UNIVERSIDAD DE EXTREMADURA

TESIS DOCTORAL

Aplicación de Algoritmos Evolutivos al Problema de Gestión de la Localización en Redes Móviles

Application of Evolutionary Algorithms to the Location Management Problem in Mobile Networks

Autora:
Sónia Maria Almeida da Luz

Departamento: Tecnología de los Computadores y de las Comunicaciones

Conformidad del Director:

Fdo.: Dr. Miguel Ángel Vega Rodríguez

Cáceres, 2014

Acknowledgments

First of all, I would like to acknowledge all those who have helped, oriented and supported me, during all the phases of my PhD thesis.

Mainly, I would like to thank to my thesis director Dr. Miguel Ángel Vega Rodríguez, for all his educational support, ideas, suggestions, encouragement and patience from the first day of my PhD investigation until now, and his grateful guidance on the accomplishment of my thesis goals.

I would like to thank to my parents and other member of my family their support from the beginning of my studies and for their precious encouragement to always follow in front. Unfortunately I lost my father in the end of 2013, so I also want to dedicate him this hard work, because one of its concerns was about me being able to complete this thesis and accomplish my goals.

I would like to make a very special thanks to my husband David, and also for my baby Pedro, for their encouragement, patience and understanding.

I would like to thank to my school colleagues and closest friends for their support principally in moments of bigger stress and for their companionship. Their encouragement and motivation were of great important to help me to accomplish my thesis objectives.

I also would like to thank to all those that in same way contributed to the evolution of this investigation.

Finally I also want to thank several entities that provided partial support, in an economical or material way, to the development of this investigation and related activities. First of all, the Polytechnic Institute of Leiria, for their technical support and also same partial financial support through their Innovation, Development and Science Association – AIDC – IPL. Second, the Informatics Engineering Department of the School of Technology and Management of Leiria - DEI (http://www.dei.estg.ipleiria.pt/) ESTG. Following the Portuguese Science and Technology Foundation (FCT) and the Spanish Ministry of Science and Innovation and FEDER under the contract TIN2008-06491-C04-04 (the M* Project).

Abstract

The increase of mobile networks' users, nowadays, is an important fact that must be considered, because it also involves the growth of network dependent services and applications. Considering this, mobile communication networks must maintain a good response, without losing quality or availability, supporting the increase of users and their respective applications. With the objective that mobile networks keep this quality and availability, it is necessary to consider the Location Management (LM) when the network infrastructures are planned.

The Location Management problem corresponds to the characterization of the network configuration with the objective of minimizing the cost involved, mainly those associated to the user movements and respective tracing. The location management is partitioned in two main operations: location update that corresponds to the notification of current location, performed by mobile terminals when they change their location in the mobile network, and location inquiry (paging) that represents the operation of determining the location of the mobile user terminal, performed by the network when it tries to direct an incoming call to the user.

Considering the LM problem there exist several strategies divided between static and dynamic schemes, but our focus were the Location Area (LA) and the Reporting Cell (RC) problems, two of the most common static ones in actual mobile networks, because both consider, for all the users, the same network behavior.

The LA scheme is characterized by partitioning the network into groups of cells, where each group of cells represents a LA (also designated by region). 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.

The RC strategy is characterized by selecting and designating a subset of cells as reporting cells and setting the others as non-reporting cells (nRC). This scheme is not dependent of the individual movement characteristics of each user, but only of the arrangement of these reporting cells in the network. 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.

The core work of this thesis was the investigation and application of Evolutionary Algorithms (EA) to both of the problems, including the analysis and comparison of results achieved using test networks, generated with pattern simulation, and also realistic networks, based on real data extraction.

Initially, we had the goal of solving the Location Areas problem through several evolutionary algorithms, more specifically Differential Evolution (DE), Scatter Search (SS) and Greedy Randomized Adaptive Search Procedure (GRASP), analyse and compare the results accomplished, trying to understand the most adequate one. For each of these algorithms we studied and developed a respective based approach, which for the best of our knowledge it was the first time that these algorithms were applied to solve the LA problem.

To assure the statistical relevance of the results and conclusions exposed, we decided to execute 30 independent runs, for each configuration tested, over each of the approaches developed, and executions planned. Furthermore, we have also performed, for the majority of the experiments a statistical analysis using the ANOVA test, considering a confidence level of 95% (i.e. significance level of 5%, which means a p-value under 0.05).

After this work, and considering the results achieved over the LA problem, we decided to study and develop two main approaches, respectively based on Differential Evolution (DE) and Scatter Search (SS) algorithms with the goal of solving the RC strategy. Our main objective was to understand the best configuration of DE and SS when applied to the RCs strategy.

Taking in consideration the experiments performed, the results achieved and the respective comparison with those accomplished by using other artificial life techniques published in the literature, we noticed that our results are very competitive, mainly those obtained by our SS based approaches, when applied to the LA and the RC problems.

Finally, we decided to use the same test networks and compare the results accomplished by the Location Area based approaches with those obtained by the Reporting Cells based approach taking conclusion over what is the most effective in the lowering of the involved location management costs. Considering this results comparison

we noticed that the LA based approaches are mainly the ones that achieved the best solutions, those which represent the network configurations with lowest location management costs. Relatively to the algorithms used, like, in the previous results analysis we concluded that SS always equals or surpasses the results obtained by DE.

Resumen

El aumento de usuarios de redes móviles, hoy en día, es un hecho importante que debe ser considerado, ya que también implica el crecimiento de los servicios y aplicaciones que dependen de la red. Teniendo en cuenta esto, las redes de comunicaciones móviles deben mantener una buena respuesta, sin perder la calidad o disponibilidad, apoyando el aumento de usuarios y sus respectivas aplicaciones. Con el objetivo de que las redes móviles mantengan esta calidad y disponibilidad, es necesario considerar la gestión de localización (LM) cuando se planifican las infraestructuras de red.

El problema gestión de localización corresponde a la caracterización de la configuración de la red con el objetivo de minimizar los costes involucrados, principalmente los asociados a los movimientos de los usuarios y respectivo seguimiento. La gestión de localización está dividida en dos operaciones principales: actualización de localización que corresponde a la notificación de la localización actual, realizada por los terminales móviles cuando cambian su localización en la red móvil, y búsqueda de localización (paginación) que representa la operación de determinar la localización del terminal de usuario móvil, realizada por la red cuando quiere dirigir una llamada entrante para el usuario.

Teniendo en cuenta el problema de LM existen diversas estrategias divididas entre los esquemas estáticos y dinámicos, pero nuestro objetivo era los problemas *Location Areas* (LA) y *Reporting Cells* (RC), dos de los estáticos más comunes en las redes móviles actuales, debido a que ambos consideran, para todos los usuarios, el mismo comportamiento de la red.

El esquema de LA se caracteriza por la partición de la red en grupos de celdas, donde cada grupo de celdas representa una LA (designada también por región). En este esquema, cuando un terminal móvil se mueve a una nueva LA, su localización es actualizada, lo que significa una actualización de localización. Cuando el usuario recibe una llamada, la red debe hacer la paginación por todas las celdas de la nueva LA del usuario, en busca de su terminal móvil.

La estrategia de RC se caracteriza por seleccionar y designar un subconjunto de celdas como *reporting cells* (RC) y el establecimiento de las otras como *non-reporting*

cells (nRC). Este esquema no depende de las características del movimiento individual de cada usuario, sino sólo de la disposición de estas *reporting cells* en la red. Los terminales móviles sólo realizan una nueva actualización de localización cuando cambian su localización y se mueven a una *reporting cell*. Si una llamada recibida necesita ser encaminada a un usuario móvil, la búsqueda puede restringirse a su última *reporting cell* conocida y a sus vecinas que son *non-reporting cells*.

El trabajo principal de esta tesis ha sido la investigación y aplicación de Algoritmos Evolutivos (EA) para los dos problemas, incluyendo el análisis y la comparación de los resultados obtenidos mediante las redes de prueba, generados con un modelo de simulación, así como de los resultados obtenidos mediante redes realistas, basadas en la extracción de datos reales.

Inicialmente, teníamos el objetivo de resolver el problema de *Location Areas* a través de varios algoritmos evolutivos, más específicamente Evolución Diferencial (DE), Búsqueda Dispersa (SS) y *Greedy Randomized Adaptative Search Procedure* (GRASP), analizar y comparar los resultados alcanzados, tratando de comprender lo más adecuado. Para cada uno de estos algoritmos se estudió y desarrolló su respectiva propuesta, que según nuestro mejor conocimiento era la primera vez que se aplicaron estos algoritmos para resolver el problema de LA.

Para asegurar la relevancia estadística de los resultados y conclusiones expuestas, se decidió realizar 30 ejecuciones independientes, para cada configuración probada, sobre cada una de las propuestas desarrolladas, y las ejecuciones planificadas. Por otra parte, también hemos realizado, para la mayoría de los experimentos un análisis estadístico mediante el test de ANOVA, considerando un nivel de confianza del 95% (es decir, el nivel de significación del 5%, lo que significa un valor de p inferior a 0,05).

Después de este trabajo, y teniendo en cuenta los resultados obtenidos sobre el problema de LA, decidimos estudiar y desarrollar dos propuestas principales, respectivamente basadas en los algoritmos Evolución Diferencial (DE) y Búsqueda Dispersa (SS), con el objetivo de resolver la estrategia de RC. Nuestro principal objetivo era comprender la mejor configuración de DE y SS cuando se aplican a la estrategia de RC.

Tomando en consideración los experimentos realizados, los resultados obtenidos y la comparación respectiva con los alcanzados mediante el uso de otras técnicas de vida

artificial publicadas en la literatura, nos dimos cuenta de que nuestros resultados son muy competitivos, sobre todo los obtenidos por nuestras propuestas basadas en SS, cuando se aplica a los problemas de LA y RC.

Por último, se decidió utilizar las mismas redes de prueba y comparar los resultados alcanzados por las propuestas basadas en la estrategia *Location Areas* con los obtenidos por las propuestas basadas en la estrategia *Reporting Cells*, sacando conclusiones sobre lo que es más eficaz en la reducción de los costes de gestión de localización involucrados. Teniendo en cuenta estos resultados de comparación nos dimos cuenta de que las propuestas basadas en LA son principalmente las que obtuvieron las mejores soluciones, las que representan las configuraciones de red con los costes de gestión de localización más bajos. Relativamente a los algoritmos utilizados, al igual que en el análisis de los resultados anteriores se concluye que el algoritmo SS siempre iguala o supera los resultados obtenidos por el algoritmo DE.

Contents

List of Abbreviations

Abbreviation	Description
ACO	Ant Colony Optimization
BDT	Ball Dropping Technique
CPU	Central Processing Unit
DE	Differential Evolution
EA	Evolutionary Algorithms
EC	Evolutionary Computation
EELA	E-science grid facility for Europe and Latin America
EGEE	Enabling Grids for E-sciencE
EIA/TIA	Electronic and Telephone Industry Associations
EP	Evolutionary Programming
GA	Genetic Algorithms
GLS	Guided Local Search
GPSO	Geometric Particle Swarm Optimization
GRASP	Greedy Randomized Adaptive Search Procedure
GSM	Global System for Mobile Communications
HNN	Hopfield Neural Network
ILS	Iterated Local Search
IS	Interim Standard
LA	Location Area
LM	Location Management
LP	Location Paging
LS	Local Search
LU	Location Update
MAP	Mobile Application Part
MLM	Mobile Location Management
nRC	Non-Reporting Cell
PCN	Personal Communication Networks
PSO	Particle Swarm Optimization
QoS	Quality of Service
RC	Reporting Cell
RCL	Restricted Candidate List
SA	Simulated Annealing
SS	Scatter Search
TS	Tabu Search
VNS	Variable Neighborhood Search

List of Figures

List of Tables

List of Algorithms

1. Introduction

1.1. General Overview

In the last decade, the development of network infrastructures has been growing, principally those directed for personal communication networks (PCN) [1, 2, 3], because they must support the increase of user services and communications that enable all the users to make or receive calls at any time of the day and for, or from, any location. In order to these networks support the mobility of users and be able to find them, also when they change their location, it is essential to consider mobility management when the network infrastructures are defined.

Mobility management involves the process of location management (LM) 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 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 and considered a critical task for the current (and future) generations of mobile networks.

Location management involves two elementary operations: location update and location inquiry (or terminal paging) [4]. 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 (paging) 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.

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 (LA) and Reporting Cell (RC) schemes, which are two of the more common ones [5].

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. 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 [6, 7, 8].

The RC strategy is characterized by defining a subset of cells as reporting cells (RC) and the others as non-reporting cells (nRC). 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 neighbors which are non-reporting cells [9].

The location management is an NP-hard optimization problem [10], therefore, the use of Evolutionary Algorithms (EA) is a very interesting approach. In fact, we will apply the Differential Evolution (DE) [11, 12, 13] and Scatter Search (SS) [14, 15, 16] algorithms to find the best configuration for the location areas and reporting cells strategies, which principally consider the location update and paging costs. We also will analyse the results of sequential and parallel implementations of Greedy Randomized Adaptive Search Procedure (GRASP) [17, 18, 19] based approach over the location areas strategy.

The main goal of this PhD Thesis is to implement an approach where we want to define the best configuration of each of these evolutionary algorithms, using test networks [7, 10, 20, 21], and also realistic networks [22, 23, 24], as well as compare our results with the ones obtained by other authors and surpass those other results.

1.2. Main Goals and Motivation

As said, the main goals of this investigation are the definition of new solutions to solve the location management problem, as a cost optimization problem of current mobile networks. These new solutions correspond to the investigation and development of two new approaches based respectively on the location areas and reporting cells schemes. The Location Areas and Reporting Cells schemes are two location management strategies that help making the network partitioning with the objective of minimizing the involved costs.

The main work of this thesis is the investigation and application of evolutionary algorithms to both of the problems, including the analysis and comparison of results obtained using test networks, generated with pattern simulation, and also realistic networks, based on real data extraction.

There exist several works using different types of metaheuristics applied to the location areas and reporting cells schemes, but for the best of our knowledge, it is the first time that the evolutionary computation algorithms that we have studied and implemented are applied and that is one of our main contributions of this thesis. We had the objective of obtain better results than the rest of authors by using other evolutionary algorithms and that goal was accomplished, which is one of our main contributions with this investigation.

Furthermore, this Thesis presents several points of interest and benefits according to different perspectives:

1. Considering the economic and industrial point of view, this Thesis involves the study of a real problem applied to telecommunications and mobile networks. These networks have an increasing importance. Furthermore, the optimization of the location management means the minimization of the involved costs, which is very important because if we can lower those costs, also the telecommunication costs for the final users can be lowered.
2. Regarding the scientific point of view, this Thesis represents the deepening of knowledge of evolutionary algorithms and great experience in its respective implementation applied to optimization problems. In this project these optimization problems are specifically over

telecommunication and mobile network problems, but that can be extrapolated to distinct optimization problems.

3. Considering the country point of view, this problem is part of a national research project with the objective of being applied into the Spanish telecommunication and mobile networks. If we can enhance the current solutions, this could be a major benefit to the telecommunication systems.

The main work points of this Thesis correspond to the investigation, adaptation, implementation and analysis of the evolutionary algorithms applied to the location areas and reporting cells strategies.

To accomplish the objectives of this Ph.D. Thesis, several tasks, which are listed as follows, were performed:

- The first task of this research was the study of bibliography and start gathering information concerning the state of the art over the Location Management problem.
- After understanding the general characteristics of the LM problem we had focused on the Location Areas strategy trying to perceive their main parts and how it could be implemented. With the study of the LA problem we also wanted to know what type of algorithms have already been implemented by other authors.
- The study and implementation of the Differential Evolution based approach applied to the LA problem. The application of the approach was performed over test and realistic networks with the objective of performing a statistical evaluation of the experimental results and making comparison with the results achieved by other authors' approaches. Study the DE parameters intensively in order to determine its most adequate values was other task performed.
- Study of the Reporting Cells strategy of location management with the major goal of understanding its main characteristics, how it tackles the LM costs and analysing the principal differences when compared with the LA strategy.
- Implementation of the differential evolution based approach applied to the reporting cells problem and the respective application to test networks. Perform a statistical evaluation of the experimental results and comparison

with the results of other authors. Tuning of the DE parameters considering the results obtained.

- Study and implementation of the Scatter Search based approach applied to the location area problem and the respective application of the approach to test and realistic networks. Identification and tuning of the SS parameters during the experiments considering the results achieved in each phase. Analysis and statistical evaluation of the experimental results, including comparison with other authors and also with the results achieved with the differential evolution based approach.
- Implementation of the scatter search based approach applied to the reporting cells problem and respective application to test networks. Perform analysis and statistical evaluation of the experimental results followed by respective comparison with those obtained by other author's approaches and with our differential evolution approach.
- Study of the Greedy Randomized Adaptive Search Procedure algorithm with the goal of identifying its main characteristics and understand how it could be applied to the location areas strategy, using sequential and parallel implementations. Present analysis and evaluation of the experimental results achieved using a grid computing environment.
- Perform a direct comparison between LA and RC strategies in order to determine the most adequate one to accomplish the lowest LM costs, which are the update costs and the paging costs.

The final conclusions had the objective of comparing the results obtained with all the different evolutionary algorithms implemented and also with those achieved by other authors.

1.3. Document Organization

The dissertation is organized as follows. In chapter 2 the Location Management problem is explained concerning its contextualization in mobility management of communication networks. After exposing the characteristics of LM problem we present the respective classical strategies, always-update and never update, which are considered the two extreme strategies of LM. We follow focusing on Location Area and Reporting Cell strategies, presenting their features and respective way of facing the LM costs. These are two of the most common strategies included in the category of static schemes of LM.

Chapter 3 describes the evolutionary algorithms used, based respectively on Differential Evolution, Scatter Search and Greedy Randomized Adaptive Search Procedure. We explain the structure of each of these algorithms and expose their features including the outline to address each one to the respective strategies of LM addressed in our investigation.

