15
Cr parameter				
Network	5 × 5	5 × 7	7 × 7	7 × 9
p-Value	2.78E−15	<1E−15	<1E−15	<1E−15
F parameter				
Network	5 × 5	5 × 7	7 × 7	7 × 9
p-Value	5.33E−02	1.81E−01	7.15E−01	8.66E−01

5.1.5. *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. 6.

Comparing with studies of other authors, as Taheri and Zomaya [6,7,22], that use respectively genetic algorithms, simulated annealing and hopfield neural network approaches, our results are very similar and in some cases even better.

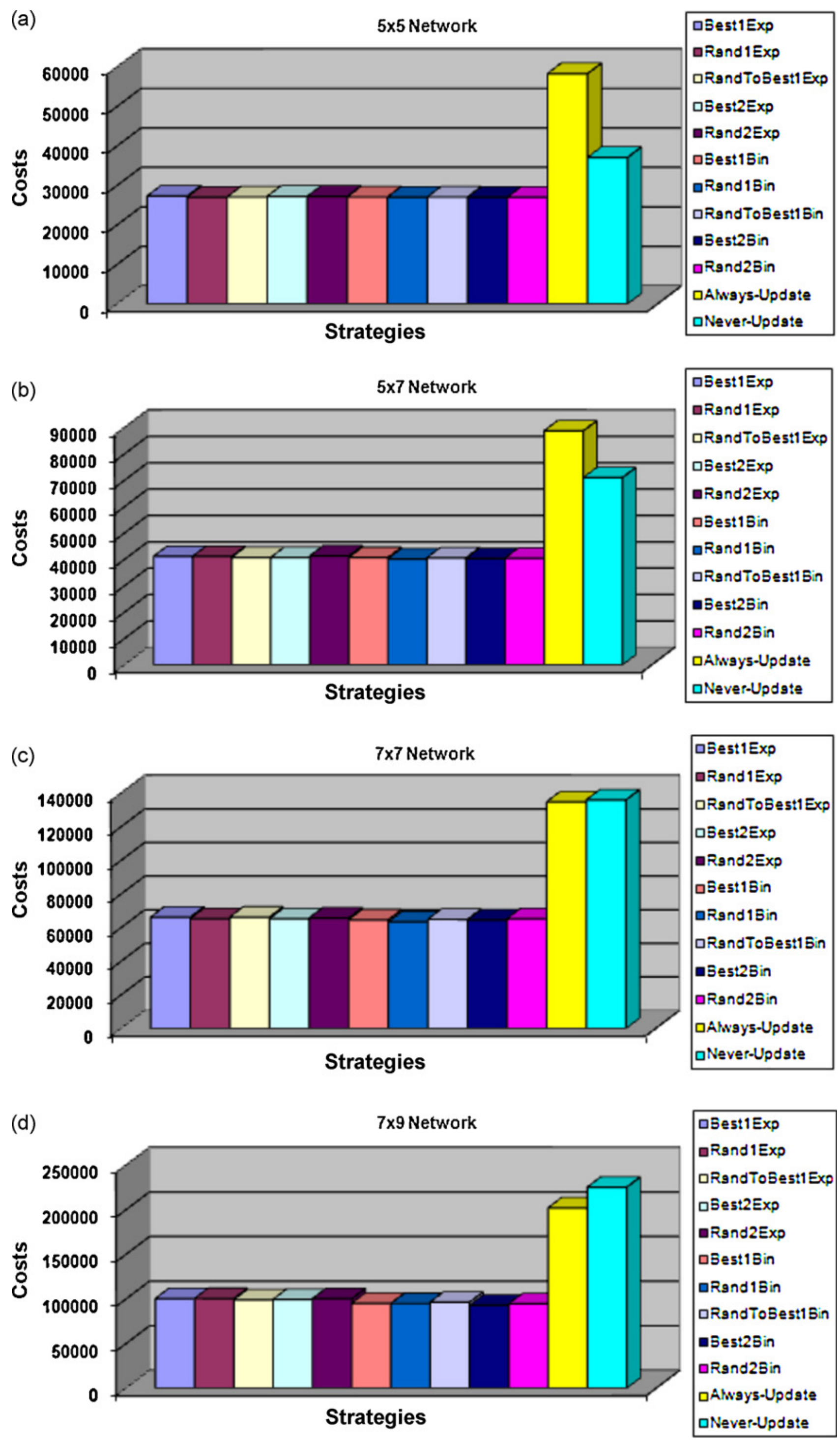

Fig. 6. Comparison results: (a) 5 × 5 network, (b) 5 × 7 network, (c)7 × 7 network, and (d)7 × 9 network.