In chapter 4 we present some literature review considering the research performed over the related work developed by other authors when trying to solve the LA and RC strategies. This review includes the exposition of different techniques explored and implemented by others with the goal of solving the LM problem.

Chapter 5 and 6 represent the development and implementation of our investigation, respectively, over the LA based approach and the RC based approach. In each of these chapters we expose the implementation considerations over the respective approaches and include the experiments and analysis over the results achieved. Each chapter also includes two sections where we make, first, the comparison between our results obtained by each approach and, second, the comparison of our results with the results obtained by other authors. Finally, in chapter 6, we have one last section dedicated to the direct comparison between LA and RC strategies with the objective of understanding the most adequate one to accomplish the lowest LM costs, respectively update costs and paging costs.

Finally, chapter 7 includes the major conclusions over this work and future lines of work.

2. Location Management Problem

In this chapter we expose the main characteristics of Location Management (LM) which is one of the two processes of mobility management of mobile networks. The LM is responsible for enabling the network to find the current location of each mobile terminal in order to forward calls to the respective cell within the network configuration. The process of LM has the purpose of reducing the overhead costs required in locating mobile terminals in a mobile network.

In our work, we centre the investigation over the Location Area and the Reporting Cells strategies of location management, two of the more common static strategies applied to the mobile networks of GSM systems. We take in consideration that the current static LM standards are the Electronic and Telephone Industry Associations (EIA/TIA) Interim Standard IS-41/TIA-41 [25] and the Global System for Mobile Communications (GSM) Mobile Application Part (MAP) GSM MAP [26]. Both of these two standards are very similar because they consider as main tasks the location update and the call delivery, or paging [27, 28].

We will start this chapter making a general outline of the location management problem. After that we will explain the classical strategies always-update and never-update, which are considered the two extremes of static LM strategies.

Following, we describe the main characteristics of LA and RC strategies, considering that they are the main focus of our investigation over LM problem, with the objective of minimizing the associated costs.

2.1. General Overview

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 between many other mobile services. Because of this, communication networks must support a big number of users and their applications maintaining a good response without losing quality and availability. With the intention that mobile networks keep this quality it is necessary

to consider the mobility management when making design of the network infrastructure and also during the network operation.

The mobility management is a very important point because includes the process of handoff management that enables the mobile network to locate roaming mobile terminals, and also the process of location management that enables the mobile network to find the current location of a mobile terminal in order to make or receive calls.

In this investigation our main concern over mobility management is the process of location management, which involves the user movements and tracing, and their respective requests [29, 30]. These requests usually occur when a mobile terminal changes its location, or even when the quality of the received signal becomes deteriorated. Due to that, this process becomes even more important for the current and future generations of mobile networks, because it is responsible for enabling the network to find the most up to date location of each mobile terminal, allowing the users to receive or make calls, independently of their location and time of the day.

Location update and location paging are the two main and most important operations of LM over the mobile networks. The location update (LU) is used to report the current location, performed by mobile terminals when they change their location. The location paging (P) corresponds to the operation of determining the location of the mobile user terminal, which is performed by the network when it needs to forward an incoming call to the user.

With the objective of reducing the required location updates and location paging a variety of LM schemes have been proposed. These several strategies of location management are often divided into two main categories: static and dynamic schemes [5, 27, 31, 32, 33].

The static schemes consider that all users of one network have the same behaviour. They define the frequency and occurrence of location updates apart of the users' characteristics. These schemes are less complex, require less computational effort, considering the lack of independent user tracking and parameterization, and because of that are more common in actual mobile networks.

There exist several static techniques based on static location update and static location paging (this is, mobile location management in static mode) as distance-based strategies, movement-based strategies, time-based strategies, and as more common ones

we have always-update, never-update, location area and reporting cells schemes [5, 28, 34].

Always-update and never-update correspond to the two simple and classical location management strategies and normally correspond to the extremes of this type of strategies and will be explained in more detail in section 2.2. *Classical Strategies*. Because of that most of existing network systems use a combination of them and then compare the results with these classical ones.

The Location Areas scheme uses a combination of these two classical strategies and will be explained with more detail in section 2.3. *Location Areas Problem*.

The Reporting Cells scheme also uses a combination of these two classical strategies, but also considers a vicinity factor for each cell of the network, which will be exposed with more detail in section 2.4. *Reporting Cells Problem*.

The dynamic schemes consider different network topologies for different users, based on the individual calling and movement patterns of each user.

Dynamic schemes are more complex, require more computational effort and because of that are not as common as the static ones. The dynamic techniques are manly based on users' behaviour adjusting the location update frequency per user. Dynamic strategies are divided into two parts, the first one where the user's behavioural patterns are modelled and the second one where the extracted model is used to locate the user. There exist several dynamic strategies in the literature for solving this problem such as timer-based [35, 36, 37, 38, 39], distance-based [40, 41], movement-based [5, 42, 43, 44], history-based[45], replication-based, pointer-based [46, 47] (among others).

2.2. Classical Strategies

There are two classical strategies of location management: always-update and never-update. They correspond to the two simple location management strategies and are considered as the extremes of this type of strategies.

In the always-update strategy, each mobile terminal performs a location update every time it enters in a new cell, but however no search operation would be required for incoming calls, because the network always has complete knowledge of the user's location when an incoming call arrives.

In this strategy it is considered that all cells have different location areas, which means that the number of location areas is equal to the number of cells. In Figure 1 it is possible to observe a network configuration considering the always-update strategy.

The always-update strategy presents a good performance for networks including users with low mobility or high call arrival rates. However, has a quite poor performance for users with high mobility and due to that requires additional location updates leading to excessive use of resources.

Figure 1 – Network with Always-Update configuration

In the never-update strategy, no location update is performed, because all cells are considered as belonging to the same location area (just one location area for all cells of the network), as is possible to see in Figure 2, but when there is an incoming call a search operation is executed, over all the cells, with the objective of finding the corresponding user.

This strategy involves no location update overhead, however, when it is executed over larger networks or networks with high call arrival rates, tends to result in excessive paging.

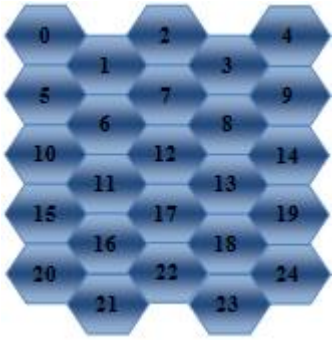


Figure 2 – Network with Never-Update configuration

These two strategies represent the extremes of location management strategies where always-update minimizes the paging costs and the never-update minimizes the location update cost.

2.3. Location Areas Problem

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

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

The LA scheme is characterized by partitioning the network into groups of cells. Each group of cells represents a LA (also designated by region). In Figure 3, it is possible to observe the configuration of a network of 25 cells distributed by four LAs.

Figure 3 – Network configuration partitioned in LAs

In this scheme, when a mobile terminal moves to a new LA (by example when a user moves from one cell of blue LA to the cell, number 13, marked with the X in the green 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.

The LA problem can be defined as the problem of finding an optimal network configuration of location areas, minimizing the location management cost. The location management cost, of the LA problem, is normally divided in two main parts: the location update cost, which is based on the moves of a mobile terminal from one LA to a new one; and the location paging cost, which must be considered due to the paging actions that the network must perform when it is necessary to route incoming calls to the users [5, 6].

2.3.1 Location Update Cost

The location update (LU) cost illustrates the cost related with the location updates performed by mobile terminals in the network, when they change their location to a cell that belongs to a different LA. Considering this, 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, analyzing if they move from different LAs.

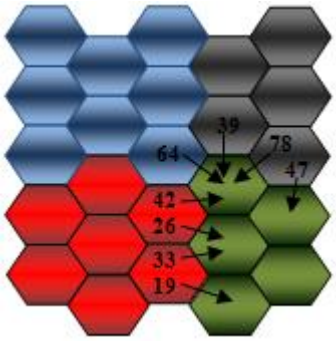


Figure 4 – Network configuration with entering flow of users to the green LA.

If we consider the network of Figure 4, it is possible to observe the total number of users who enter in the green LA (from the neighborhood cells, which belong to the other LAs). To calculate the location update cost for the green LA, it is necessary to sum up those numbers of users that enter (from cells that belong to another LA) on each cell of this LA, as in (1).

$$\boldsymbol{N_{LU} = 47 + 78 + 39 + 64 + 42 + 26 + 33 + 19 = 348} \quad \textbf{(1)}$$

This is just the calculus, for one LA, but it must be calculated for all LAs that could exist in the network.

2.3.2 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 in general, the number of paging transactions is directly related to the number of incoming calls that must be routed.

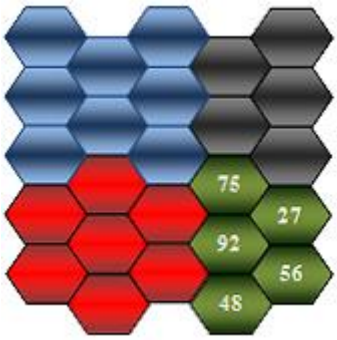


Figure 5 – Network configuration with incoming calls to the green LA.

The calculus of the paging cost is simpler, because we only need to sum the number of incoming calls in the selected LA and then multiply the value by the number of cells that compound the respective LA. Considering the incoming calls to the green LA shown in Figure 5, the calculus of paging cost is:

$$\boldsymbol{N_P = (75 + 92 + 27 + 48 + 56) \times 5 = 1490} \quad \textbf{(2)}$$

Also for the calculus of paging cost would be necessary to apply the formula to all the LAs that compound the network.

2.3.3 Location Management of Location Areas Total Cost

Besides the location update and paging costs, the location management cost involves other parameters and components, like the cost of database management and maintenance of infrastructures, between others. However, those costs are considered to be equal for all strategies [7] and do not influence the comparison of different strategies. Due to that, we will not consider them for the total cost. Therefore, the combination of location update cost and location paging cost is sufficient to compare the results obtained by different strategies.

The generic formula to calculate the total cost of location management [48] is given as:

$$\boldsymbol{LMCost = \beta \times N_{LU} + N_P} \quad \textbf{(3)}$$

The total cost of location updates is given by N_{LU}, the total cost of paging transactions is specified by N_P, and β corresponds to a ratio constant used in a location update relatively to a paging transaction in the network. This ratio constant is needed, because the cost of each location update is considered to be much higher than the cost of each paging transaction, because of 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 or receiving any call [7]. Due to all of that, the cost of a location update is normally considered to be 10 times greater than the cost of paging transactions, that is, β =10 [6].

Considering the network configuration presented in Figure 4 and Figure 5, the total location management cost for the green LA, by (3), would be:

$$\boldsymbol{LMCost = 10 \times 348 + 1490 = 4970} \quad \textbf{(4)}$$

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. For this network configuration (see Figure 4 and Figure 5) and considering the following LM costs of each LA: blue LA= 5230; grey LA = 3478; red LA = 6723; green LA = 4970; the total cost calculus is shown in (5):

$$\boldsymbol{Total\ LMCost = 5230 + 3478 + 6723 + 4970 = 20401} \quad \textbf{(5)}$$

2.4. Reporting Cells Problem

The Reporting Cells (RC) strategy, proposed by Bar-Noy and Kessler in [9], has the objective of minimizing the location management costs involved with the operation of tracking mobile users.

This strategy is characterized by selecting and designating a subset of cells as reporting cells and setting the others as non-reporting cells (nRC). This scheme is not dependent of the individual movement characteristics of each user, but only of the arrangement of these reporting cells in the network. When an incoming call arrives for the mobile terminal, it is necessary to conduct a search around the vicinity of the last reporting cell from which the user reported and updated their location [5, 49].

If we consider Figure 6, we can observe the configuration planning of a 4x4 network and observing Figure 7 we can see that RCs are represented with value 1 and nRCs are represented with value 0. The location update of mobile users is performed only when they change their location and its mobile terminal enters in a reporting cell. If an incoming call must be routed to the mobile user, the search is restricted to all the cells that compound the vicinity of the last known RC, which means that it is not necessary a wide search over the entire network.

Figure 6 – Reporting Cells Network Planning

2.4.1 Vicinity Values

We need to take in consideration that, for each cell in the network, it is necessary to calculate the vicinity value, which corresponds to the maximum number of cells that the network might page when an incoming call occurs and need to be routed.

Figure 7 – Network configuration between RCs (1) and nRCs (0)

The vicinity value of a RC indicates the number of nRCs that are reachable from this RC, without passing over other reporting cells, but considering the RC itself. Considering, as an example, the calculus of vicinity value for the cell number 9 in Figure 6, we need to count all the neighbor cells that are nRC, respectively cells 8, 13, 14 and 15; and also the RC itself. With this calculus we obtain the vicinity value of 5 for the RC number 9.

The calculus of vicinity value, for a non-reporting cell, must consider the maximum vicinity value among the RCs from where this nRC can be achieved. This is, if the nRC belongs to the neighborhood of several RCs, the calculus must be performed for each of them and then select the highest vicinity value. For example, considering the cell number 2 in Figure 6, we observe that it belongs to the neighborhood of the RCs number 4, 5, 6 and 7, which have respectively the vicinity values of 6, 6, 5 and 5 (see

Figure 8). In this process we must select the highest vicinity value from those RCs, so we must set the vicinity value of nRC number 2 as 6.

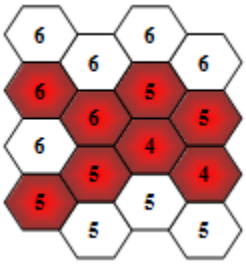


Figure 8 – Vicinity values of each cell

Using the RC planning of Figure 6 and calculating the vicinity values, for all the cells, we obtain the final result shown in Figure 8.

2.4.2 Location Management of Reporting Cells Total Cost

Also for the RC strategy, the location management cost is principally divided into two main operations: location update and location paging. The location update (LU) is used to report the current location, executed by mobile terminals when they change their location. The location paging (P) corresponds to the operation of determining the location of the mobile terminal, which is performed by the network when it needs to forward an incoming call to the user.

Considering the explanation made in section *2.3 Location Areas Problem*, also for the reporting cells scheme, we only considered these two main costs, because although location management cost involves other parameters and components, they are considered to be equal for all strategies [7].

As we mentioned earlier, in the RC scheme the location updates only occur when a mobile user enters in a reporting cell, so the vicinity value of each cell must be considered. Due to this, the generic formula given by (3) must be readjusted and re-formulated in (6) as [17]:

$$\boldsymbol{LMCost = \beta \times \sum_{i \in S} N_{LU}(i) + \sum_{i=0}^{N-1} N_P(i) \times V(i)} \qquad \textbf{(6)}$$

Where we have the $N_{LU}(i)$ that is the total number of location updates for RC i, S indicates the subset of cells defined as RCs, $N_P(i)$ corresponds to the number of incoming

calls attributed for cell *i*, *N* represents the total number of cells, which compound the mobile network configuration, and *V(i)* corresponds to the vicinity value calculated for cell *i*.

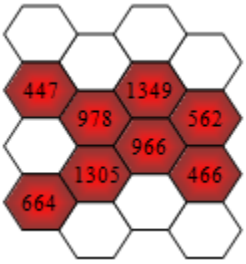


Figure 9 – Network configuration including Location Updates of Reporting Cells

In this approach, we have the intention of determining the best network configurations, defining what cells will be set as RCs, with the main goal of minimizing the LM costs. Following we explain the calculus of LM costs, using the RC configuration, and respective vicinity values, of network shown in Figure 8, the location updates presented in Figure 9 and the incoming calls presented in Figure 10.

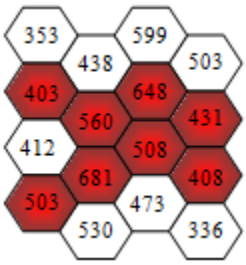


Figure 10 – Network configuration including Incoming calls of each cell

Considering this formula we know that the location update values (see Figure 9) are only considered for the RCs and the respective calculus of update cost would be:

$$\boldsymbol{N_{LU} = 447 + 978 + 1349 + 562 + 1305 + 966 + 466 + 664 = 6737} \quad \textbf{(7)}$$

Relatively to calculus of paging cost, we must consider the incoming calls of each cell, shown in Figure 10, the respective vicinity values, shown in Figure 8, and it would be as presented in (8).

$$\begin{aligned}\boldsymbol{N_P} &= \boldsymbol{353 \times 6 + 438 \times 6 + 599 \times 6 + 503 \times 6 + 403 \times 6 +}\\ &\boldsymbol{560 \times 6 + 648 \times 5 + 431 \times 5 + 412 \times 6 + 681 \times 5 + 508 \times 4 +}\\ &\boldsymbol{408 \times 4 + 503 \times 5 + 530 \times 5 + 473 \times 5 + 336 \times 5 = 41\,282}\end{aligned} \quad \textbf{(8)}$$

Finally, after obtaining these two partial costs, we are able to calculate the total LM cost. Considering (6), we know that it is necessary to apply a ratio constant of value 10 to the location update cost, because it is considered much higher than the paging cost, so the final result would be as:

$$\boldsymbol{Total\ LMCost = 10 \times 6\,737 + 41\,282 = 108\,652} \quad \textbf{(9)}$$

3. Evolutionary Algorithms

Metaheuristics are generally used for combinatorial optimization, where an optimal solution is sought over a discrete search-space of candidate solutions [50]. In the last few years some researchers proposed several formal definitions of metaheuristics.

In [51] Osman and Laporte proposed: "A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions."

In [52] Voβ et. al, proposed: "A metaheuristic is an iterative master process that guides and modifies the operations of subordinate heuristics to efficiently produce high-quality solutions. It may manipulate a complete (or incomplete) single solution or a collection of solutions at each iteration. The subordinate heuristics may be high (or low) level procedures, or a simple local search, or just a construction method."