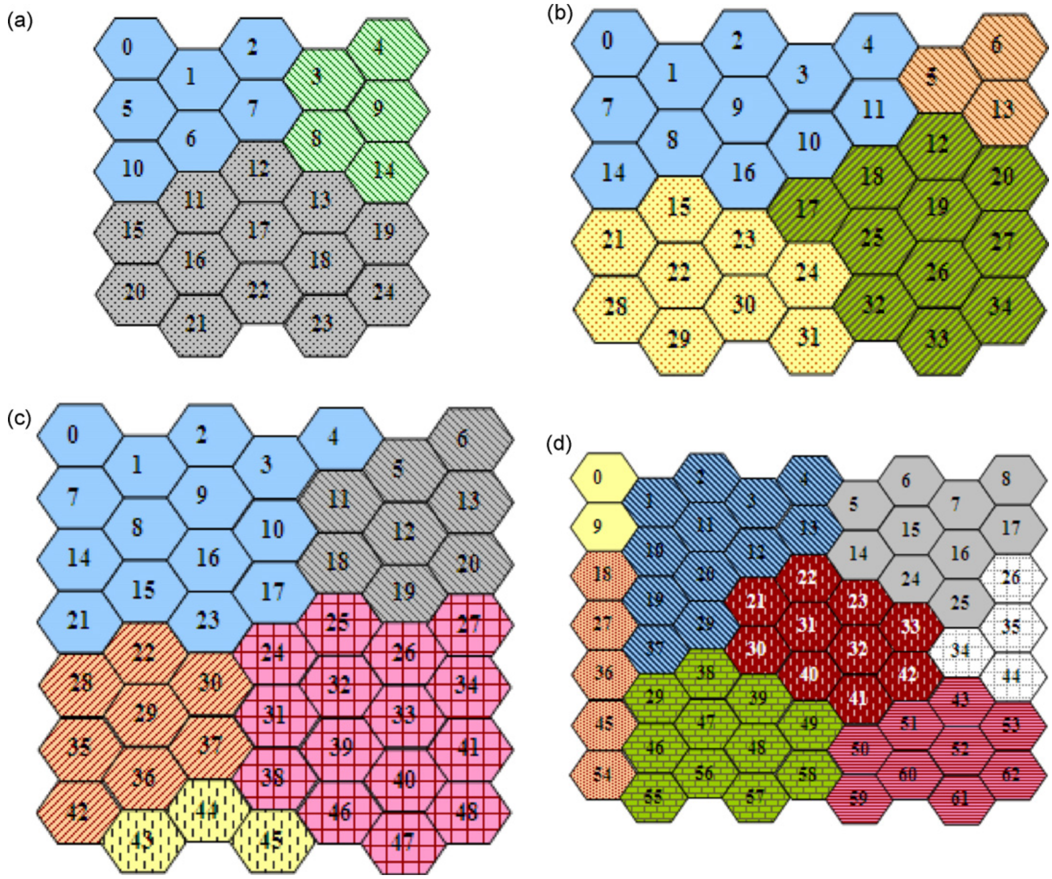

Fig. 7. Best LAs configuration: (a) 5 × 5 network, (b) 5 × 7 network, (c) 7 × 7 network, and (d) 7 × 9 network.

For example, for the 5 × 5 network, our best fitness solution corresponds to a cost of 26,990 and their best result is between 25,000 and 30,000. Using the 5 × 7 network, our best fitness solution represents a cost of 40,205 and their results are between 40,000 and 45,000. In the 7 × 7 network our lower cost is 63,307 and their best value is between 60,000 and 65,000. Finally, for the 7 × 9 network the best fitness value obtained by our approach is 92,900 and their best solution is between 90,000 and 95,000.

All of these costs were calculated with the network partitioning defined by the DE algorithm and represented in Fig. 7.

With respect to the ideal number of location areas, we observed that, for the 5 × 5 network, all the best solutions correspond to a network partitioning in 3 location areas (see Fig. 7(a)). When we refer to the 5 × 7 network, the best solution corresponds to a partitioning in 4 distinct LAs (see Fig. 7(b)). Moving to the 7 × 7 network, the ideal partitioning is represented in 5 LAs (see Fig. 7(c)). For the bigger network, the 7 × 9, the best configuration corresponds to a partitioning in 8 LAs (see Fig. 7(d)).

Relatively to the shape of the LAs, the most of them do not have a circular shape, as in the actual GSM systems. Their

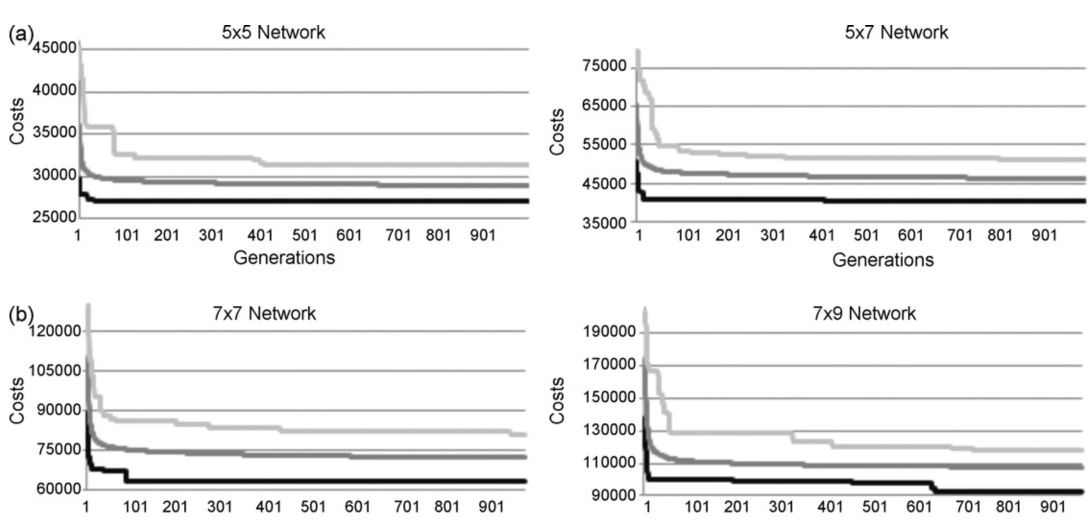

Fig. 8. Convergence curves of DE applied to the LA networks. The *Y* axis corresponds to the LM cost values and the *X* axis corresponds to the generations. Each graphic shows the evolution of the best (black line), the worst (grey line) and the average solution (light grey line).

Table 10
CPU time (s) of LA experiments.

Network	Average	Best
5 × 5	41	40
5 × 7	60	60
7 × 7	89	87
7 × 9	124	122

forms are diverse but principally of triangular or rectangular shape.