There exist several types of metaheuristics, which are mainly classified and divided into two main groups, the population-based and the trajectory-based [50]. The population-based algorithms work over a population of solutions in any iteration, while the trajectory-based algorithms work over a single solution.

The most studied population-based methods in combinatorial optimization are Evolutionary Computation (EC) and Ant Colony Optimization (ACO, one of the most used swarm intelligence algorithms) [50]. Considering the EC algorithms [53], Evolutionary Programming (EP) [54, 55] and Genetic Algorithms (GA) [56, 57, 58] are the most common algorithms, although Differential Evolution (DE) [11, 12, 59, 82] and Scatter Search (SS) [15, 61, 62] are also very well-known evolutionary algorithms.

All the Evolutionary Computation algorithms [63] are included in the population-based heuristics where the search process describes the evolution of a population compound by a set of solutions in the search space.

In the last years Evolutionary Computation algorithms, as genetic algorithms, evolution strategies and genetic programming (among others), have received a lot of attention with the objective of solving a wide range of non-linear optimization problems.

Evolutionary Algorithms (EA) are search methods that follow the metaphor of natural biological evolution, which adapts the environment changes to find an optimal solution to a problem through evolving a population of candidate solutions, during a certain number of generations. EA use each solution to create a new one to the next generation based on the fitness values of each candidate and applying techniques of crossover, mutation and selection [56, 64]. Each type of EA has its own specificities and that make them different one of each other.

On the other hand, the more recently proposed trajectory-based methods are Greedy Randomized Adaptive Search Procedure (GRASP) [17, 18, 65], Tabu Search (TS) [66, 67, 68], Simulated Annealing (SA) [69, 70], Variable Neighborhood Search (VNS) [71, 72, 73], Guided Local Search (GLS) [74, 75, 76] and Iterated Local Search (ILS) [77, 78, 79, 80, 81].

In the next sections will be detailed the Differential Evolution (DE) and Scatter Search (SS), which are population-based evolutionary algorithms, because those correspond to the ones that have been implemented during the development of this work. We will also detail the Greedy Randomized Adaptive Search Procedure (GRASP), a trajectory-based algorithm, because in our work we also analyze the results of its sequential and parallel implementations over the LA problem.

3.1. Differential Evolution

The differential evolution (DE) is a population-based evolutionary algorithm, developed by Price and Storn [11, 12, 59, 60]. DE is one strategy based on evolutionary algorithms with some specific characteristics, whose main objective is functions optimization.

The main strategy of DE algorithm is to generate new individuals by calculating vector differences between other randomly-selected individuals of the population. This new generated individual, called candidate solution, is considered to be a better and succeed to the next generation, if it represents a better fitness value, which in our case corresponds to a solution with a lower location management cost.

This algorithm includes four important parameters: population size, mutation, crossover and selection operators; there are different differential evolution strategies, which follow specific schemes.

3.1.1 Initial Population

DE algorithm, as other evolutionary algorithms, works with a population of *NI* individuals (each one representing a candidate solution) and this number never changes during the optimization process.

Considering the LA problem, each individual represents a network configuration, where the cells are divided in one or more groups representing distinct location areas.

Relatively to the RC problem, each individual represents a network configuration of N cells divided between reporting cells and non-reporting cells.

Usually the initial population is generated using a random process and then, that population will be improved by the algorithm iteratively, through the mutation, crossover and selection operators (in [12, 13] it is possible to see more details about the DE flowchart).

3.1.2 Mutation Operator

The mutation 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 by using equation (10):

$$I_{i,g+1} = X_{1,g} + F\left(X_{2,g} - X_{3,g}\right) \qquad \textbf{(10)}$$

The value of mutation *F* must be greater than zero and will be used to 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 defining the quantity of perturbations in the generation of a new individual like the majority of other evolutionary algorithms do.

3.1.3 Crossover Operator

Crossover operator probability *Cr* corresponds to 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 equation (11):

$$T_{ji,g+1} = \begin{cases} I_{ji,g+1} & if\ rjn \leq Cr \\ X_{ji,g} & otherwise \end{cases} \quad \textbf{(11)}$$

where $j = 1, 2, \ldots G$. G corresponds to the number of genes of an individual and *rjn* corresponds to the random value generated. If the *rjn* generated is lower or equal to the *Cr* predefined, the index of the trial individual position is provided by the mutant individual ($I_{ji,g+1}$); otherwise it will be inherited from the target individual ($X_{ji,g}$).

3.1.4 Selection Operator

Selection operator has the purpose of comparing the trial individual (offspring) produced by the crossover operator with the target individual (parent) and then it determines the one that will be part of next generation. If a trial individual has a smaller cost function value (which in our case corresponds to a lower location management cost)

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 (12):

$$\begin{aligned} &If\ f(T_{i,g+1}) \leq f(X_{i,g}), \quad set\, X_{i,g+1} = Ti, g+1 \\ &Otherwise \qquad\qquad\qquad\qquad X_{i,g+1} = X_{i,g} \end{aligned} \tag{12}$$

3.1.5 DE Schemes

Price and Storn [11, 12] have suggested 10 mainly 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 on the type of crossover used.

Table 1 – Schemes of Differential Evolution Algorithm

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)*

3.1.6 DE Algorithm

The outline of the DE algorithm, considering the *DE/best/1/exp* is presented in Algorithm 1. It starts by defining, validating and evaluating the initial population through calculating the fitness value for each individual.

Following, until the termination condition is not reached, the necessary individuals are selected and a new one is generated 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 elected and pass for population of the next generation.

Algorithm 1 – Outline of DE algorithm with scheme DE/best/1/exp

```
Differential Evolution Algorithm
Initialize the population
Validate the initial population
Evaluate the initial population
While (termination condition not satisfied) {
    For each individual in the population do {
        Randomly select individual xr1 ≠ xbest
        Randomly select individual xr2 ≠ xr1 and ≠ xbest
        Generate trial individual: xtrial = xbest + F(xr1 – xr2)
        Use Cr to define the amount of genes changed in trial individual
        Validate the trial individual
        Evaluate the trial individual
        Deterministic selection
    }
}
```

3.2. Scatter Search

Scatter search (SS) is an evolutionary algorithm introduced by Glover in 1977 as a heuristic for integer programming [14]. The objective of SS is to maintain and improve a set of quality and diverse candidate solutions through an iterative process of improving an initial population. The processes within SS are not restricted to a uniquely design and that allows the investigation of different strategic possibilities that may prove to be effective in a particular implementation. These observations and principles lead to the outline of the scatter search implementation (presented in Algorithm 2), based on five main methods [15, 16, 61, 62]: *Diversification Generation method*, *Improvement method*, *Reference Set Update method*, *Subset Generation method* and *Solution Combination method*.

The advanced and most specific features of SS are related to the implementation of these five methods that will be detailed in the following subsections.

Algorithm 2 – Outline of SS algorithm

```
Scatter Search Algorithm
Create Population with PSize different solutions.
Using Diversification Generation method and Improvement method.
Define a RefSet = {x^1,…,x^b} with b/2 best solutions and b/2 most diverse solutions of P.
Order the RefSet of solutions, applying their fitness function.
MakeNewSolution = TRUE.
while (Exist (NewSolution) and (termination condition not satisfied)) do
        MakeNewSolution = FALSE
        Use the Subset Generation method and create all different subsets
        while(Exist (subsets not examined)) do
                Select a subset and label it as examined
                Apply the Solution Combination method to the solutions of the subset
                Apply the Improvement method to each new solution x
                if (f(x)<f(x^b) and (x∉RefSet) then
                        Set x^b = x and order solutions of RefSet
                        MakeNewSolution = TRUE
                endif
        endwhile
        if (Not Exist (New Solution)) then
                Create b/2 most diverse solutions for RefSet and order RefSet
                MakeNewSolution = TRUE.
        endif
endwhile
```

3.2.1 Diversification Generation Method

The Diversification Generation method has the objective of generating an initial population of *P* diverse solutions. The size of population *P* (*PSize*) is usually, at least, 10 times the size of the reference set. The diversification generation method is normally adapted to specific problems and can be totally deterministic or partially random. The quality of solutions is not important, but the guaranty that all of them are different is a requirement [15, 83].

3.2.2 Improvement Method

The Improvement method is used with the goal of improving each solution generated through the diversification generation method and the solution combination method. Normally it is implemented by means of a local search, and with that should be possible to generate multiple instances of the initial solution, generating improved solutions.

This method is the one that may not be implemented in the SS methodology and it is used only if high quality solutions are desired [15, 83].

3.2.3 Reference Set Update Method

The Reference Set Update method is implemented with the objective of building and maintaining a reference set (*RefSet*) of *b* solutions, divided between high quality and most diverse solutions. These solutions will generate new trial solutions using the Solution Combination method. More specifically, the size of the reference set corresponds to the junction of $b1 + b2 = b$, where $b1$ is the size of the subset of high quality solutions and $b2$ represents the size of the subset of diverse solutions.

The initial *RefSet* is built with the selection of these best $b1$ solutions, which are removed from *P* and added to the *RefSet*. After that, the most diverse $b2$ solutions are chosen, normally, through a distance function relatively to the best solutions. After the initial *RefSet* being built, it will be updated along the SS process, while exist new solutions to analyze.

The *RefSet* can be statically or dynamically updated. The static update implementation means that the new solutions generated are kept in a pool and only will be added (by replacement of the worst solutions) in the *RefSet* after all being created. On contrary, in the dynamic update the *RefSet* is dynamically updated while the solution combination method is applied. In each time that a new trial solution is generated, it will be subject to the Improvement method, to guarantee a feasible solution, and then compared with the worst solution in the *RefSet*. If a newly created solution has a better

value, considering the objective function of the problem, it will replace the old one (with worst objective function value) in the *RefSet*. Each time that the Subset Generation method is applied, it will generate all the new subsets that will be considered in the Solution combination method [15, 62, 83].

3.2.4 Subset Generation Method

The Subset Generation method is employed to generate all the different subsets of solutions, which will be combined with the Solution Combination method.

Usually there are considered four main types of subsets to be generated. Type 1 corresponds to all 2-solution subsets. Type 2 represents 3-solution subsets derived from 2-solution subsets, by including in each 2-solution subset the best solution of the *RefSet*, which is not in this subset. The subset Type 3 corresponds to the 4-solution subsets derived from 3-solution subsets, where in each 3-solution subset is included the best solution not in this subset. Finally, type 4 consists in all the subsets of the best i solutions, where $i=5$ to b [15, 62, 83].

3.2.5 Solution Combination Method

The Solution Combination method is used to combine the solutions in each subset generated with the Subset Generation method, with the objective of creating one or more trial solutions. The Combination process depends specifically on the nature of the problem and the type of solutions to be combined. Typically, it is implemented as a type of crossover between the solutions. After a new trial solution is generated, it is necessary to apply the Improvement method (if it exists), in a way to guarantee that all the solutions generated are feasible, and only the final solution is considered to be compared with those that compound the *RefSet*.

The process continues iteratively with the employment of the Solution Combination method, followed by the Improvement method and then the Reference Set

Update method. This, until all the subsets have been subjected to the Solution Combination method and there are no further changes in the reference set [15, 61, 62, 83].

3.3. GRASP

The Greedy Randomized Adaptive Search Procedure (GRASP) [17, 18, 19] is a metaheuristic, which combines constructive heuristics and local search, usually applied to combinatorial optimization problems, and it is the one that we will use on the solving of the LA problem.

The GRASP metaheuristic [19, 84, 85] is a multi-start or iterative process, where each iteration is divided into two phases: construction phase where a greedy solution is produced and local search phase over that new solution, in which a local optimum in the neighborhood of the constructed solution is required. In Algorithm 3 it is presented the outline of the GRASP algorithm, where *imax* is a constant that indicates the number of the algorithm iterations, *f** represents the best found cost of the solution of the problem and *x* represents the solution. Considering the outline associated to the algorithm implementation, we may see that initially the solution has assigned an infinite cost, which will correspond to the worst possible one.

Algorithm 3 – Outline of GRASP Algorithm

GRASP Algorithm

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

After that, the initial solution will improve during a predetermined number of iterations where, in each iteration, it will be included the first phase of creation of a greedy randomized solution, using the respective method, and a second phase of improving that

solution with a local search method adapted to the LA problem (in our case). After these two steps we compare the cost of the new solution with the best until the moment and if the new one is better (this means the cost is lower), the solution is stored to the next iteration. Finally, concluding all the iterations we will obtain the solution with the best cost, which represents the best configuration of the network, by means of LAs and respective cells.

3.3.1 Greedy Randomized Method

There exist several variants of the GRASP algorithm [17] that are responsible for the specification of the method used to generate the greedy randomized solution, and principally, for the construction of the restricted candidate list (RCL).

Basically the RCL can be limited either by the number of elements (cardinality-based), which is the variant C, or their quality (value-based) that corresponds to the variant V. The variant RG is characterized by being, first random and latter greedy. Finally, the variant PG uses a perturbation factor to change the value of determined parameter [17, 18].

Besides these four variants of GRASP, we must also consider the bias function that modifies the mode of selection of the elements that are taken from the RCL. There exist several bias functions [17, 18], however we have used three types in our work, commonly referred as bias=0, bias=1 and bias=2, which respectively correspond to random, linear and exponential functions.

3.3.2 Local Search Method

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

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

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

In our case, where the LS is applied as part of the GRASP algorithm, it has the objective of improving the initial solution getting a new one with lower cost. That can be achieved reorganizing each location area of the network by adding or subtracting one cell in each step of the improvement. In Figure 11a we have an initial configuration of a network and in Figure 11b we have the same network after one iteration of the local search improvement where the cell number 6 have been moved from the blue LA to the red LA.

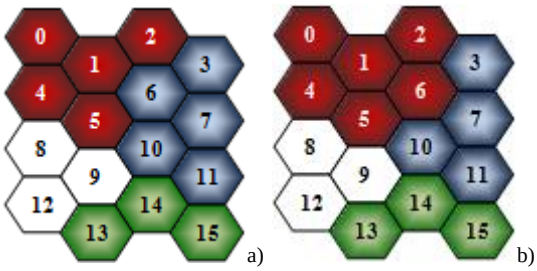


Figure 11 – Local Search improvement iteration:
a) Original network configuration; b) Network configuration after improvement.

4. Related Work

In this chapter we address the main works related to the research that we have executed considering the Location Management problem. We noticed that there exist several research groups working with the Location Area and the Reporting Cell problems, the two more common strategies of Location Management.

We start exposing the main research over the Location Area strategy, presenting the work developed with an outline of the main characteristics of each implementation and the major conclusions shown in each work.

After that we outline the main research results over the Reporting Cell strategy. Considering each work that we have analyzed, and when it was possible, we indicate the major contributions and the data used. Finally we expose the conclusions presented by each author.

4.1. Research approaches over Location Area Strategy

The location area problem have been implemented by several authors that are applying computationally efficient algorithms like genetic algorithms (GAs) [6, 7, 86], tabu search (TS) [86], simulated annealing (SA) [20, 86] and clustering techniques [87] (among others).

Following, we expose the more relevant results achieved in those works, considering the main characteristics of the implementations and results obtained.

In [6] Genetic Algorithms are applied to the location area partitioning problem with the main objective of grouping cells in an efficient mode, preserving bandwidth. Genetic algorithms are applied by using elitism, linear normalization of chromosome and it is used edge-based crossover to speed up the convergence time. The author uses systems with different number of cells and arbitrary values to the parameters. The author considers that near optimal solutions are obtained in an acceptable computation time and considers that the implemented algorithm is robust and flexible because easily could be extended

to incorporate other parameters or characteristics of cells. The author explicitly explains how to apply the algorithm and what fitness function uses but does not present the cellular system used, so we could not test our approach to compare results.

In [7, 20] the authors present respectively the application of a modified Genetic Algorithm and Simulated Annealing to location area scheme with the objective of finding the optimal location area configuration. When applying the genetic algorithms, the authors have defined a different mutation operator, applying changes to the classical one. They introduce three types of mutation: gene mutation, location areas split and location areas merge. In the application of simulated annealing it was developed a process to modify the reheating pattern and to compute the energy difference between two consecutive states to get a faster convergence.

In both works the authors refer that consider the obtained results very good and that the actual GSM networks are far from optimality. In these works simulation networks are presented and the data used are explicit and available to future use. So we decided to use these networks to apply our approach. The results obtained are very similar to the ones of these works and that gives us strength to continue, considering that we are in a good direction in this investigation.

The work presented in [86] considers that an optimal location area problem algorithm exhibits a very high complexity. This work has the objective of presenting the development of three different computationally efficient algorithms (genetic algorithms, simulated annealing and tabu search) to perform comparative studies and analyze the quality of the solutions. The developed algorithms were applied to micro cells and macro cells systems that are represented, but the network data is not available for future comparisons. The authors consider that the achieved signalling costs are comparable and represent good solutions, however tabu search algorithm presents better results when compared with the other two algorithms.

In [88] the authors are presenting the application of Simulated Annealing to location area planning and consider that the results are very interesting and much better than the results obtained by the application of multiple runs of greedy search. In this work the authors just refer that the data sets are obtained from a GSM network in Istanbul, but they do not present the network data used to obtain the results. Because of that, it was not possible to us apply our approach and compare results.

In [87] the authors start by applying clustering algorithms in combination with the location area scheme, with the objective of solving the location management problem. They start by leading with the problem in a static field with the intention of using that movement history as input information of users' location to the dynamic location management. The results presented indicate that, when using these techniques, around sixty per cent of location updates can be saved once compared with the always-update scheme and also indicate that less than one per cent of network pagings are terminated without success.

The work exposed in [89] presents a new approach based on the Hopfield Neural Network (HNN) to solve the location management problem and find the optimal location area configuration of a mobile network. In this work, the total cost of mobility management is related to the energy value of this artificial neural optimizer and the modification of the searching procedure is the main difference between this new approach and the generic HNN.