In Fig. 8 we present the convergence curves of DE, using its best configuration, for each test network. We can observe that there is a big convergence since the earlier generations. In each graphic the black line corresponds to evolution of the best fitness value (the solution with the lowest cost). The grey line represents the evolution of the average fitness value. Finally, the light grey line shows the evolution of the worst fitness value.

Considering the computational CPU time we may say that it depends of the networks, principally by their size and data associated. In Table 10 we present the best and the average CPU time for each of the four networks used in the executions of the experiment 4 (those with the best configuration of DE parameters elected), because it is the most real one. Furthermore we must explain that the LAs calculi are not executed in a continuous form (i.e., in real time for each call). Each calculus is performed previously during the configuration of the network (partitioning of the network in LAs) and this configuration is maintained during all the time that the network is used (while the users make and receive calls).

5.1.6. *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 [6,23] 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 test networks using 5000 generations as stop criterion.

In Table 11 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.

Now, in Table 12 we compare the new results with the ones presented by Taheri and Zomaya in [23], 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 [23]), in the most of the cases the results are similar or even better.

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.

Table 11
Evolution of results over 5000 generations.

Test network (Dim)	Generation				
	1000	2000	3000	4000	5000
5 × 5	26,990	26,990	26,990	26,990	26,990
5 × 7	40,205	40,117	40,085	40,085	39,859
7 × 7	63,307	62,720	61,951	61,567	61,037
7 × 9	92,900	91,104	90,687	90,437	89,973

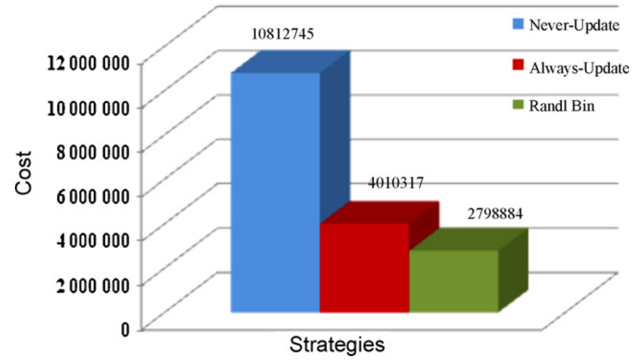

Fig. 9. Comparison of strategies results.

5.1.7. *SUMATRA: using realistic networks*

After determining the best configuration of DE with the four previous experiments, we decided to apply it to one realistic network, based on SUMATRA data [2,16] and using the BALI-2 trace.

If we compare our results obtained, using this configuration, with the classical strategies always-update and never-update we can say that our approach always obtains better solutions (lower costs), as it is possible to see in Fig. 9.

Comparing our results with studies of other authors, as Subrata and Zomaya [17], the results are very close to, which is very interesting, considering that we are using static LAs and they use dynamic LAs. In fact, our results are very similar to the ones obtained with important dynamic LA strategies like DBLA (Distance-Based Location Area). For example, our best result is 2,798,884 cost units, and DBLA obtains 2,695,282 cost units [17] for this network.

In Fig. 10 we present the analysis of the hourly location management (LM), for the classical strategies and for our best result. It is possible to conclude that our approach obtains lower location management costs than the classical ones.

Once more, if we compare these costs per hour with the ones of Subrata and Zomaya [17], they are very competitive, and this gives us the notion that our approach is viable.

5.2. *Reporting cells scheme*

In this section we explain the four distinct experiments applied to each test network with the objective of studing in more detail the best configuration of DE, applied to the reporting cells problem. For each experiment, and for all combination of parameters,

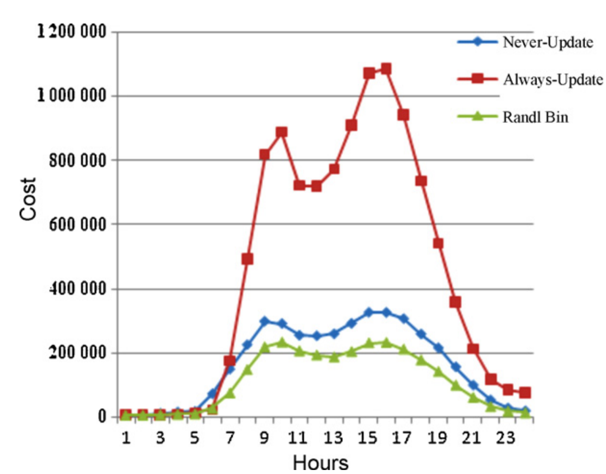

Fig. 10. Hourly LM cost comparison.

Table 12
Comparison of network costs with different algorithms.

Test network (Dim)	Algorithm						
	DE	GA	HNN	SA	GA-HNN1	GA-HNN2	GA-HNN3
5 × 5	26,990	28,299	27,249	26,990	26,990	26,990	26,990
5 × 7	39,859	40,085	39,832	42,750	40,117	39,832	39,832
7 × 7	61,037	61,938	63,516	60,694	62,916	62,253	60,696
7 × 9	89,973	90,318	92,493	90,506	92,659	91,916	91,819

Table 13
Experiment 1: determining the best *NI*.

Test network (N. Dim)	*NI*– fitness evaluation									
	10	25	50	75	100	125	150	175	200	225
1 (4 × 4)	99,137	100,881	98,535	98,535	98,535	98,535	98,535	98,535	98,535	98,535
2 (4 × 4)	101,250	98,879	97,156	97,156	97,156	97,156	97,156	97,156	97,156	97,156
3 (4 × 4)	98,106	101,403	95,038	95,038	95,038	95,038	95,038	95,038	95,038	95,038
4 (6 × 6)	195,283	185,092	176,800	173,701	173,701	173,701	173,701	173,701	173,701	173,701
5 (6 × 6)	205,859	192,426	185,937	182,331	182,331	182,331	182,331	182,331	185,059	182,331
6 (6 × 6)	193,540	186,423	175,321	174,519	174,519	174,519	174,519	174,519	174,519	174,519
7 (8 × 8)	338,196	321,575	315,097	310,888	308,853	309,342	308,401	308,401	308,991	308,401
8 (8 × 8)	319,912	304,750	294,548	287,149	289,051	287,149	287,149	287,149	289,935	287,149
9 (8 × 8)	292,467	276,299	270,171	265,272	264,204	264,204	264,204	264,316	264,204	264,204
10 (10 × 10)	425,866	405,127	394,168	388,206	386,775	387,551	386,695	386,474	387,543	386,893
11 (10 × 10)	390,793	377,363	361,299	360,210	361,581	359,224	358,778	359,224	359,697	358,944
12 (10 × 10)	401,704	385,022	380,846	375,233	376,631	374,711	375,001	375,722	373,733	374,220

30 independent runs have been performed in order to assure its statistical relevance. In each experiment the final results, of the best fitness values obtained (lower location management cost value), are presented and explained the decisions taken.

After that we analyze the results obtained, compare them with those shown in [13] and present the configuration for best solutions.

Finally we apply the best DE configuration, achieved to the RC problem, to the test networks presented in [9,10] and compare results produced with this DE configuration, with the ones accomplished by other artificial life techniques.

5.2.1. Experiment 1 – determining NI

The number of individuals that will compound the initial population must be the first experiment because it is the basis of the algorithm implementation. In order to accomplish that, we have fixed, as referred in 4.2.3 *Parameters definition*, the values of crossover $Cr = 0.1$, the mutation $F = 0.5$, DE scheme as $DE/rand/1/bin$ and the stop criterion as 1000 generations, considering our experience from earlier experiments that we have performed [19,21].

With this experiment we have concluded that, increasing of NI value, it is possible to observe a positive evolution of the results (see fitness results in Table 13), but just until the value of $NI = 175$, because after that we start observing worse results and stop increase in $NI = 225$. Considering this and the average evolution we concluded that $NI = 175$ would be the elected value for the second experiment.

5.2.2. Experiment 2 – determining Cr

The second experiment has the objective of selecting the *Cr* value that obtains the best results. To proceed with this experiment we fixed the value of *NI* to 175 (from experiment 1), and maintained the other parameters as defined in the beginning of experiment 1.

This experiment has been executed using different values for *Cr*: 0.1, 0.25, 0.50, 0.75 and 0.90. Analyzing the results obtained, we could conclude that best values were obtained with $Cr = 0.1$ and $Cr = 0.25$. Because of that, with the objective of taking more complete conclusions, we decided to execute the algorithm with values 0.15 and 0.20. Finally, as it is possible to see in Table 14, we could conclude that $Cr = 0.15$ is the one that performs better.

5.2.3. Experiment 3 – determining F

The determination of the best value for mutation, *F*, is the purpose of the third experiment. So, in order to perform this exper-

Table 14
Experiment 2: determining the best *Cr*.

Test network (N. Dim)	*Cr*– fitness evaluation						
	0.1	0.15	0.20	0.25	0.50	0.75	0.90
1 (4 × 4)	98,535	98,535	98,535	98,535	98,535	98,535	98,535
2 (4 × 4)	97,156	97,156	97,156	97,156	97,156	97,156	97,258
3 (4 × 4)	95,038	95,038	95,038	95,038	95,038	95,038	98,216
4 (6 × 6)	173,701	173,701	173,701	173,701	173,701	177,647	177,889
5 (6 × 6)	182,331	182,331	187,990	183,264	184,679	183,991	185,966
6 (6 × 6)	174,519	174,519	174,519	175,321	175,182	175,321	178,255
7 (8 × 8)	308,401	308,401	308,401	311,646	313,378	313,607	319,069
8 (8 × 8)	287,149	287,149	289,573	289,051	293,248	302,812	309,609
9 (8 × 8)	264,204	264,204	265,452	264,786	272,249	266,876	275,489
10 (10 × 10)	387,318	386,681	388,357	386,959	393,510	393,492	420,650
11 (10 × 10)	360,262	358,669	360,072	360,128	360,596	367,508	374,405
12 (10 × 10)	373,695	374,966	374,554	374,921	377,190	383,782	391,001

Table 15
Experiment 3: determining the best F.

Test network (N. Dim)	F– fitness evaluation				
	0.1	0.25	0.50	0.75	0.90
1 (4 × 4)	98,535	98,535	98,535	98,535	98,727
2 (4 × 4)	97,156	97,156	97,156	97,156	97,156
3 (4 × 4)	95,038	95,038	95,038	95,038	95,038
4 (6 × 6)	174,112	173,701	173,701	173,701	176,530
5 (6 × 6)	182,331	182,331	182,331	182,331	182,331
6 (6 × 6)	174,519	175,321	174,519	174,519	174,519
7 (8 × 8)	310,162	310,426	308,401	311,492	308,401
8 (8 × 8)	293,093	304,911	287,149	292,913	295,557
9 (8 × 8)	265,494	264,643	264,204	268,312	265,750
10 (10 × 10)	388,849	389,438	386,681	389,125	387,533
11 (10 × 10)	359,221	360,072	358,669	358,167	361,441
12 (10 × 10)	373,298	375,087	374,966	371,829	375,232

iment it was fixed the value of NI to 175 (from experiment 1), the value of Cr to 0.15 (from experiment 2) and the others maintained as in the two earlier experiments.