With the results achieved the authors consider that current GSM networks are far from optimality and that simulation results are very promising considering that they lead to network configurations that are unexpected. It is stated that the most important findings of this work that could influence the design of the future GSM networks are relevant to the shape of location areas, the number of cells in each location area, the number of neighbors for each location area, and the location areas bounding cell properties.

In [90] a combination of a Genetic Algorithm and the Hopfield Neural Network is used to find the optimal configuration of location areas in a mobile network. In this work the location areas configuration of the network is modelled so that the general condition of all the chromosomes of each population improves rapidly by the help of a Hopfield Neural Network. The Hopfield Neural Network is incorporated into the genetic algorithm optimization process, to expedite its convergence, since the generic genetic algorithm is not fast enough.

4.2. Research approaches over Reporting Cell Strategy

Relatively to the reporting cells problem, also several computationally efficient algorithms have been applied, like GAs, TS and Ant Colony algorithm (AC) [91], or a combination of the Hopfield Neural Network (HNN) and Ball Dropping Technique (BDT) as in [92] (among others).

Analyzing these related works, we had the goal of studying and using, when possible, the same network data (many of them are data from realistic networks) with the objective of comparing and analyzing the results achieved with our approaches.

In [91] the authors use a Genetic algorithm, Tabu Search, and Ant Colony algorithm to solve the reporting cells planning problem. Results of the experiment show that the three methods can be effectively used for reporting cells.

In [92] it is presented an approach (HNN-BDT-PC), which is a combination of the Hopfield Neural Network and the authors' Ball Dropping Technique, to solve the reporting cells problem. The approach is inspired by the phenomenon that results from the natural movement of balls when they are dropped onto a non-even plate (a plate with troughs and crests). Each trough of the plate corresponds to a reporting cell, and the network corresponds to the whole plate.

In [10] the authors use two different nature inspired algorithms to assign the reporting cells of a network following the reporting cells scheme. The first algorithm, designated Geometric Particle Swarm Optimization (GPSO), is a generalization of the Particle Swarm Optimization (PSO) for virtually any solution representation, which works according to a geometric framework. This approach is customized for the Mobile Location Management (MLM) problem by using the concept of Hamming spaces. The second algorithm consists of a combination of the Hopfield Neural Network coupled with a Ball Dropping technique. The location management cost of a network is embedded into the parameters of the Hopfield Neural Network. The authors consider that the results presented are very encouraging for current applications, and show that the proposed techniques outperform existing methods in the literature.

The work presented in [93] introduced a new concept to minimize the location management cost by maintaining the mobility history. The authors proposed a new technique which maintains the history of mobility patterns (of size *h*) of the last visited reporting cells [94]. The location update is not performed when the user roams within the reporting cells that belong to his mobility pattern. This means that the location information is updated when the user enters to a new reporting cell, which is not in his history. This new technique presents a reduction of the update cost proportional to the number of entries in the history.

In [95] the authors present a new approach based on simulated annealing technique to solve the Reporting Cell Planning Problem [96] of location management in mobile computing system. In this work, a simulated annealing meta-heuristic is used to obtain near to optimal solution for a reporting cell planning problem. The simulation results presented were executed over test networks with 19 and 36 cells. The authors consider that the simulation results show that simulated annealing algorithm can be effectively used to obtain near to optimal results for reporting cell planning problem. They showed that it can produce better results than the two classical location management strategies, respectively, always-update and never-update.

5. Development over the Location Areas Problem

After the classical strategies, always-update and never-update, the Location Areas (LA) strategy is probably the most well-known and used static scheme of Location Management problem. The LA scheme is used with the objective of minimizing the involved costs and reducing signalling traffic caused by paging messages and location updates in cellular network systems.

Concerning the LAs strategy, we have developed two main approaches based on Differential Evolution (DE) and Scatter Search (SS) algorithms. We also have studied and implemented, but in cooperation with other members of the research group, a third approach based on the Greedy Randomized Adaptive Search Procedure (GRASP).

Our main goal is to determine and understand the optimal configuration of each algorithm when applied to the LA strategy, through the intensive study of the most adequate values for each set of parameters.

In this chapter, we will expose the implementation of the approaches developed to tackle the Location Areas problem. We start by exposing the generic implementation considerations, which are common to all the approaches developed to solve the LA problem. After that, we expose the main experiments executed with the goal of determining the best configuration of each algorithm implemented through the tuning of respective parameters.

Relatively to the implementation considerations we will detail the source and preparation of the networks and then we explain the location management cost calculus, through the fitness function used. After that we will expose the most significant decisions and adjustments of our algorithms implementation, which includes the respective definition for the initial values of each parameter.

Following, we will present and analyse the major experiments executed and respective results achieved through each of the three approaches developed. It is important to refer that the experiments performed and the analysis of results take in

consideration the main objective of reaching the most adequate configuration for the specific parameters of each algorithm used.

To accomplish the experiments defined to study the LAs strategy we have used 4 diverse test networks and also one other bigger network based on real user network behaviour. These networks will be explained in section 5.1.1 *Networks Configuration*.

Finally we have the intention of compare the results achieved by DE, SS and GRASP based approaches with the results presented by Taheri and Zomaya in [7, 20, 89, 90], which were obtained with very different algorithms as GA (Genetic Algorithm), HNN (Hopfield Neural Network), SA (Simulated Annealing) and GA-HNNx (different combinations of Genetic Algorithm and Hopfield Neural Network [90]).

5.1. Implementation Considerations

Before we start the exposition of experiments of each approach developed, over the LA problem, we must expose some common implementation considerations. In the following subsection we will present and explain the configuration of the networks used by each approach and also the fitness function applied to evaluate each potential solution generated.

5.1.1 Networks Configuration

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 [7, 20] and also used in our previous work [97]. These test networks have distinct sizes and include a data set of information for each cell, including the number of location updates and incoming calls, as presented in Table 2 for the 5x5 network from [20]. 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 neighbors change their LAs to the same one. In this work, we initial use four distinct networks with respective sizes of 5x5, 5x7, 7x7 and 7x9 cells, presented respectively in Table 2, Table 3, Table 4 and Table 5, from [7, 20], with the objective of test the performance of DE approach applied to networks with distinct sizes.

Table 2 – 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 3 – 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) (2:22,27) (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) (2:17,33) (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)

Table 4 – 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	11	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)

Table 5 – 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)

5.1.1.1 SUMATRA: Realistic Data

Beyond the use of these test networks, we decided to test our DE and SS based approaches using more realistic data. These data were obtained from SUMATRA [22, 23]. 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 [22]. This test network is compound by 90 cells and 66,550 mobile users.

5.1.2 Fitness Function

The fitness function for the LA based approach will be defined according to (3), which is explained in section 2.3.3 *Location Management of Location Areas Total Cost*. This fitness function corresponds to the calculus of the location management costs involved, which means that for each generated solution (network configuration by means of LAs), we need to calculate its fitness value (i.e., the sum of the total cost for each of the LAs that compound the network).

5.1.2.1 Fitness Function using a Two-Step Paging

However, relatively to the experiments where we use realistic data, the fitness function will be applied by using a process of paging in two steps. This process was initially proposed by Subrata and Zomaya in [24] and applied to dynamic LAs, with the objective of maintaining a minimum time delay to locate each user and in order to preserve the required level of Quality of Service (QoS).

The process is divided into two steps because in the first step the paging is performed only to the last known localization of the user (it is considered that the initial localization of each user is known). If the user is not found in the first step, it is necessary

to apply the second step that consists in making a network wide search. Although, considering that we are applying the LA strategy, this wide search is just performed over the other cells (excluding 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.

5.2. DE Based Approach

An important part of our work was the development of the DE based approach applied to the LAs problem. For the best of our knowledge it was the first time that this evolutionary algorithm was used to solve the LM problem through the LA strategy, and the results achieved are very promising when compared with those obtained by the classical strategies and also by other authors' approaches.

In this section we start by explaining the implementation details which are specific for the DE implementation, respectively, the validation of each potential individual generated and the initial definition of DE parameters. After that, we explain and present the most important results of all the experiments that we have performed, over four test networks, with the goal of tuning the DE parameters and determining the best DE configuration when applied to the LAs problem. Finally, we also decided to perform the experiments of defining the best DE configuration, when applied to realistic networks, making the analyses and comparison of the obtained results.

5.2.1 Implementation Details

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

Considering the algorithm specificities we must consider the validation of each potential solution and also expose the most significant decisions and adjustments of the DE algorithm implementation, which includes the respective definition of initial parameter's values.

5.2.1.1 Individuals Validation

An important point that must be considered in the implementation process is the validation of each potential solution generated. We only may consider valid solutions, but when the process is executed there are generated some invalid solutions that need to be revised or discarded. A solution is considered to be invalid when it represents a network configuration that includes scattered LAs, where there exist cells belonging to the same LA but that are not connected between them, like the example shown in Figure 12.

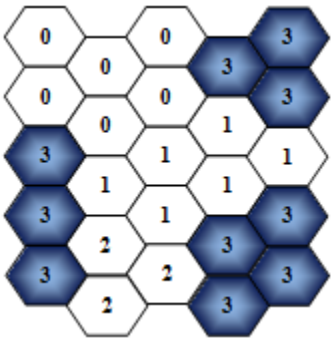


Figure 12 – Network configuration that includes a scattered Location Area (LA 3).

With the aim of solving this problem, we developed a set of methods to verify and make feasible every potential solution. This process, which needs to be repeated for all the potential solutions generated, to assure that the final solution will be a valid one, includes one first method to split the scattered LAs into small ones. Following, we designed a second method to merge LAs, because we do not want LAs with only one cell, when all their neighbors belong to different LAs. Finally, we have defined a third method to renumber the LAs because during this 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, like the one exposed in Figure 13 [103].

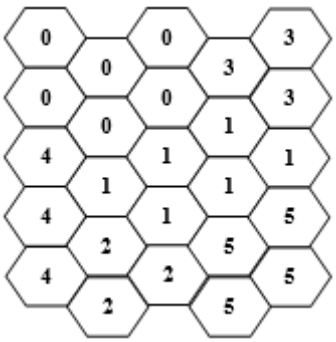


Figure 13 – Network configuration after individuals' validation.

5.2.1.2 Parameters Definition

For the implementation of the DE based approach we first need to consider the definition of the initial population of individuals. Each individual, or candidate solution, represents a possible network configuration and it is compound by *N* genes. *N* corresponds to the number of cells in the network, and each gene includes the information about the number of the LA where the cell belongs to.

In the generation of the initial population we consider, like in the works [7, 97], only two LAs, where one of them is set to each cell with a probability of fifty percent. Then we follow adjusting the parameter values to the ones elected in each experiment. For the initial experiment we set the population size (*NI*) to 10, the value of crossover parameter (*Cr*) to 0.1 and the value for the mutation factor (*F*) to 0.5. Relatively to the DE scheme we selected *DE/rand/1/bin*, because it was the one presenting best performance in testing experiences. Finally, we defined that the termination condition would be the number of generations, which the algorithm should be executed, and set it to 1000.

5.2.2 Experiments and Analysis

In this section we will expose the different experiments performed through the DE based approach, the results obtained and the respective analysis and conclusions taken.

The study of the DE based approach had the objective of determining the best configuration of parameters, when applied to the LA problem, and to accomplish this

assignment we have executed four main experiments, each one adjusted to the DE parameters, respectively, population size *NI*, crossover value *Cr*, mutation factor *F* and also the DE scheme.

We have also calculated the cost values obtained with the classical strategies (always-update and never-update), using all the test networks, with the objective of comparing results.

To assure the statistical relevance of the results and conclusions exposed, we decided to execute 30 independent runs, for each configuration tested. Furthermore, we have also performed a statistical analysis using the ANOVA test, considering a confidence level of 95% (i.e. significance level of 5%, which means a p-value under 0.05).

Like other authors, as Taheri and Zomaya [7, 20], in this study, four distinct test networks, presented in section 5.1.1 *Networks Configuration*, are used to ensure the reliability of results. The fact that the results are similar for those test networks (existing networks of different sizes, respectively from small to medium size) ensures that the best configuration of parameters can be generalized to any network.

After that, in the next section, we will apply the best configuration of DE to a realistic network based on SUMATRA data [22, 23] with the objective of test our approach with major and realistic networks.

5.2.2.1 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 [98, 99]. 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 6) 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 different colours in the tables of results. The red colour was used to mark the best fitness values. The green colour was used to point the best of the maximum (worst) fitness values. Then a yellow colour was used to show the best average fitness values and the blue colour was used to point the best median fitness value. Finally the grey colour was used to highlight the minimum standard deviation values.

Table 6 – Results of the Experiment 1: Defining the best *NI*.

Fitness Evaluation													
NI	**10**	**25**	**50**	**75**	**100**	**125**	**150**	**175**	**200**	**225**	**250**	**275**	**300**
5x5 Network													
Best	27216	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990
Worst	41152	28104	27992	27637	27536	27464	27522	27464	27336	27336	27336	27282	27336
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
Median	27945.5	27425.5	27282	27336	27216	27216	27182	27048	27153	27211	27048	27048	26990
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
Worst	57123	44005	43120	43424	42600	42919	43033	42483	42635	42690	42236	42227	41852
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
Median	42925.5	42079	41893.5	41662.5	41346	41488	41484.5	41217	41340	41465	41340	41303	41080.5
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
Worst	109549	69760	68273	68280	68628	68348	67525	67606	67869	67414	67624	67359	66986
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
Median	69052.5	68036.5	67401	67033	66914	66523	66347.5	66172.5	66404	66041.5	66506.5	65566	65963.5
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
Worst	106179	102941	101975	101204	100589	101035	100794	100774	100014	99855	99829	99475	99375
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
Median	102791	100588	99841	99466.5	99463	98701	98768.5	98094.5	97934	98280	97647.5	97568	97861.5
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

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.

5.2.2.2 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 7 – Results of the Experiment 2: Defining the best *Cr*.

Fitness Evaluation										
Cr	**0.01**	**0.03**	**0.05**	**0.07**	**0.09**	**0.1**	**0.25**	**0.5**	**0.75**	**0.9**
5x5 Network										
Best	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990
Worst	27499	27512	27398	27336	27464	27336	27216	27336	27398	27398
Average	27249.0	27277.3	27159.5	27145.8	27107.2	27070.6	27038.6	27090.4	27086.5	27136.1
Median	27213.5	27215.5	27211	27211	27048	26990	26990	27048	27048	27048
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
Worst	43359	43114	42765	42424	42433	42086	42056	42318	42919	43318
Average	41661.7	41552.2	41674.8	41763.8	41405.4	41188.6	41228.7	41398.2	42384.7	42616.6
Median	41610	41475.5	41721.5	41868.5	41354	41138	41227.5	41474	42403	42571
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
Worst	68829	68010	67382	67819	67926	67567	67055	68852	70191	72662
Average	66819.0	66739.1	66368.1	66185.7	66029.1	66057.7	65915.5	67396.8	68937.2	69361.9
Median	66642	66817.5	66512	66208	66027	66009	65950.5	67836.5	69097	69091
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
Worst	102145	101299	100284	99687	100181	99373	101678	107827	109540	107819
Average	100178.8	98751.6	98362.5	98015.2	97943.1	97547.0	99332.5	103542.7	105689.5	105707.1
Median	100498.5	98883	98762	97967.5	98096	97441	99477.5	104316.5	105741	105755.5
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 followed 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 7), it is possible to conclude that really *Cr*=0.1 is the best and more stable value to obtain better results.

5.2.2.3 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 8 – Results of the Experiment 3: Defining the best *F*.

Fitness Evaluation					
F	**0.1**	**0.25**	**0.5**	**0.75**	**0.9**
5x5 Network					
Best	26990	26990	26990	26990	26990
Worst	27512	27336	27398	27336	27336
Average	27080.6	27072.7	27141.8	27134.3	27078.4
Median	27048	27019	27211	27153	27019
St. Dev.	123.7	106.8	125.5	112.8	114.2
5x7 Network					
Best	40473	40466	40328	40496	40328
Worst	42801	42600	42045	42440	42403
Average	41491.0	41221.5	41194.3	41287.8	41266.3
Median	41340	41138	41183.5	41318.5	41261.5
St. Dev.	558.2	536.1	447.2	477.9	475.3
7x7 Network					
Best	64893	64893	64671	64879	64207
Worst	67141	67796	67338	67448	67111
Average	66051.2	66140.0	66192.1	65993.3	65981.9
Median	66020	66036.5	66066	65971	66003.5
St. Dev.	554.3	749.8	602.5	679.3	790.2
7x9 Network					
Best	96220	95076	93040	94774	95105
Worst	100336	99779	100211	99355	100118
Average	98116.6	97821.8	97826.5	97884.3	97849.8
Median	98129	98051.5	97952	97910.5	97966.5
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 8, 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.

5.2.2.4 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 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 9), and that permits to obtain the best fitness value in three of the four test networks.

5.2.2.5 Statistical Analysis Using ANOVA

Furthermore, to assure the statistical relevance of the results and conclusions exposed, 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 10, where we can see that the fitness differences when we use distinct values for each DE parameter have been found as significant in most of the cases.

Table 9 – Results of the Experiment 4: Defining the best DE Scheme.