After finishing these executions and examining the results (see Table 15) we conclude that $F = 0.5$ is the one that permits to obtain the best results.

5.2.4. *Experiment 4 – determining DE scheme*

Finally, with this fourth experiment we pretend to select the most adequate DE scheme, the one that permits to obtain the best results (lower fitness value). For that, we fixed the best values for each parameter (defined in the three earlier experiments) as: $NI = 175$, $Cr = 0.15$ and $F = 0.5$, and executed the algorithm applying the 10 DE schemes presented in Section 3.5.

Once finished all the executions, and observing the respective results shown in Table 16, it was possible to conclude that the scheme $DE/rand/1/bin$ is the one with a better performance, because it is the one that obtains better fitness values for all the test networks. With these results we may say that the binomial schemes perform better than the exponential ones and that it is also better to choose randomly the individuals used to create the trial individual.

Finishing these four experiments we had determined the best DE configuration, applied to the reporting cells planning problem, setting the parameters as $NI = 175$, $Cr = 0.15$, $F = 0.5$ and $DE/rand/1/bin$ as the most adequate DE scheme.

Also for the RC experiments we have performed a statistical analysis using the ANOVA test. Similar to the LA experiments we consider a confidence level of 95%. In Table 17 we show the results obtained with this test, using the distinct values for each DE parameter. Once again we can observe that the fitness differences have been found significant in almost all the cases.

5.2.5. *Analysis and comparison of results*

Analyzing the experimental results we could conclude that with this approach it is possible to obtain the same minimum fitness values (considered optimal in [13]), as the ones obtained in [13] with a Hopfield Neural Network with Ball Dropping (HNN+BD) and a Geometric Particle Swarm Optimization (GPSO) for 10 of the 12 test networks used.

For the other two test networks the results are very similar because: for the test-network-10 our fitness value is 386,951 and in [13] the one obtained by the HNN+BD is 386,351; and for the test-network-12 our fitness value is 371,829 and in [13] the value obtained by HNN+BD and GPSO is 370,868. Relatively to the average values it is possible to say that they are very similar.

In Fig. 11 the configuration for each test-network solution is shown and it is possible to observe that most of them split each one in subnetworks.

The convergence curves of DE relatively to each test-network solution are presented in Fig. 12. In all the graphics we can observe the best solution (black line), the worst (grey line) and the average solution (light grey line). In all the cases we can see a good convergence.

Table 16
Experiment 4: determining the DE scheme.

Test network (N. Dim)	DE scheme – fitness evaluation									
	Exponential crossover					Binomial crossover				
	Best1	Rand1	RTB1	Best2	Rand2	Best1	Rand1	RTB1	Best2	Rand2
1 (4 × 4)	98,535	98,535	98,535	98,535	99,008	98,535	98,535	98,535	98,727	98,535
2 (4 × 4)	97,156	97,156	97,156	97,156	97,156	97,156	97,156	97,156	97,156	97,156
3 (4 × 4)	95,038	95,038	95,038	95,038	95,038	95,038	95,038	95,038	95,038	95,038
4 (6 × 6)	173,701	173,701	173,701	176,530	178,038	176,041	173,701	173,701	173,701	173,701
5 (6 × 6)	182,331	182,331	187,801	190,779	191,279	182,331	182,331	182,331	182,331	182,331
6 (6 × 6)	174,519	175,182	183,992	177,276	177,892	181,850	174,519	174,519	174,519	174,519
7 (8 × 8)	322,973	319,772	328,327	323,391	332,472	320,236	308,401	308,401	309,855	308,730
8 (8 × 8)	304,214	307,139	313,010	310,708	316,849	305,236	287,149	287,149	287,149	287,149
9 (8 × 8)	277,408	279,177	290,646	291,684	289,936	269,984	264,204	264,316	265,164	264,353
10 (10 × 10)	420,701	423,017	420,452	421,353	425,228	394,176	386,681	386,951	393,471	386,695
11 (10 × 10)	385,950	380,824	384,661	387,273	380,241	366,156	358,167	359,486	367,202	359,517
12 (10 × 10)	394,636	388,468	395,290	395,404	394,767	379,227	371,829	376,015	379,544	376,165

Table 17
ANOVA analysis over DE parameters in the RCs problem.

NI parameter												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
p-Value	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15
Cr parameter												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
p-Value	5.67E−07	2.63E−08	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15	<1E−15
F parameter												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
p-Value	0.634	0.907	0.110	0.582	0.739	0.0658	0.0617	<1E−15	0.187	0.213	1.18E−04	2.22E−05