Fitness Evaluation										
Scheme	**Exponential Crossover**					**Binomial Crossover**				
	Best1	**Rand1**	**RToB1**	**Best2**	**Rand2**	**Best1**	**Rand1**	**RToB1**	**Best2**	**Rand2**
5x5 Network										
Best	27282	26990	27048	27211	27211	27048	26990	27048	26990	26990
Worst	28247	28144	28485	28242	28242	27536	27336	28036	27637	27398
Average	27871.2	27620.37	27946.3	27784.17	27572.8	27304.93	27077.77	27420.63	27291.37	27102.67
Median	28008	27586.5	28052	27824	27524	27279	27048	27431	27282	27019
St. Dev.	305.049	297.275	395.853	354.614	295.87	132.90	111.40	255.54	169.40	125.56
5x7 Network										
Best	41141	41141	40722	40722	41340	40706	40205	40645	40346	40525
Worst	45077	45372	44623	43972	44683	42720	42600	43560	43108	42305
Average	42772.1	42499.07	42853.63	42497	42542.53	41691.97	41261.77	41917.43	41627.5	41351.4
Median	42610.5	42094	42784	42154	42124	41720	41141	41843	41661	41327.5
St. Dev.	987.131	1010.54	1136.64	794.39	927.93	400.767	562.21	676.12	600.44	387.18
7x7 Network										
Best	66215	65281	66243	65188	65658	64625	63307	64890	64560	65290
Worst	71240	70336	70282	71856	70149	68140	67398	68623	67676	67200
Average	67709.3	67509.97	68417.03	67708.2	67367.37	66366.3	65737.13	66976.23	66247.57	66273.23
Median	67568	67101.5	68827	67520	66668	66344.5	65711	66924	66345	66217.5
St. Dev.	1093.52	1330.84	1276.80	1399.43	1126.46	828.25	854.21	893.70	790.65	520.63
7x9 Network										
Best	100386	100484	98967	99512	100295	95125	94841	96408	92900	94483
Worst	105524	106960	106126	106188	105434	99473	99735	100101	100587	99635
Average	102580.57	103191.6	103152.4	103199.77	103100.77	97947.5	97895.13	98346.37	97479.8	97598.4
Median	102479.5	103116.5	102942.5	102884	103179.5	97896.5	97874.5	98238.5	97730	97745.5
St. Dev.	1166.39	1212.18	1643.05	1751.73	1213.01	996.365	1275.42	945.17	1408.71	1285.84

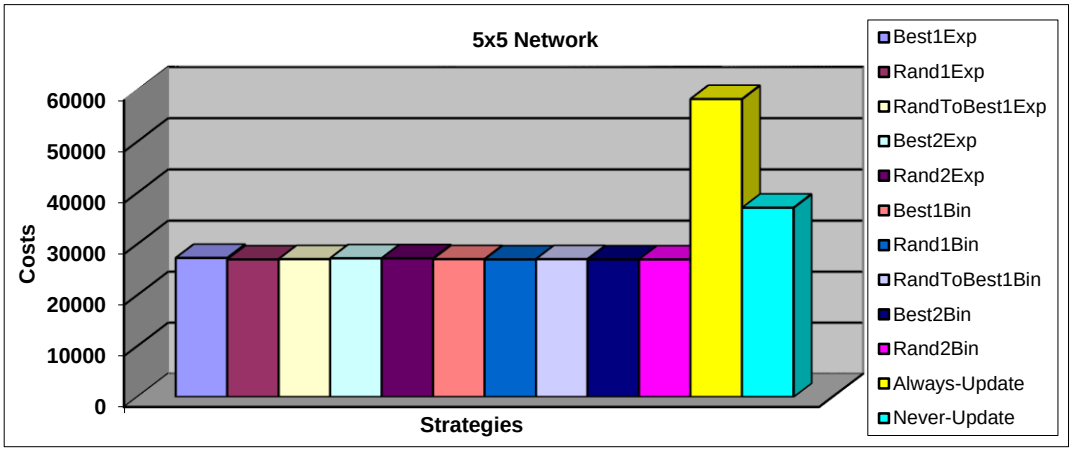
Table 10 – ANOVA analysis over DE parameters in the LAs problem.

***NI* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	2.10E-14	<1E-15	2.82E-14	<1E-15
***Cr* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	2.78E-15	<1E-15	<1E-15	<1E-15
***F* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	5.33E-02	1.81E-01	7.15E-01	8.66E-01

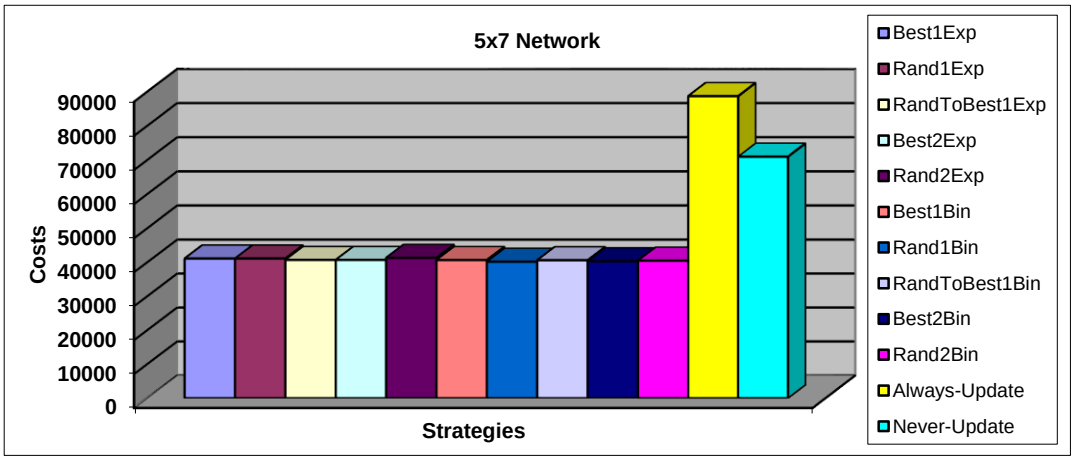
5.2.2.6 Analysis of Results

Concluding 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*.

After finishing all these experiments we started analysing the results achieved and comparing those with the ones obtained when the classical strategies are applied. Considering this, 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 Figure 14 and Figure 15.

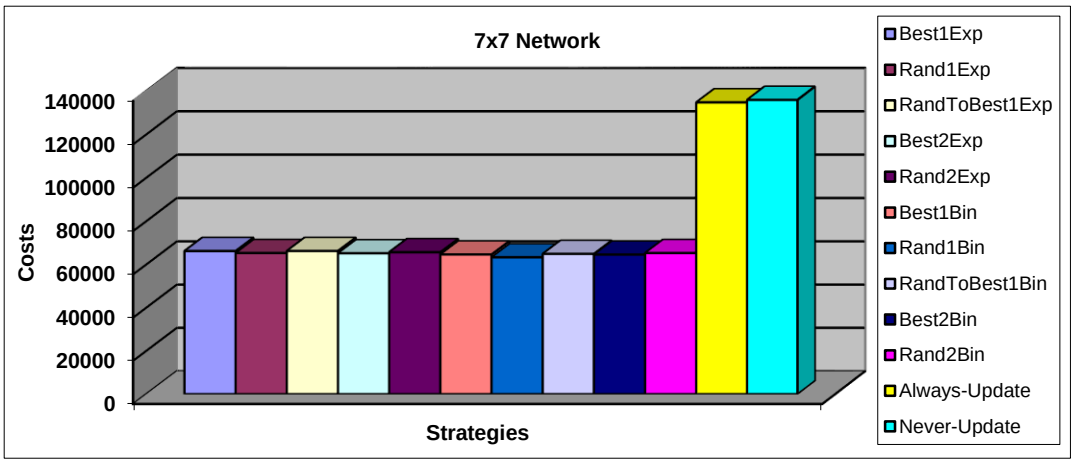


a)

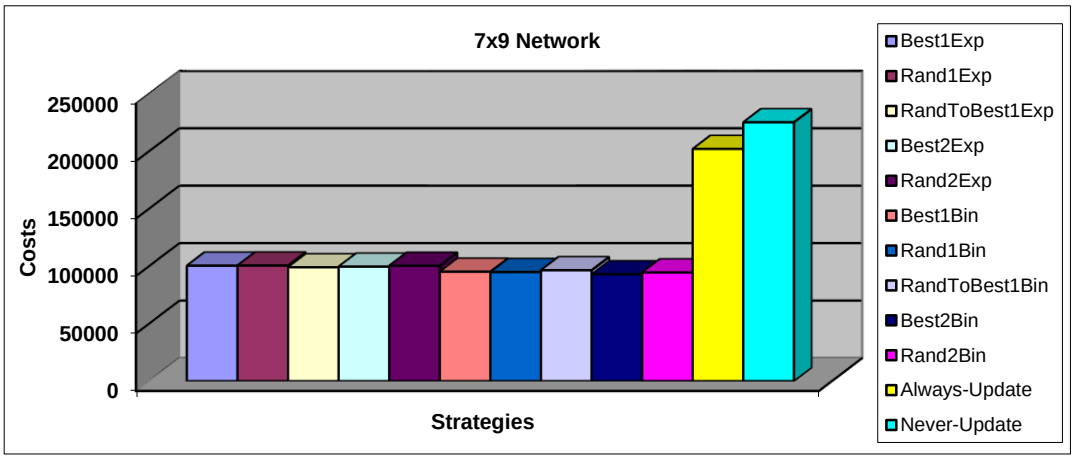


b)

Figure 14 – Comparison results of DE over LAs (Part 1): (a) 5×5 network, (b) 5×7 network.



a)



b)

Figure 15 – Comparison results of DE over LAs (Part 2): (a) 7×7 network, and (b) 7×9 network.

Relatively to the networks configuration of each best solution, we may say that all of these costs were calculated with the network partitioning defined by the DE algorithm and represented in Figure 16.

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 Figure 16(a)). When we refer to the 5×7 network, the best solution corresponds to a partitioning in 4 distinct LAs (see Figure 16(b)). Moving to the 7×7 network, the ideal partitioning is represented in 5 LAs (see Figure 16(c)). For the bigger network, the 7×9, the best configuration corresponds to a partitioning in 8 LAs (see Figure 16(d)).

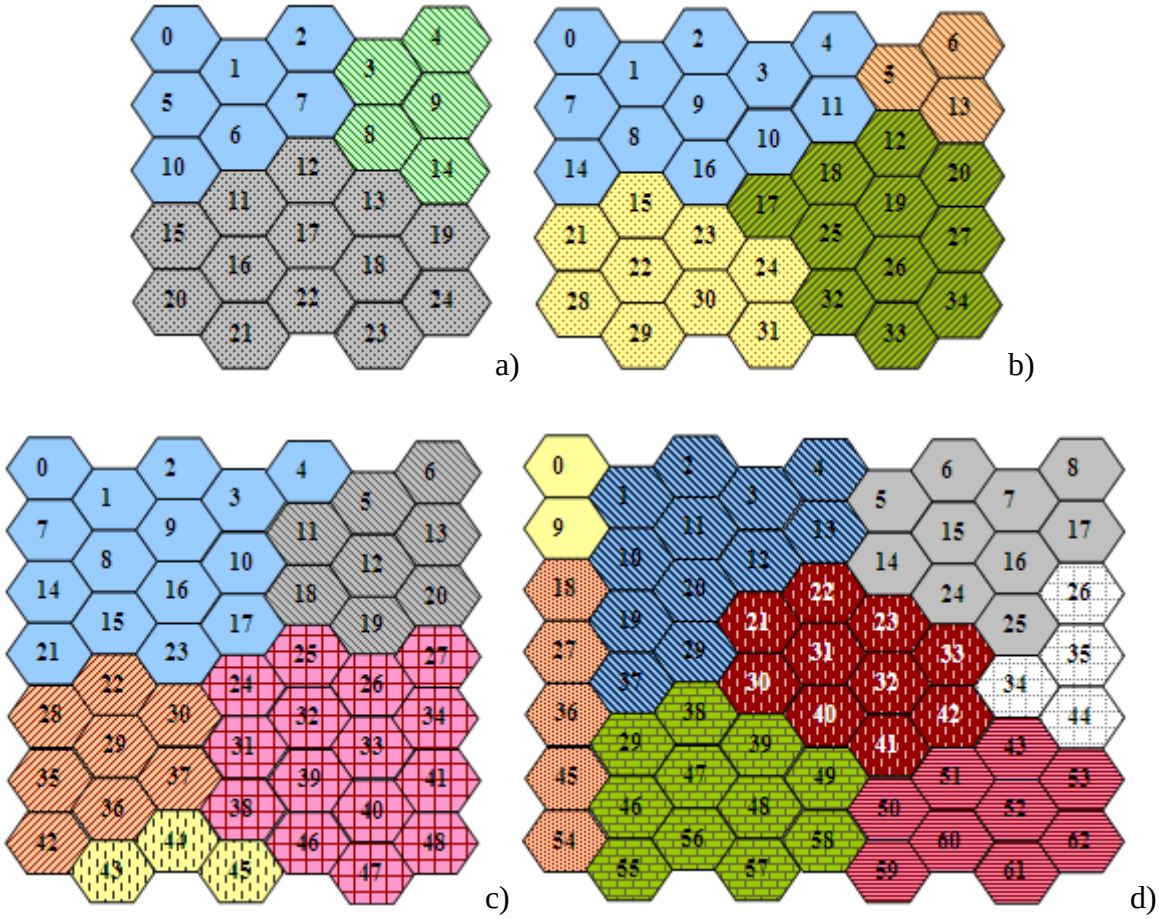


Figure 16 – Best DE LAs configuration: (a) 5×5 network, (b) 5×7 network, (c) 7×7 network, and (d) 7×9 network.

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.

In Figure 17 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 blue line corresponds to the evolution of the best fitness value (the solution with the lowest cost). The red line represents the evolution of the average fitness value. Finally, the green line shows the evolution of the worst fitness value.

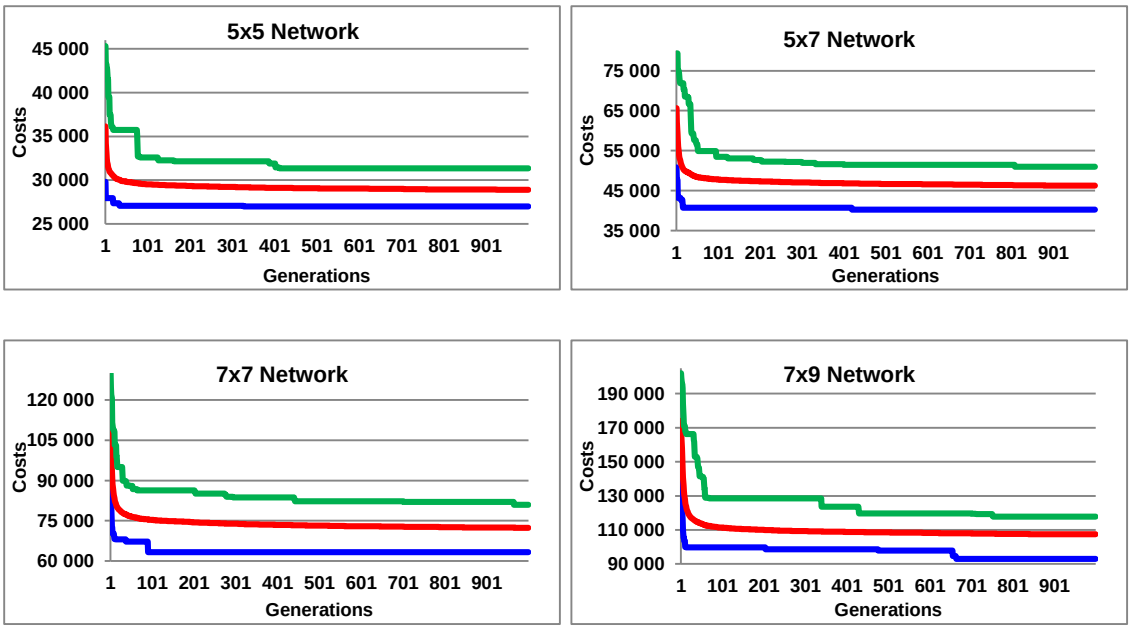


Figure 17 – Convergence curves of DE applied to the LA networks. The Y axis corresponds to the LM cost values and the X axis corresponds to the generations. Each graphic shows the evolution of the best (blue line), the worst (green line) and the average solution (red line).

Considering the computational CPU time we may say that it depends on the networks, principally by their size and data associated. In Table 11 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).

Table 11 – CPU time (s) of DE over LA experiments.

Network	Average	Best
5x5	41	40
5x7	60	60
7x7	89	87
7x9	124	122

5.2.2.7 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 [7, 90] 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 12 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 12 – Evolution of DE results over 5000 generations.

Test Network (Dim)	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 13 we compare the new results with the ones presented by Taheri and Zomaya in [90], 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 [90]), 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 13 – Comparison of network costs with different algorithms over LAs.

Test Network (Dim)	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

5.2.3 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 [22, 23] and using the BALI-2 trace.

In order to analyze and compare results, the values of always-update and never-update strategies were calculated for the test network, using the two-step paging process.

Then, with the objective of studying in more detail the best configuration of DE, when applied to realistic networks, 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 a network trace of medium size to validate our approach, when it is applied to networks based on real data patterns. For each test executed we calculate the location management cost for the complete network trace, as well as for each one of the 24 hours that the trace includes.

In this section we expose our different experiments and conclusions taken from each one to the next experiment and finally we analyse our results.

5.2.3.1 Determining the Best *NI*

The determination of the best number of individuals, *NI*, for the population is the objective of the first experiment. To accomplish that, we have fixed the values of *Cr*=0.1, *F*=0.5, DE scheme as *DE/Rand/1/Bin* and 1000 generations for the stop criterion, taking into account our experience from earlier experiments that we have executed [98, 99], and which the most important ones are exposed in section 5.2.2.

With this experiment we have concluded that, increasing the *NI* value, a positive evolution of the results (see the average fitness in Table 14) is produced, but 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*=300, to pass to the second experiment, as an equilibrium point between good results and times of execution.

The same conclusions were obtained when we evaluated the partial results, corresponding to every hour.

Table 14 – Experiment 1: Determining the best *NI*.

BALI - 2: 90 Cells Network			
Fitness Evaluation			
NI	***Best***	***Average***	***St. Dev.***
10	2831755	2858306.3	15054.4
25	2832313	2853917.8	13591.2
50	2816935	2840647.1	12334.7
75	2815512	2842153.0	11461.9
100	2825839	2839150.5	9023.9
125	2818853	2837443.7	9381.8
150	2821038	2833997.3	6909.2
175	2814416	2834466.4	9591.9
200	2816454	2833201.3	9466.7
225	2815534	2833626.4	7839.5
250	2812772	2831618.9	8001.3
275	2817306	2830843.0	6209.0
300	2815037	2829617.1	7734.0

5.2.3.2 Determining the Best *Cr*

In the second experiment we pretend to elect the *Cr* value that obtains the best results. To proceed with this experiment we initialized and fixed the value of *NI* to 300 (determined from experiment 1), and the other parameters as defined in the experiment 1.

After these parameters being fixed, the experiment was executed, using different values for *Cr*: 0.1, 0.25, 0.50, 0.75 and 0.9. Observing all the obtained results, we could conclude that the best value was *Cr*=0.1. However, to take more complete conclusions we decided to experiment lower values, from 0.01 to 0.09. Finally, looking to all the results provided from different *Cr* values (see Table 15), it is possible to conclude that

Cr=0.05 is the best and more stable value to obtain better results. This same behaviour has been observed also in the hourly partial results.

Table 15 – Experiment 2: Determining the best *Cr*.

BALI - 2: 90 Cells Network					
Fitness Evaluation					
CR	**0.01**	**0.03**	**0.05**	**0.07**	**0.09**
Best	2804291	2809056	2802159	2809541	2814811
Average	2821108.3	2819724.0	2818859.5	2821170.0	2826607.5
St. Dev.	6715.5	5892.6	6865.9	5749.8	5867.2
CR	**0.1**	**0.25**	**0.5**	**0.75**	**0.9**
Best	280919	2865100	3026010	3309097	3540267
Average	2830500.5	2890633.5	3083386.4	3422698.9	3667616.2
St. Dev.	8282.7	10860.2	26478.6	49048.5	58134.1

5.2.3.3 Determining the Best *F*

In the third experiment we have the objective of defining the best value for mutation *F*. So, in order to execute this experiment we fixed the value of *NI* to 300 (from experiment 1), *Cr* to 0.05 (from experiment 2) and the other parameters 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 *F* values of 0.25, 0.50, 0.75 and 0.9.

Observing the results obtained with this experiment (see Table 16), it is possible to verify that, the *F* value of 0.5 is the one that permits to obtain better results. Relatively to the hourly partial results, the ones with better performance are *F*=0.25 and *F*=0.5. In conclusion, *F*=0.5 was the elected one because it is the one that performs better when considering the fitness results, the fitness average evolution and the partial results.

Table 16 – Experiment 3: Determining the best *F*.

BALI - 2: 90 Cells Network					
Fitness Evaluation					
F	**0.1**	**0.25**	**0.5**	**0.75**	**0.9**
Best	2805624	2803237	2803200	2805278	2804848
Average	2817434.3	2818645.1	2816911.2	2818843.7	2818432.3
St. Dev.	5777.3	5617.8	5554.8	6526.9	6069.9

5.2.3.4 Determining the Best DE Scheme

After the three earlier experiments we have determined and fixed the best values for the DE parameters as *NI*=300, *Cr*=0.05 and *F*=0.5. Considering this, in this last experiment we have the goal of defining the most appropriate DE scheme, which permits to obtain the best results. For that, 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 a good election (see Table 17), because it obtains similar results to *DE/randtobest/1/bin*, being less complex, and with good results in both the best fitness and the average fitness. We also may say that the binomial schemes perform better than the exponential ones.

Table 17 – Experiment 4: Determining the best DE Scheme.

BALI - 2: 90 Cells Network					
Fitness Evaluation					
Exponential Crossover					
	Best1	**Rand1**	**RandToBest1**	**Best2**	**Rand2**
Best	2924672	2916871	2919480	2930842	2917442
Average	2969946.8	2973383.8	2965833.5	2965783.5	2974340.7
St. Dev.	33791.13	34707.87	33625.86	29749.62	33425.53
Binomial Crossover					
	Best1	**Rand1**	**RandToBest1**	**Best2**	**Rand2**
Best	2811583	2802388	2798884	2803461	2806182
Average	2821864.5	2817313.9	2807310.61	2821273.4	2818920.2
St. Dev.	14136.34	22497.44	14552.34	17610.00	14448.83

5.2.3.5 Analysis and Comparison of Results

Finishing these four experiments we had determined the best DE configuration, applied to the Location Area problem (using SUMATRA), setting the parameters as *NI*=300, *Cr*=0.05, *F*=0.5 and DE scheme as *DE/rand/1/bin*.

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 Figure 18.

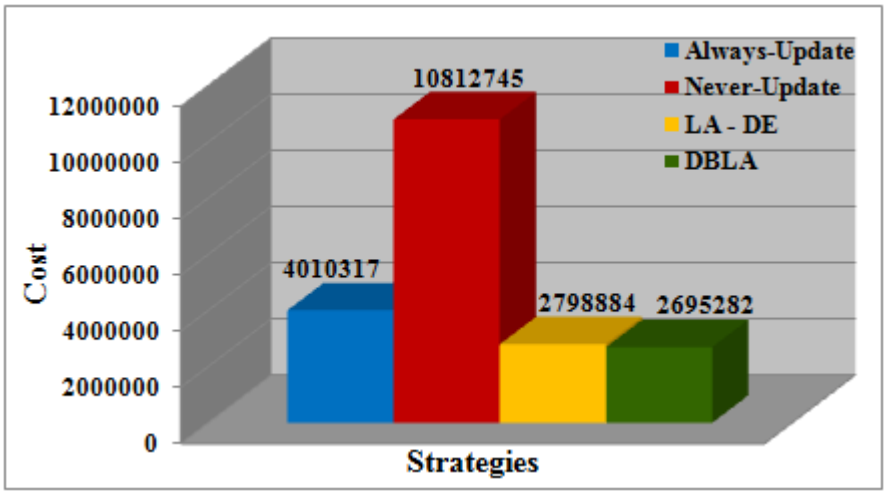


Figure 18 – SUMATRA: Comparison of strategies results.

Comparing our results with studies of other authors, as Subrata and Zomaya [24], 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 2798884 cost units, and DBLA obtains 2695282 cost units [24] for this network.

5.2.3.6 Analysis of Total Costs for each Hour

In Figure 19 we present the analysis of the hourly location management, 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 [24], they are very competitive, and this gives us the notion that our approach is viable.

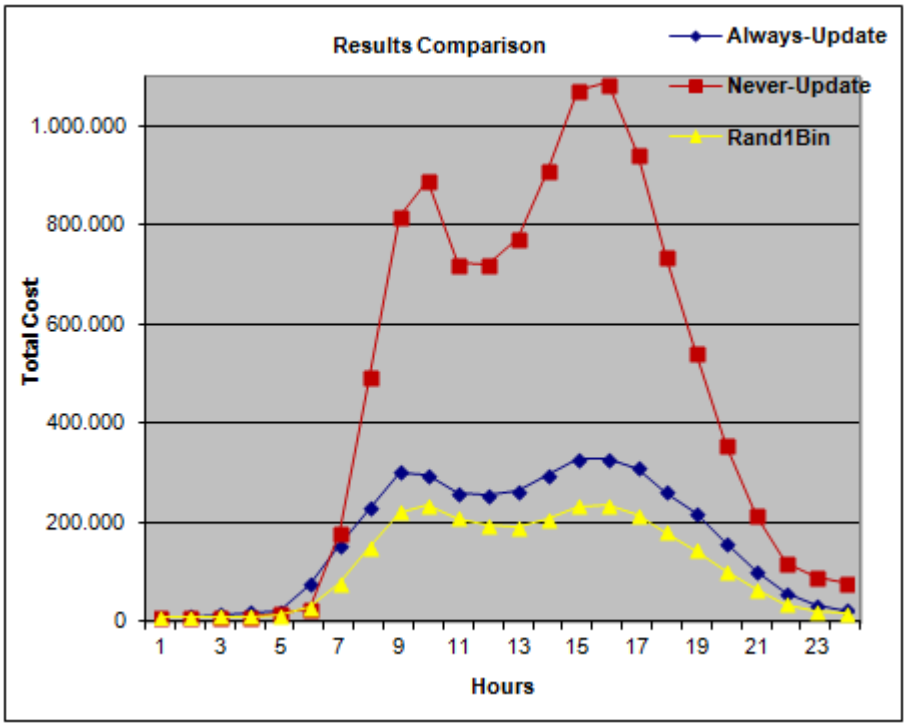


Figure 19 – Hourly LM Cost Comparison.

5.2.4 Summary

In this part of our work we have developed an approach based on the differential evolution (DE) algorithm applied to the location management problem with the objective of minimizing the involved costs. This approach is specified for the location areas (LAs) strategy of location management problem and applied to realistic and test networks.

Considering the results of the experiments performed, we have shown that it improves the results obtained with the 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.

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

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 test networks, are NI of 250, Cr of 0.1, F of 0.5 and DE/rand/1/bin as the best scheme.

We have also studied the best configuration of DE using networks based on realistic data with the objective of understanding the influence of the DE parameters and schemes over this type of networks. The best configuration achieved was *NI*=300, *Cr*=0.05, *F*=0.5 and DE scheme as *DE/rand/1/bin*, which is similar to the one obtained for test networks.

Analysing the best configurations achieved it is also possible to conclude that in general the binomial schemes perform better than the exponential ones.

5.3. SS Based Approach

After our study using the DE based approach that we have developed, we decided to study and implement a SS based approach also applied to the Location Area problem, with the objective of determining the best network configurations, which minimize the LM costs. We decided to implement this approach based on the SS algorithm because we have noticed, in our literature review, that this algorithm has shown good results in other optimization problems [62].

With this work we want to analyze and set the main parameters of scatter search, using four distinct test networks and also realistic networks based on SUMATRA data (the same ones as we have used with the DE based approach) and compare our results with those achieved by other authors. This is a new approach to this problem and the results obtained are very encouraging because they show that the proposed technique outperforms the existing methods in the literature.

This section is organized as follows. First we expose the implementation details specific for the SS algorithm implementation, which are respectively the individuals' validation and the definition of starting SS parameters. Then, we expose and detail the experimental results, over the four test networks, with the intent of defining the best SS configuration of parameters. After that, we have performed the same set of experiments applied to the realistic network, with the goal of defining the respective best SS configuration, making the analysis of the results obtained and finally comparing the results achieved by this approach with those achieved by DE based approach and also with the results obtained by classical strategies.

5.3.1 Implementation Details

Like for the DE based approach, in this section we will give explanation about the considerations that must be taken before implementing the experiments applying the SS based approach, concerned to the LAs strategy.

Each iteration of the SS algorithm leads to the generation of a set of potential solutions, which must be validated, and that will be the first point to take in consideration before starting the implementation. After that, we will expose the main decisions and adjustments of our SS algorithm implementation, considering the outline of the algorithm and including the respective definition of initial values for the four core parameters.

5.3.1.1 Individuals Validation

Like for the DE based approach, also in the implementation of the SS algorithm with the objective of solving the LA problem, we must consider the validation of each potential solution generated. This, because we only may consider valid solutions, but when the process is executed there are generated some invalid solutions that need to be revised or discarded.

The process, which is the same explained in section 5.2.1.1 *Individuals Validation*, corresponds to the three respective steps of split, merge and renumber the scattered LAs,

with the objective of guaranteeing that each potential solution considered in the results will represent a valid network configuration.

5.3.1.2 Parameters Definition

Considering the implementation of the SS based approach, and following the outline shown in Algorithm 2, we decided to apply a local search in the boundary cells of each LA, as the *improvement* method. This local search is characterized by switching a boundary cell of a LA to one of its neighboring LAs, as shown in Figure 20 where the cell number 12 changes from the blue LA to the red LA. With the *reference set update* method we had the intention of maintaining a set of the best and most diverse solutions. For the *subset generation* method we decided to implement subsets of size 2. Relatively to the *combination* method we developed a crossover that could be applied to a maximum of four crossover points according to a predetermined probability.

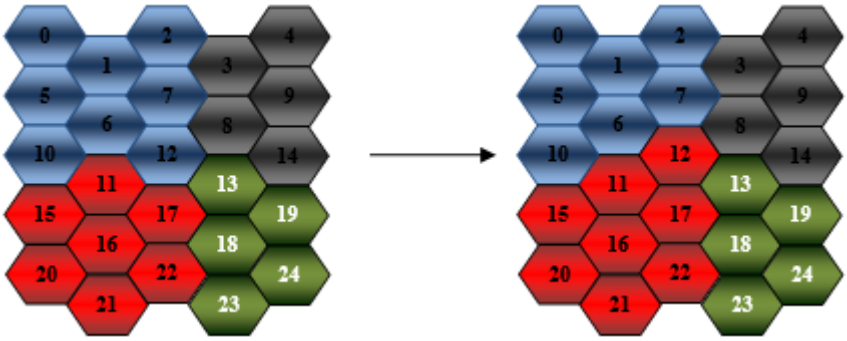


Figure 20 – Application of a Local Search in a boundary cell as the *Improvement* method of SS.

In conclusion, our SS implementation uses four core parameters: initial population size *PSize*; reference set size *RSSize*; probability of combination (crossover) *Cr*; and the number of iterations of local search *nLS*. Furthermore, the *RSSize* is divided into two parameters, the set size of the quality solutions *nQrs* and the set size of the most diversity solutions *nDrs*.

To proceed with the initial experiments we set the following values: *PSize*=100; *RSSize*=10; *nQrs*=5; *nDrs*=5; *Cr*=0.2; *nLS*=1, based on what several authors suggest [15, 16].

5.3.2 Experiments and Analysis

After the work using the DE based approach we decided to implement and study another approach, to solve the LA problem, based on the Scatter Search algorithm, with the objective of comparing the results achieved. We also want to understand what would be the most adequate alternative, considering the final solutions and the lowest LM costs.

In this section we will expose the different experiments accomplished using the SS based approach, observing the results obtained and presenting the respective analysis and conclusions taken.

With the objective of studying in detail the best configuration of SS, we have executed five distinct experiments, each one directed to one of the SS parameters.

In order to assure the statistical relevance of the results we have performed 30 independent runs, for each parameters' combination in each experiment. Additionally, we also accomplished a statistical analysis using the ANOVA test, and, similar to the DE based strategy, we have considered a confidence level of 95% (i.e. significance level of 5%, which corresponds to a p-value under 0.05).

In this study, like for DE based approach and like other authors, as Taheri and Zomaya [7, 20], four distinct test networks, presented in section 5.1.1 *Networks Configuration*, are used to ensure the reliability of results. Also for the SS we may conclude that the best configuration of parameters can be generalized to any network, because the results achieved are similar for those test networks (existing networks of different sizes, respectively from small to medium size).

After that, in section 5.3.3, we will apply the best configuration of SS to a realistic network based on SUMATRA data [22, 23] with the objective of testing our approach with major and realistic networks.

5.3.2.1 Experiment 1 – Defining the *PSize*

The first experiment had the objective of defining the best size of the population (parameter *PSize*). With the four test networks and using the initial values set to each parameter, we tested *PSize* with the values 10, 25, 50, 75, 100, 125, 150, 175 and 200. Analyzing the results, partially exposed in Table 18, we observed that, considering the four test networks, the best configurations for this parameter were 75 and 100. Finally, and after a more accurate statistical analysis, we decided to proceed for the second experiment with *PSize*=100.

Table 18 – Results of the Experiment 1: Defining the Best *NI*.

Fitness Evaluation									
NI	**10**	**25**	**50**	**75**	**100**	**125**	**150**	**175**	**200**
5x5 Network									
Best	26990	26990	26990	26990	26990	26990	26990	26990	26990
Worst	30868	27792	28180	27220	28142	28905	27832	28304	27516
Average	27845.67	27112.83	27127.83	27058.20	27072.00	27202.50	27123.73	27172.90	27084.57
St. Dev.	1087.03	166.70	261.82	104.18	215.46	406.80	188.59	294.56	148.20
5x7 Network									
Best	39832	39832	39832	39832	39832	39832	39832	39832	39832
Worst	43783	42633	43562	40085	40085	42665	42633	43200	43130
Average	40462.47	40096.60	40313.87	39914.40	39924.77	40044.50	40249.97	40252.53	40160.00
St. Dev.	1129.54	582.64	961.11	116.96	121.92	502.24	852.09	790.30	737.97
7x7 Network									
Best	60685	60685	60685	60685	60685	60685	60685	60685	60685
Worst	66436	63443	66446	66139	63232	66151	63710	66741	65218
Average	62154.33	61350.93	62216.93	61670.93	61347.30	61800.13	61723.33	61658.67	61838.37
St. Dev.	1242.02	734.84	1380.10	1213.97	856.49	1314.86	950.53	1422.95	1080.23
7x9 Network									
Best	89085	89211	89801	89085	89085	89551	89085	89085	89245
Worst	100405	101485	100882	99763	101884	99279	100009	104629	99010
Average	92950.90	93719.63	92942.43	93199.63	93115.67	93256.97	92984.03	92701.23	93265.03
St. Dev.	2695.68	3007.86	2383.35	2693.27	3625.55	2403.39	2613.73	3086.17	2925.68

5.3.2.2 Experiment 2 – Defining the *RSSize*

The second experiment, defining the *RSSize*, was defined to elect the optimal size of the *RefSet* that obtains the best result for all the networks. Using the *PSize*=100, determined in the first experiment, and maintaining the other initial parameters' values, we checked the following configurations for *RSSize*: 2, 4, 6, 8, 10, 12, 14, 16, 18 and 20.

Evaluating the results, including the best and the average fitness, shown in Table 19, we noticed that from the *RSSize*=6 it generally obtained good fitness values for all networks, but it was with the *RSSize*=18 that the best average results were obtained. Considering these results, we decided to elect *RSSize*=18 and follow for the third experiment.

Table 19 – Results of the Experiment 2: Defining the Best *RSSize*.