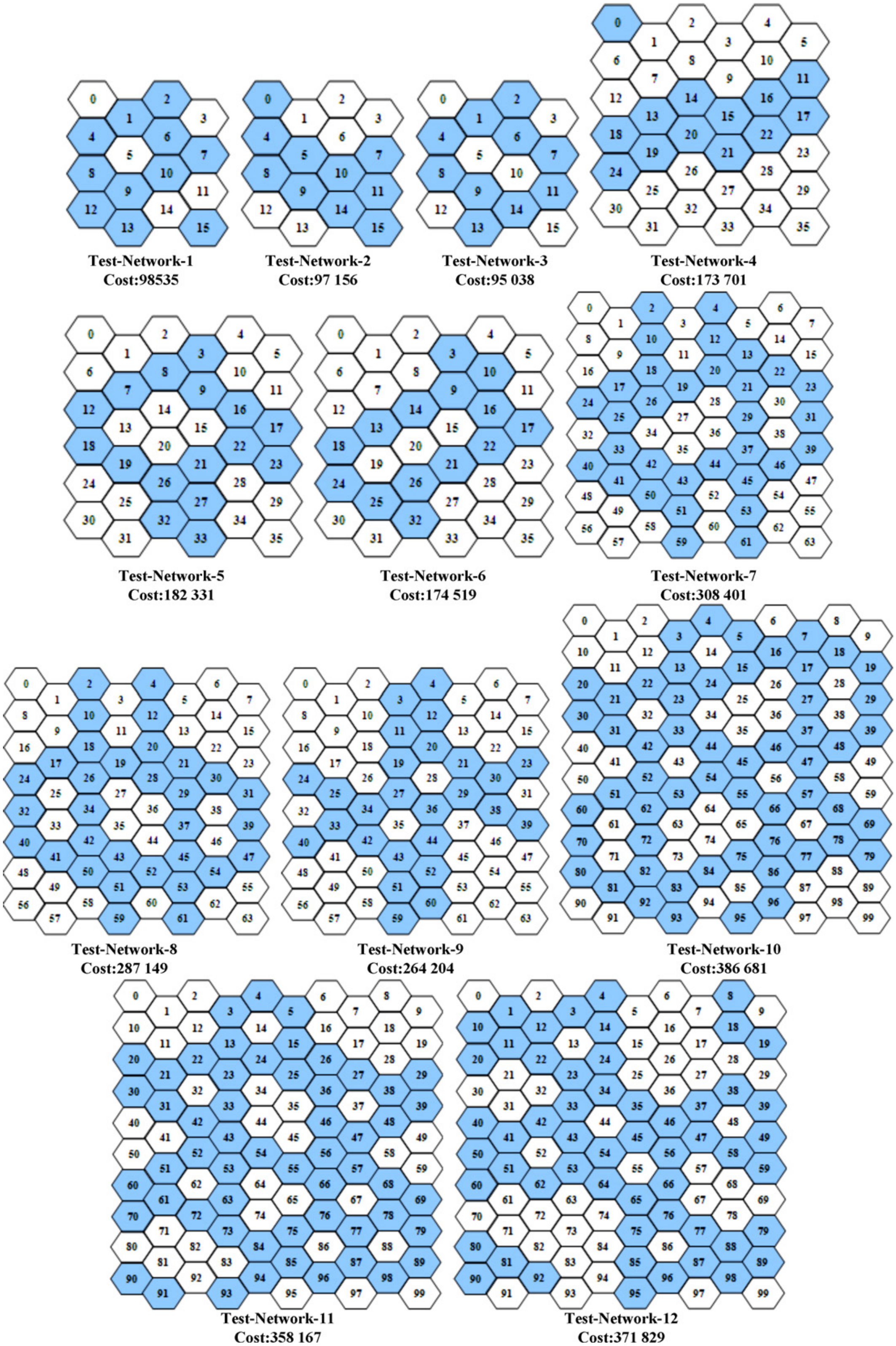

Fig. 11. Test-network solution with reporting cells configuration.

Table 18 shows the best and the average CPU time of each of the 12 networks, relatively to the executions of the fourth experiment, considering the best configuration of DE parameters. Like in the LAs, we must explain that the RCs planning calculi are not executed in real time for each call. Each calculus is performed previously during the configuration of the network (division of the network in RCs and nRCs) and this one is maintained during all the time that the network is used.

5.2.6. Comparison with other artificial life techniques

After obtaining the best DE configuration applied to the reporting cells planning problem we decide to apply it in other test

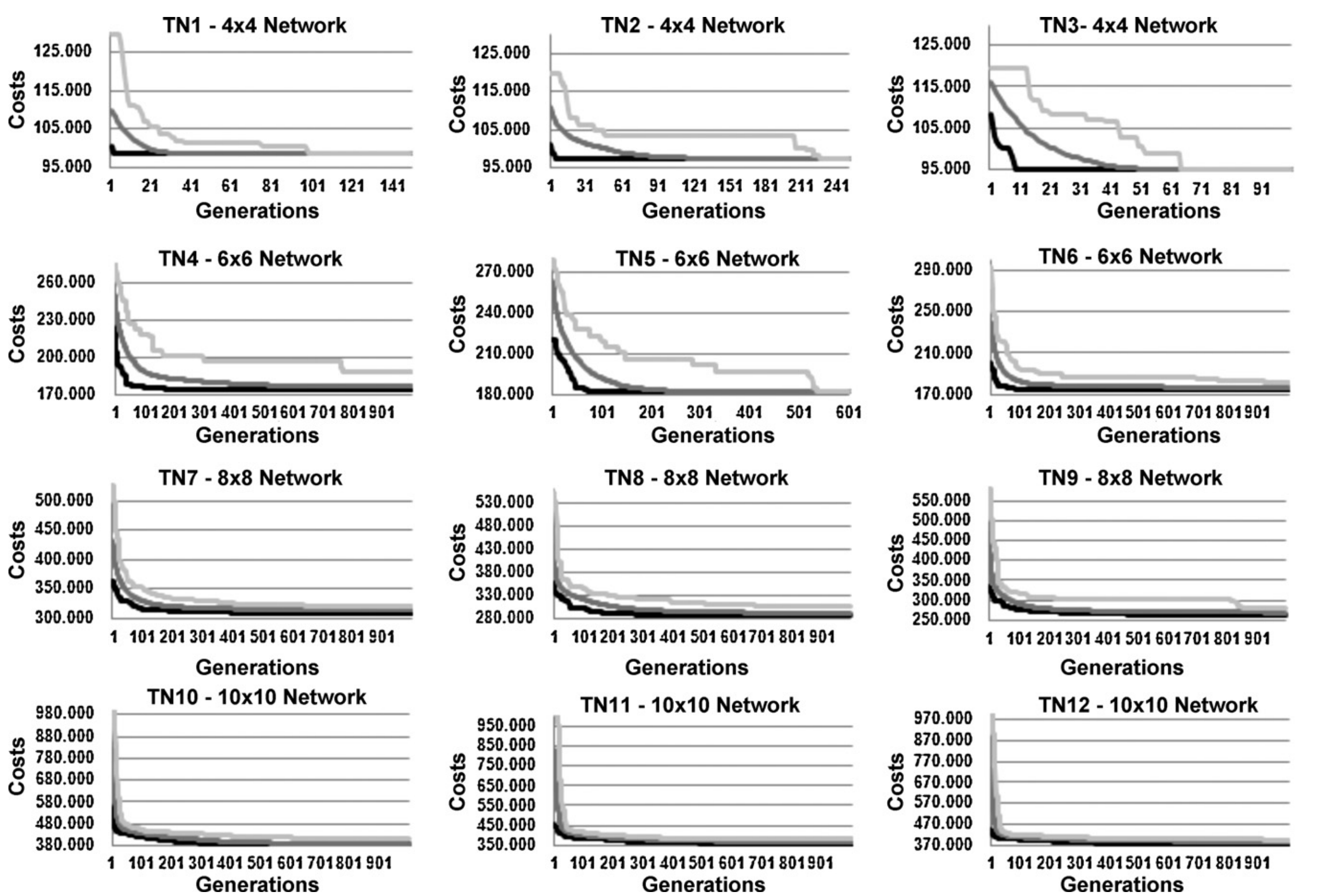

Fig. 12. DE convergence curves: LM cost values (*Y* axis) vs. generations (*X* axis). Each graphic shows the evolution of the best (black line), the worst (grey line) and the average solution (light grey line).

networks, and analyze the performance with the objective of comparing with other artificial life techniques. We decided to use three test networks presented in [9] (also referred in [13]) in order to compare our results with those produced by the application of genetic algorithms (GA), tabu search (TS) and ant colony algorithm (AC).

Making the additional experiments we conclude that our approach performs well. Using the Test-Network-1 (4×4 instance provided in [9]) we obtain the same fitness value as the GA, TS and AC, that is, 92,883 with a total of 10 reporting cells.

Our approach generates a better solution when solving the Test-Network-2 (6×6 instance provided in [9]), comparing with GA and AC. The fitness value obtained by the GA is 229,556 with a total of 26 reporting cells in the network [9,13], while, the cost obtained by DE in this work is 211,278 with 24 reporting cells. TS presents the same cost 211,278 and the fitness value obtained by AC is 211,291.

For the Test-Network-3 (8×8 instance in [9]) DE again surpasses the results obtained in [9] by the GA, TS and AC. Specifically, the fitness value obtained by the GA and TS is 436,283, the one obtained by AC is 436,886 while, the cost obtained by DE in this work is 436,269 with a total of 39 reporting cells.

Finally, we applied the best DE configuration in two networks presented in [10] that applies a combination of Hopfield Neural Network and Ball Dropping Technique (HNN-BDT). These test networks are considered to be produced using sophisticated routines used to generate the users' behaviour, in way to match real world traffic and reflect realistic simulations [10].

Again, with these additional experiments, we conclude that our approach presents a good performance. Using the first test network (7×9 instance) provided in [10] and also used in [24] we concluded that our approach accomplish a better solution, because our best fitness value is 120,904, with 28 RCs and the fitness value obtained by the HNN-BDT [10,24] is 123,474 with 27 RCs.

For the second test network (9×11 instance provided in [10]), the cost obtained by DE with this work is 243,957, with 47 RCs, which is very similar to the one obtained by the HNN-BDT in [10] that presents a fitness value of 243,414 units of cost with 43 RCs.

After all of these additional experiments we can conclude that the configuration achieved for DE configuration, to the reporting cells problem, permits our approach to be very competitive when compared with the results obtained with other artificial life techniques.

Table 18
CPU time (s) of RC experiments.

Network	Average	Best
TN1-4 × 4	19	16
TN2-4 × 4	21	16
TN3-4 × 4	20	15
TN4-6 × 6	110	91
TN5-6 × 6	68	58
TN6-6 × 6	77	71
TN7-8 × 8	140	133
TN8-8 × 8	139	125
TN9-8 × 8	163	148
TN10-10 × 10	187	169
TN11-10 × 10	203	192
TN12-10 × 10	232	211

6. Conclusions and future work

In this paper we present two approaches based on the differential evolution (DE) algorithm applied to the location management problem with the objective of minimizing the involved costs. Each approach is specified, respectively, for the location areas (LAs) and the reporting cells (RCs) strategies of location management problem.

Considering the LAs based approach, we have shown that it 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. Also when we apply our approach to realistic networks, using static LAs, the results are very similar when compared with the ones of other authors, which are using dynamic LAs.

Furthermore, we have studied in detail the best configuration of DE, applied to the LAs problem, and the best parameters, after a big number of experiments (more than 5000 independent runs) 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.

If we refer to the approach developed to be applied to the RCs problem we may say that we have studied in detail the best configuration of DE including parameters and scheme. After more than 10,000 runs, they are $NI = 175$, $Cr = 0.15$, $F = 0.5$ and $DE/rand/1/bin$ as the best DE scheme.

We have shown that our approach produces interesting results because when compared with the ones of other authors, that use HNN+BD and GSPO, they are equal or very similar.

Furthermore, comparing the performance of DE algorithm with other artificial life techniques as genetic algorithm (GA), tabu search (TS), ant colony algorithm (AC), and hopfield neural network and ball dropping technique (HNN-BDT) we may say that it performs well because improves the results obtained by those ones.