Fitness Evaluation										
RSSize	**2**	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
5x5 Network										
Best	27220	26990	26990	26990	26990	26990	26990	26990	26990	26990
Worst	31614	29620	28944	28180	28142	27282	27220	27249	26990	27216
Average	28985.63	27829.57	27263.47	27162.67	27106.07	27052.60	27012.73	27006.23	26990.00	27005.07
St. Dev.	1228.83	749.66	370.78	315.59	219.28	104.41	68.20	60.87	0.00	56.37
5x7 Network										
Best	40085	39832	39832	39832	39832	39832	39832	39832	39832	39832
Worst	51366	46094	43684	43562	43130	41821	40085	40085	40085	40085
Average	44768.20	41808.10	40476.27	40198.67	40137.83	39965.77	39899.47	39874.17	39846.03	39840.43
St. Dev.	2895.37	1912.11	1180.32	906.89	721.13	362.00	111.88	94.29	53.64	45.42
7x7 Network										
Best	62746	61357	60685	60685	60685	60685	60685	60685	60685	60685
Worst	82452	67240	66061	66761	65466	62977	62807	61916	62422	63003
Average	69828.47	64540.47	62573.17	62104.73	61557.23	60938.63	61032.70	60785.03	60780.63	60800.20
St. Dev.	5751.34	1838.82	1433.13	1392.11	1095.34	439.94	608.99	233.83	316.01	420.24
7x9 Network										
Best	90557	90115	89085	89085	89085	89085	89085	89085	89085	89085
Worst	109967	109268	100416	98982	97685	98061	99353	95779	99627	100927
Average	97655.17	95706.63	93398.63	94072.00	91932.90	92896.70	92931.00	91809.73	92038.93	92957.77
St. Dev.	5197.16	4104.87	2620.31	2405.86	2159.50	2159.39	2662.60	2099.61	2613.60	3116.05

5.3.2.3 Experiment 3 – Defining the *nQrs* and the *nDrs*

In the third experiment, defining the *nQrs* and the *nDrs*, we intended to determine the division of the *RefSet* between quality *nQrs* and diversity *nDrs* solutions. So, in order to execute this experiment, we assigned *PSize*=100, *RSSize*=18 and maintained the initial values of *Cr* and *nLS*. Using these values we tested all the possible configurations of *nQrs* and *nDrs* (knowing that their sum must be 18 and also that the number of diversity solutions must be equal to or lower than the number of quality solutions). The obtained results (see Table 20) were very similar for all the combinations. Because of that, we

decided to maintain an equal division of 9 solutions for each subset (the standard configuration of SS).

Table 20 – Results of the Experiment 3: Defining the *nQrs* and the *nDrs*.

Fitness Evaluation										
nQrs/nDrs	**9/9**	**10/8**	**11/7**	**12/6**	**13/5**	**14/4**	**15/3**	**16/2**	**17/1**	**18/0**
5x5 Network										
Best	26990	26990	26990	26990	26990	26990	26990	26990	26990	26990
Worst	27216	27249	27282	27282	28142	28142	27282	28903	29620	28304
Average	26997.53	27029.03	27007.27	27014.87	27082.50	27068.27	27016,87	27182.53	27178.83	27126.43
St. Dev.	40.57	87.43	65.17	75.23	219.67	218.55	75,49	451.86	498.02	273.83
5x7 Network										
Best	39832	39832	39832	39832	39832	39832	39832	39832	39832	39832
Worst	40085	40085	40085	40085	42776	39832	42420	40085	40085	40085
Average	39848.87	39874.17	39857.30	39846.03	39967.53	39832.00	39963,20	39840.43	39857.30	39871.33
St. Dev.	63.11	94.29	75.90	53.64	528.34	0.00	465,25	45.42	75.90	89.04
7x7 Network										
Best	60685	60685	60685	60685	60685	60685	60685	60685	60685	60685
Worst	62189	62422	63003	62807	62244	62845	61121	62929	62807	63269
Average	60885.57	60809.97	60880.20	60780.67	60873.03	60872.20	60731,53	60838.33	60942.37	60804.80
St. Dev.	411.32	321.98	568.28	379.99	400.79	531.76	90,31	460.39	470.49	462.96
7x9 Network										
Best	89085	89085	89085	89085	89085	89085	89085	89085	89801	89085
Worst	97180	97446	99317	102731	95224	95405	98031	100673	100037	97429
Average	91870.53	92251.30	92689.43	92623.40	91872.70	91680.10	91743,27	91522.47	92433.00	92053.83
St. Dev.	2040.74	2260.83	2945.89	2987.26	1625.67	1771.12	2249,96	2326.11	2556.06	2264.33

5.3.2.4 Experiment 4 – Defining the *Cr*

The fourth experiment, defining the *Cr*, was executed to elect the probability of combination (crossover) that obtains the best results for all the test networks. To proceed with this experiment we fixed the values obtained in the earlier experiments (*PSize*=100, *RSSize*=18, *nQrs*=9, *nDrs*=9), using the initial *nLS*=1, and testing the following values for *Cr*: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9. Analyzing the average results, exposed in Table 21, we could conclude that for the small networks the lowest values of *Cr* performed better but, for the bigger networks the results were more similar and that was not so obvious. Therefore, we decided to proceed with *Cr*=0.2, the one that had already given good results.

Table 21 – Results of the Experiment 4: Defining the Best *Cr*.

Fitness Evaluation									
Cr	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
5x5 Network									
Best	26990	26990	26990	26990	26990	26990	26990	26990	26990
Worst	29497	27216	27282	27216	27536	27220	27220	27220	27536
Average	2751787	26997.53	27014.93	27020.13	27033.00	27012.73	27027.80	27005.20	27023.27
St. Dev.	564.15	40.57	75.41	76.83	119.73	68.20	84.53	56.88	110.62
5x7 Network									
Best	39832	39832	39832	39832	39832	39832	39832	39832	39832
Worst	39832	40085	40085	40085	43097	41817	40085	40085	40085
Average	39832.00	39848.87	39840.43	39860.97	39971.73	39954.37	39880.67	39840.43	39882.60
St. Dev.	0.00	63.11	45.42	75.08	585.80	360.71	97.81	45.42	101.20
7x7 Network									
Best	60685	60685	60685	60685	60685	60685	60685	60685	60685
Worst	63314	62043	62845	61588	62845	60873	63025	61187	62422
Average	61362.50	60803.03	60969.80	60729.30	60773.07	60702.60	60865.23	60728.07	60783.40
St. Dev.	823.36	316.32	573.31	162.95	387.32	52.87	516.21	103.72	345.34
7x9 Network									
Best	89085	89085	89085	89085	89085	89085	89085	89085	89085
Worst	95678	99468	100106	91653	91163	92599	89797	90596	92789
Average	92159.70	91747.60	92689.73	89559.23	89545.53	89721.67	89275.03	89430.57	89422.70
St. Dev.	2113.22	2508.98	3023.65	688.97	580.86	836.26	244.34	429.31	786.04

5.3.2.5 Experiment 5 – Defining the *nLS*

The objective of this last experiment, defining the *nLS*, was to define the best value for the number of local search iterations *nLS*. We fixed the other parameters with the values obtained previously, and checked the following configurations for *nLS*: 1, 2, 3, 4, 5 and 6. Observing the results, presented in Table 22, we might see clearly that it was with *nLS*=5 that the best results (lowest costs) were reached, because with *nLS*=6 the average of fitness values started to increase.

5.3.2.6 Statistical Analysis Using ANOVA

As well as for the DE strategy, to assure the statistical relevance of the results and conclusions achieved, a statistical analysis using the ANOVA test has been performed. We consider here a confidence level of 95% (i.e., significance level of 5%, which corresponds to a p-value under 0.05), which means that the differences are unlikely to have occurred by chance with a probability of 95%. In Table 23, we expose the results

obtained using this test, and we can see that the fitness differences when we use distinct values for each SS parameter have been found as significant in most of the cases.

Table 22 – Results of the Experiment 5: Defining the Best *nLS*.

Fitness Evaluation						
nLS	**1**	**2**	**3**	**4**	**5**	**6**
5x5 Network						
Best	26990	26990	26990	26990	26990	26990
Worst	27220	26990	26990	26990	26990	26990
Average	27005.2	26990	26990	26990	26990	26990
St. Dev.	56.88	0	0	0	0	0
5x7 Network						
Best	39832	39832	39832	39832	39832	39832
Worst	40085	40085	39832	39832	39832	39832
Average	39874.17	39840.43	39832	39832	39832	39832
St. Dev.	94.29	45.42	0	0	0	0
7x7 Network						
Best	60685	60685	60685	60685	60685	60685
Worst	62807	60873	60873	60855	60753	60855
Average	60849.93	60709.6	60706.53	60705.8	60708.13	60716.93
St. Dev.	444.27	45.79	37.09	37.80	24.47	45.23
7x9 Network						
Best	89085	89085	89085	89085	89085	89085
Worst	96735	94165	93992	91925	90906	91065
Average	92634.13	91048.57	90092.5	89552.73	89485.4	89485.77
St. Dev.	2254.33	1660.28	1216.75	741.83	570.46	602.32

Table 23 – ANOVA analysis over SS parameters in the LAs problem.

***PSize* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	1.06E-11	7.14E-02	4.95E-02	9.67E-01
***RSSize* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	<1E-15	<1E-15	<1E-15	3.32E-14
***nQrs/nDrs* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	2.11E-02	2.35E-01	8.05E-01	5.46E-01
***Cr* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	<1E-15	2.27E-01	7.45E-08	<1E-15
***nLS* Parameter**				
Network	5x5	5x7	7x7	7x9
***p*-Value**	7.12E-02	6.73E-04	1.82E-02	<1E-15

5.3.2.7 Analysis of Results

After we finished these five experiments, we have achieved the best configuration for the SS parameters, when the algorithm is applied to the LAs problem: *PSize*=100, *RSSize*=18, *nQrs*=9, *nDrs*=9, *Cr*=0.2 and *nLS*=5.

Using this configuration of parameters we can obtain the best solutions, which correspond to the lowest fitness values, for all the four networks. In Figure 21 it is presented the best LAs configuration that we have reached for each network used.

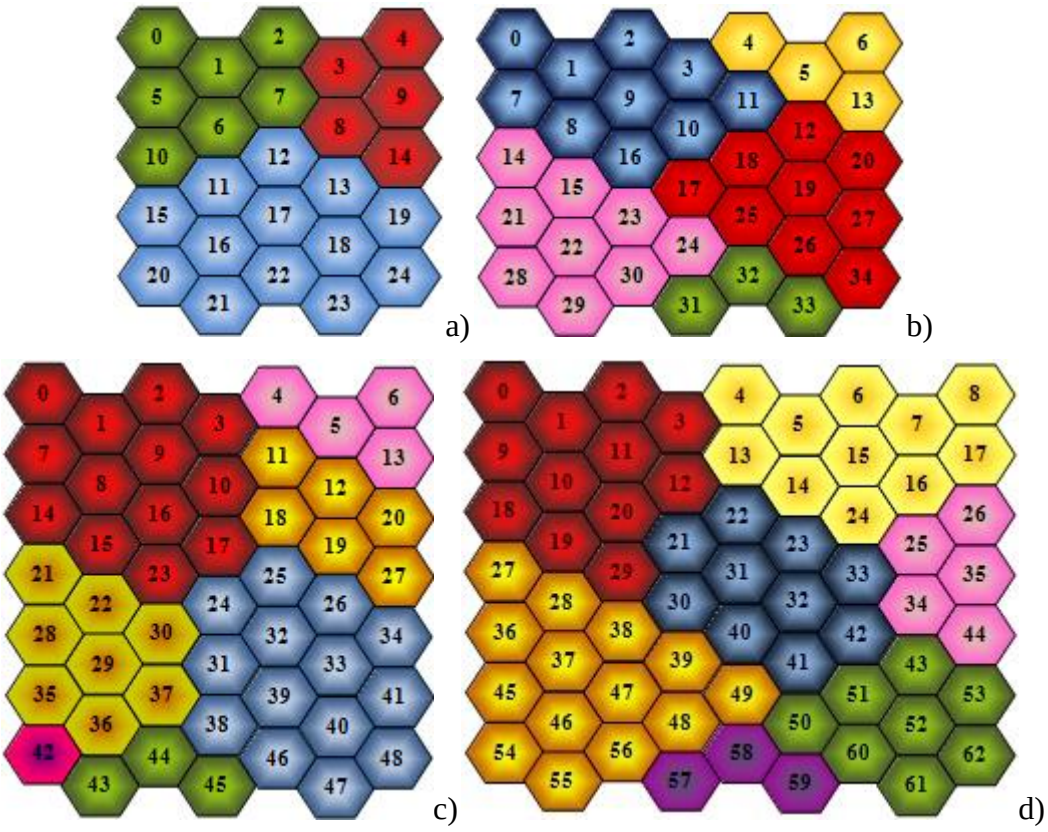


Figure 21 – Configuration of the best LAs achieved with the SS based approach: (a) 5x5; (b) 5x7; (c) 7x7; (d) 7x9 networks.

5.3.3 SUMATRA – The Use of Realistic Networks

Considering the good results obtained by applying the SS to the test networks, we decided to perform the tuning experiments over the realistic networks generated based on SUMATRA data [22, 23].

In this section we pretend to expose the experiments realized with this approach and analyze the results obtained. So, in order to analyze and compare results, the values of always-update and never-update strategies were calculated for the realistic network, using the two-step paging process. We also want to compare our results with those accomplished by other authors and with other approach, based on DE algorithm, developed by us.

With the objective of studying in more detail the best configuration of SS, we have executed five distinct experiments. For each experiment, and for every combination of parameters, 30 independent runs have been performed in order to assure its statistical relevance. For each test executed we calculate the location management cost for the complete network trace, as well as for each one of the 24 hours that the trace includes.

5.3.3.1 Defining the Population Size *PSize*

The objective of the first experiment is to define the most adequate size of the population (parameter *PSize*). To accomplish that we started by testing the *PSize* with the value 10 and followed increasing it until a *PSize* of 200 individuals, passing by the values 25, 50, 75, 100, 125, 150 and 175.

Table 24 – Experiment 1: Defining the Population Size - *PSize*.

BALI - 2: 90 Cells Network			
Fitness Evaluation			
PSize	***Best***	***Average***	***St. Dev.***
10	2770067	2818716.90	27390.90
25	2767978	2803911.27	24357.52
50	2770869	2809024.83	27375.63
75	2764829	2801723.63	25186.14
100	2759702	2796282.33	20532.83
125	2767072	2799137.70	22368.35
150	2766971	2796110.03	20071.66
175	2760254	2795013.63	23582.40
200	2762865	2795753.40	23653.35

Evaluating the results, that are presented in Table 24, we can observe that there is not a linear evolution of results but the best fitness value is obtained with the value 100

and the average of fitness normally has a positive evolution until the value of 175. Considering this, and evaluating the partial results, corresponding to every hour, that present the same behaviour of results, we decided to proceed for the second experiment with *PSize*=175.

5.3.3.2 Defining the *RefSet* Size *RSSize*

The definition of the number of individuals that will compound the *RefSet* is the objective of the second experiment. We set the *PSize* to 175, which was determined in the first experiment, maintaining the other parameters with the original values and tested *RSSize* with all these values: 2, 4, 6, 8, 10, 12, 14, 16, 18 and 20.

After obtaining the results using all the different values we could observe (see Table 25) that along the *RSSize* increasing, the evolution of the fitness average was positive until the *RSSize*=16, where the best fitness average is obtained. Analyzing also the hourly partial results we observed that is with this size of *RefSet* that the algorithm performs better, so we decided to follow with *RSSize*=16 to the next experiment.

Table 25 – Experiment 2: Defining the RefSet Size - *RSSize*.

BALI - 2: 90 Cells Network			
Fitness Evaluation			
RSSize	***Best***	***Average***	***St. Dev.***
2	2782198	2830068.43	38891.06
4	2764336	2808567.07	21205.40
6	2762892	2795416.67	20759.17
8	2758960	2795839.00	22041.31
10	2752114	2787986.60	19602.47
12	2773825	2796021.23	12576.84
14	2760288	2792865.00	17799.80
16	2756319	2787297.53	14768.22
18	2760702	2791011.27	14930.30
20	2753510	2788964.70	17642.09

5.3.3.3 Defining the Number of Quality *nQrs* and Diversity *nDrs* Solutions

The third experiment has the objective of defining the best division of the *RefSet* between quality and diversity solutions, this means defining the best values to the *nQrs*

and the *nDrs*. We started by setting *PSize*=175 and *RSSize*=16 from the earlier experiments, and maintaining the initial values for other parameters, then we tested the division of the *RSSize* within all the possible combinations (considering that their sum must be equal to 16 and knowing that the *nDrs* must be equal to or lower than the *nQrs*).

Once obtained all the results, presented in Table 26, we observed that, it is with a lower number of diversity solutions that the best results are achieved, more precisely it is the combination *nQrs*/*nDrs*=15/1 that performs better and presents the best average of fitness. The same conclusions were obtained when we observed the partial results that correspond to every hour.

Table 26 – Experiment 3: Defining the *nQrs* and *nDrs*.

BALI - 2: 90 Cells Network			
Fitness Evaluation			
RSSize	***Best***	***Average***	***St. Dev.***
8/8	2761964	2790928.67	14869.34
9/7	2757650	2789194.10	17533.81
10/6	2752113	2788186.63	19526.60
11/5	2750811	2791526.33	17955.39
12/4	2758285	2786020.23	12583.77
13/3	2762273	2789746.50	18781.92
14/2	2758913	2789163.97	16631.93
15/1	2756878	2785816.50	17232.25

5.3.3.4 Defining the Crossover Probability *Cr*

In the fourth experiment we intent to determine the probability of applying the combination method (crossover) that permits to obtain the best results. For this experiment we initialize the value of *Cr* with a probability of 0.1, and then the algorithm was also evaluated with the values 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9. The rest of parameters maintain the values obtained in our previous experiments.