As future work we have the intention of applying other evolutionary algorithms to the LA and RC problems and making the comparison of their results with the ones accomplished by the DE algorithm. Finally, the formulation of the LA and RC problems as multiobjective optimization problems will be investigated as well.

Acknowledgments

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

Appendix A. Supplementary Data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.asoc.2009.11.031.

References

[1] K. Pahlavan, A. Levesque, Wireless Information Networks, John Wiley & Sons Inc., Wiley-Interscience, New York, 1995.
[2] SUMATRA, 1998, Stanford University Mobile Activity TRAces (online), InfoLab – Stanford University, Stanford. Available from: http://infolab.stanford.edu/sumatra/ (accessed February 2009).
[3] V. Wong, V. Leung, Location management for next generation personal communication networks, IEEE Network 14 (5) (2000) 18–24.
[4] P. Gondim, Genetic algorithms and the location area partitioning problem in cellular networks, in: Proceedings of IEEE 46th Vehicular Technology Conference, 'Mobile Technology for the Human Race', vol. 3, 1996, pp. 1835–1838.
[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 23 (13) (2000) 1263–1280.
[6] J. Taheri, A. Zomaya, A genetic algorithm for finding optimal location area configurations for mobility management, in: LCN'05: Proceedings of The IEEE Conference on Local Computer Networks 30th Anniversary, IEEE Computer Society, 2005, pp. 568–577.
[7] J. Taheri, A. Zomaya, A simulated annealing approach for mobile location management, in: IPDPS'05: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, IEEE Computer Society, 2005, pp. 194–201.
[8] J. Taheri, A. Zomaya, Clustering techniques for dynamic mobility management, in: MobiWac'06: Proceedings of the 4th ACM international workshop on Mobility Management and Wireless Access, ACM, 2006, pp. 10–17.
[9] R. Subrata, A. Zomaya, A comparison of three artificial life techniques for reporting cell planning in mobile computing, in: IEEE Transactions on Parallel and Distributed Systems, vol. 14, no. 2, IEEE Press, 2003, pp. 142–153.
[10] J. Taheri, A. Zomaya, A modified Hopfield network for mobility management, in: Wireless Communications and Mobile Computing, vol. 8, no. 3, John Wiley and Sons Ltd., 2008, pp. 355–367.
[11] R. Subrata, A. Zomaya, Evolving Cellular Automata for Location Management in Mobile Computing Networks, IEEE Transactions on Parallel and Distributed Systems, vol. 14, no. 1, IEEE Press, 2003, pp. 13–26.
[12] A. Bar-Noy, I. Kessler, Tracking mobile users in wireless communications networks, IEEE Transactions on Information Theory 39 (1993) 1877–1886.
[13] E. Alba, J. Garca-Nieto, J. Taheri, A.Y. Zomaya, New research in nature inspired algorithms for mobility management in GSM networks, in: EvoWorkshops, Springer, LNCS 4974, March, 2008, pp. 1–10.
[14] K. Price, R. Storn, Web Site of Differential Evolution (online), International Computer Science Institute, Berkeley. Available from: http://www.icsi.berkeley.edu/storn/code.html (accessed February 2009).
[15] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization (Kluwer Academic Publishers) 11 (1997) 341–359.
[16] D. Lam, J. Jannink, D.C. Cox, J. Widom, Modeling location management for personal communication services, in: Proceedings of 1996 IEEE International Conference on Universal Personal Communications (ICUPC96), vol. 2, 1996, pp. 596–601.
[17] R. Subrata, A. Zomaya, Dynamic location area scheme for location management, Telecommunication Systems 22 (1–4) (2003) 169–187.
[18] S. Almeida-Luz, M.A. Vega-Rodríguez, J.A. Gómez-Pulido, J.M. Sánchez-Pérez, Applying differential evolution to a realistic location area problem using SUMATRA, in: Proceedings of The Second International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP'08), IEEE Computer Society, 2008, pp. 170–175.
[19] S. 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: J.J.M. Neves (Ed.), New Trends in Artificial Intelligence APPIA, Associacao Portuguesa para a Inteligencia Artificial, 2007, pp. 219–230.
[20] Test Networks Benchmark: http://oplink.lcc.uma.es/problems/mmp.html (accessed February 2009).
[21] S. 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, 2007, pp. 1–5.
[22] J. Taheri, A. Zomaya, The use of a Hopfield neural network in solving the mobility management problem, in: ICPS'04: Proceedings of The IEEE/ACS International Conference on Pervasive Services, IEEE Computer Society, 2004, pp. 141–150.
[23] J. Taheri, A. Zomaya, A combined genetic-neural algorithm for mobility management, Journal of Mathematical Modelling and Algorithms (Springer Netherlands) 6 (3) (2007) 481–507.
[24] J. Taheri, A. Zomaya, Bio-inspired algorithms for mobility management, in: Proceedings of ISPAN'08 – The International Symposium on Parallel Architectures, Algorithms, and Networks, IEEE Computer Society, 2008, pp. 216–223.

Sónia M. Almeida-Luz is assistant 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, nature inspired optimization methods and evolutionary computing.

Miguel A. Vega-Rodríguez is Professor of Computer Architecture and, at present, Head of 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 280 publications including journal papers, book chapters and peer-reviewed conference proceedings. Furthermore, he is editor and reviewer of several international ISI journals. Dr. Vega-Rodríguez's main research interests are parallel and distributed computing, reconfigurable computing, and also evolutionary computing.

Juan A. Gómez-Pulido is 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. He has authored or co-authored 25 ISI journals, and many book chapters and peer-reviewed conference proceedings. His current research interest is the application of the high-performance reconfigurable computing to speedup evolutionary algorithms in large optimization problems.

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.