Considering the results obtained we observed (see Table 27) that *Cr*=0.1 was the one that accomplish the best results. But, because it was the lower value we decided to test *Cr* also with the values 0.01, 0.03, 0.05, 0.07 and 0.09. Finally, as we can observe in Table 27, it is *Cr*=0.1 the most adequate decision to pass for the last experiment.

Table 27 – Experiment 4: Defining the Crossover - *Cr*.

BALI - 2: 90 Cells Network			
Fitness Evaluation			
Cr	***Best***	***Average***	***St. Dev.***
0.01	2763306	2786296.57	15203.90
0.03	2756006	2788594.63	16850.68
0.05	2757326	2787588.63	16078.42
0.07	2762509	2791238.77	18136.82
0.09	2758355	2786450.40	15672.59
0.1	2758038	2785114.07	16880.10
0.2	2753510	2787702.53	17755.62
0.3	2757764	2791493.97	21324.76
0.4	2759561	2789114.90	17719.35
0.5	2757753	2790212.87	17534.44
0.6	2755951	2786621.40	17093.97
0.7	2758960	2791630.77	18397.30
0.8	2762715	2789253.87	19496.06
0.9	2760781	2790843.20	14624.44

5.3.3.5 Defining the Number of Local Search Iterations *nLS*

Finally in this last experiment we have the goal of elect the most adequate number of local search iterations *nLS* in each solution improvement. So, we fixed the best values for each parameter considering the results of the previous experiments and testing the following values for *nLS*: 1, 2, 3, 4, 5, 6, 7, 8 and 9.

Analyzing the main results, presented in Table 28, and also the partial results relatively for each hour, we could conclude that it is until *nLS*=7 that the average of fitness presents a positive evolution.

Table 28 – Experiment 5: Defining the Number of Local Search Iterations - *nLS*.

BALI - 2: 90 Cells Network			
Fitness Evaluation			
nLS	***Best***	***Average***	***St. Dev.***
1	2757537	2786416.73	17606.37
2	2758728	2780192.40	12963.89
3	2755244	2778035.83	17274.04
4	2758285	2782365.50	12394.67
5	2755248	2775712.13	8638.13
6	2758960	2777731.33	11880.25
7	2756836	2773205.40	9228.69
8	2757639	2776201.30	11010.79
9	2754052	2774331.77	9655.85

5.3.3.6 Analysis and Comparison of Results

After these five experiments we have achieved the best configuration for the SS parameters applied to the LA problem, using SUMATRA networks: *PSize*=175, *RSSize*=16, *nQrs*=15, *nDrs*=1, *Cr*=0.1, and *nLS*=7.

Besides comparing the results between DE and SS based approaches, we also decided to compare them with those achieved by using the classical strategies always-update and never-update.

Observing Figure 22 we notice that our DE and SS based approaches always surpass the results obtained by the classical strategies. Comparing the SS based approach with the DE based approach we verified that SS performs better than DE (as shown in Figure 22). This is because the best solution achieved with SS has a cost of 2756836 and the DE best solution has a cost of 2798884.

After that we also compared our results with those achieved by other authors, as Subrata and Zomaya [24], which use dynamic LA strategies as Distance Based Location Area (DBLA). Considering that our best solution represents a fitness result of 2756836 cost units (for SS based approach) and their best solution, achieved with DBLA, is 2695282 cost units [24] we may conclude that our approach is very promising, due that we are using static LAs and they use dynamic LAs.

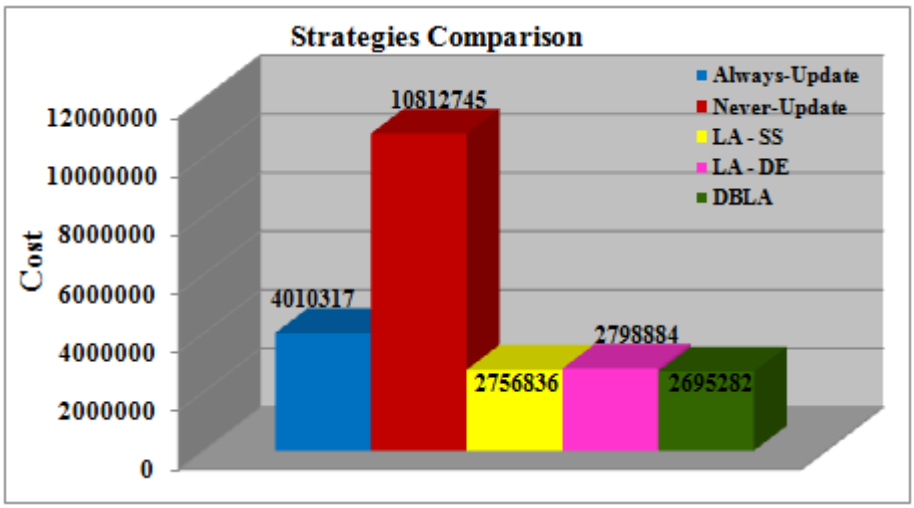


Figure 22 – Comparison of strategies/algorithms results.

5.3.3.7 Analysis of the Hourly Total Costs

Considering that BALI-2 [22] includes the 24 hours trace, we also calculated the partial results, of each hour, for the classical strategies always-update and never-update.

In Figure 23 we illustrate the LM cost obtained with the classical strategies, the DE based approach [100] and the SS based approach, and we could conclude that also in the partial results the SS outperforms the others.

If we compare these LM costs for each hour with the ones obtained by Subrata and Zomaya [24], we may say that they are very similar, which gives us the notion that our approach is very competitive and viable, considering that we are using static LAs and they are using dynamic LAs.

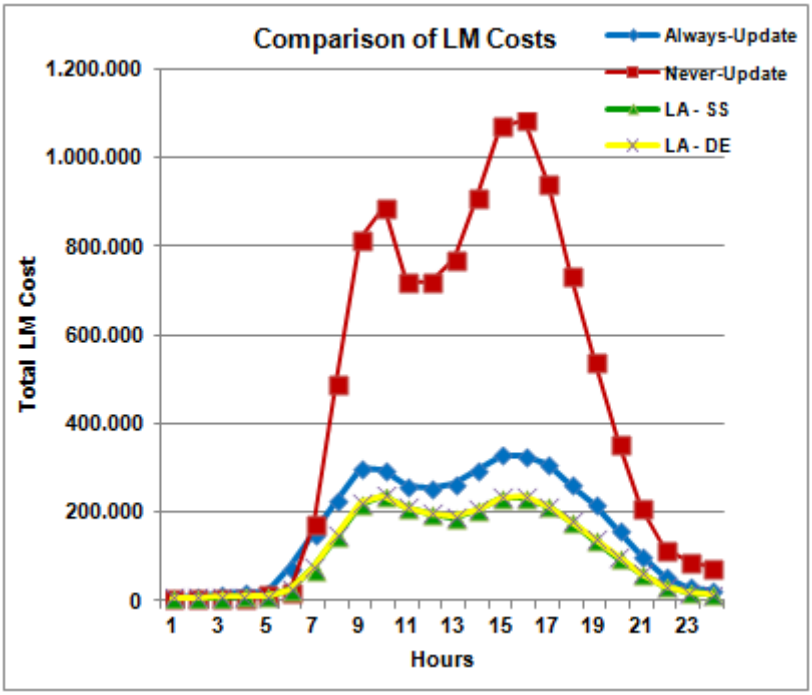


Figure 23 – SUMATRA: BALI 2 – Hourly LM Cost Comparison.

5.3.4 Summary

In this section we have presented a new approach based on SS algorithm with the objective of finding the best configuration for the LAs strategy in mobile networks.

We have studied in detail the best configuration of SS, applied to the LAs problem. The best parameters, after a big number of experiments with four distinct test networks, are *PSize*=100, *RSSize*=18 with *nQrs*=9 and *nDrs*=9, *Cr*=0.2 and *nLS*=5.

After that, we have used the scatter search (SS) algorithm to solve the location area problem applied to SUMATRA networks, with the goal of determining the best configuration of SS using these networks based on realistic data, and the best value for its parameters are *PSize*=175, *RSSize*=16, *nQrs*=15, *nDrs*=1, *Cr*=0.1 and *nLS*=7.

The results obtained by the SS algorithm outperform the results obtained with the classical strategies always-update and never-update, as well as the results obtained by us in our previous work where we applied a DE algorithm to the LAs problem.

If we compare our implementation results with the ones of other authors, which are using dynamic LAs, it is possible to conclude that they are considered very competitive because they are similar, when applied to the same test networks, and we are using static LAs.

5.4. GRASP Based Approach

In this part of our work, a Greedy Randomized Adaptive Search Procedure (GRASP) based algorithm is used to solve the Location Areas problem with the objective of determining the best network partitioning, minimizing the mobility management costs involved. Therefore, we present a new approach to this problem using a grid computing environment with different variants of GRASP. This part of the work was done in cooperation with other members of the research group.

All the experiments carried out to complete this study were executed in a real grid environment provided by a virtual organization of the European project EGEE [101]. These experiments were divided into sequential and parallel executions with the intention of analysing the behavior of the different variants of GRASP when applied to the LA problem. To perform this study, we have selected four distinct test networks, which are exposed in section 5.1.1 *Networks Configuration* and also decided to compare the results obtained by this new approach with those achieved through other algorithms from our

previous work and also by other authors. The experimental results show that this GRASP based approach is very encouraging because, with the grid computing, the execution time is much more reduced and the results obtained are very similar to those of other techniques proposed in the literature.

The study over the GRASP approach was mainly divided into three parts and will be expose in the corresponding following subsections. The first, presented in 5.4.2 *Adjustment of Local Search (LS) Parameter*, has the objective of determining the best value for the local search parameter used in the GRASP algorithm; the second, exposed in section 5.4.3 *Sequential Implementation*, corresponds to the sequential executions with the goal of achieving the GRASP variants that are most adequate to the LA problem; and the third, which is explained in section 5.4.4 *Parallel Implementation*, includes the parallel experiments executed to define the best configuration of the master/slave model implemented.

5.4.1 Grid Computing Environment

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

The access to the grid resources was granted through the EELA (E-science grid facility for Europe and Latin America) [102] virtual organization that includes hundreds of working nodes. All the experiments have been executed with a grid provided by the EGEE (Enabling Grids for E-sciencE) European project [101].

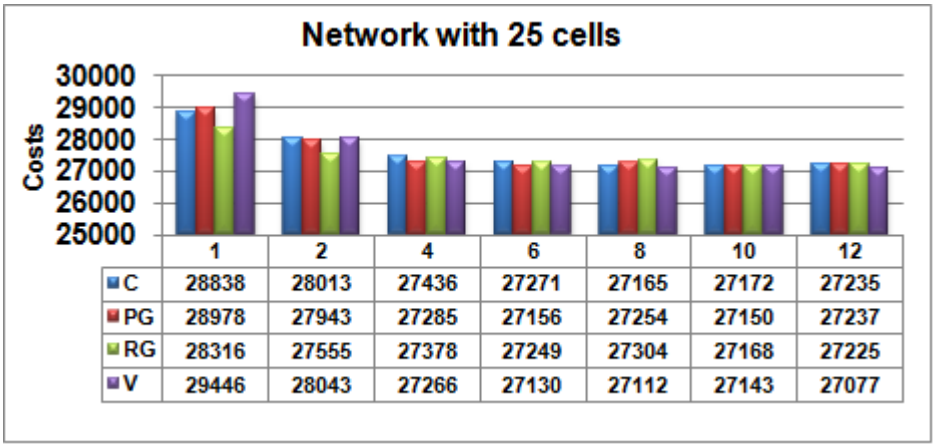
The experiments of this part of the work were performed on the grid in sequential and parallel modes. The sequential experiments were directly managed by the grid (launching many jobs by using simple scripts). For the parallel experiments we have followed a master/slave model, which were developed through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

5.4.2 Adjustment of Local Search (LS) Parameter

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

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

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



	1	2	4	6	8	10	12
C	28838	28013	27436	27271	27165	27172	27235
PG	28978	27943	27285	27156	27254	27150	27237
RG	28316	27555	27378	27249	27304	27168	27225
V	29446	28043	27266	27130	27112	27143	27077

Figure 24 – Adjustment of Local Search with network of 25 cells.

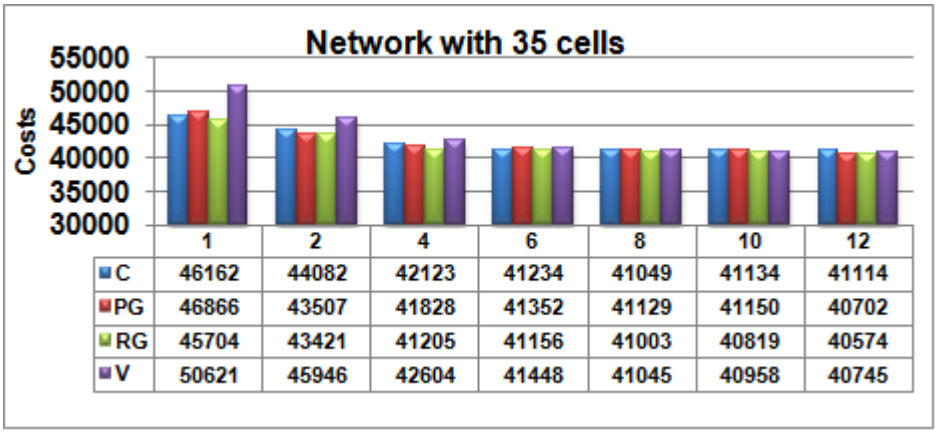
5.4.2.1 Network of 25 (5x5) cells

Using the network of 25 cells and analysing all the cost results (see Figure 24), we concluded that with the value of 12 LS we obtained the best solution, the one with the

lowest costs; but also with the value of 10 LS we obtained a very similar cost and with less computational time. Analysing the results by variant, we noticed that we obtained the lowest cost with the type V.

5.4.2.2 Network of 35 (5x7) cells

Considering the average results obtained with the network of 35 cells, exposed in Figure 25, we also concluded that the ideal number of LS was 12; however the GRASP variant that accomplished the lowest costs was RG.

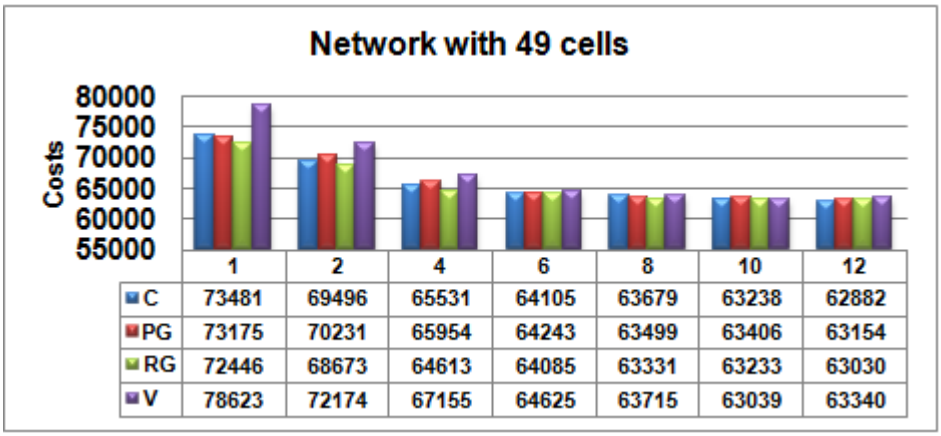


	1	2	4	6	8	10	12
C	46162	44082	42123	41234	41049	41134	41114
PG	46866	43507	41828	41352	41129	41150	40702
RG	45704	43421	41205	41156	41003	40819	40574
V	50621	45946	42604	41448	41045	40958	40745

Figure 25 – Adjustment of Local Search with network of 35 cells.

5.4.2.3 Network of 49 (7x7) cells

With the network of 49 cells and the respective solutions achieved, shown in Figure 26, we noticed that, like in the 5x7 network, the most adequate number of LS was 12 and the variant that reached lower costs was RG. Relatively to the number of LS we also observed that from 8 LS the cost variation of the solutions was very small.

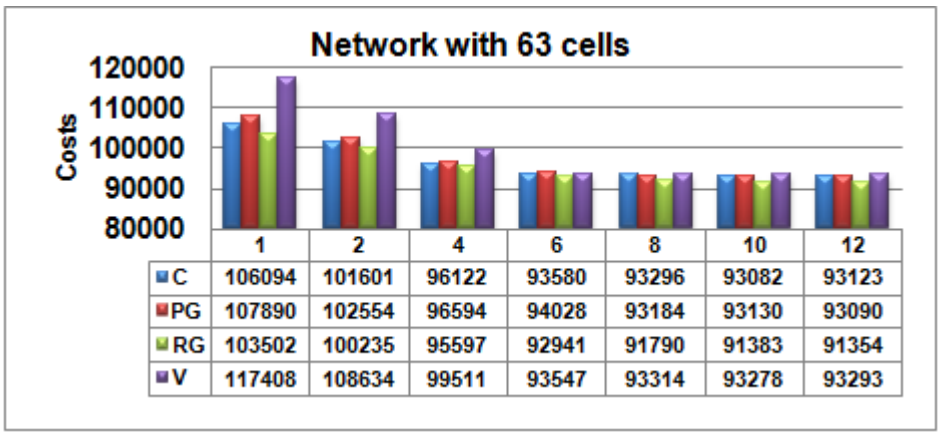


	1	2	4	6	8	10	12
C	73481	69496	65531	64105	63679	63238	62882
PG	73175	70231	65954	64243	63499	63406	63154
RG	72446	68673	64613	64085	63331	63233	63030
V	78623	72174	67155	64625	63715	63039	63340

Figure 26 – Adjustment of Local Search with network of 49 cells.

5.4.2.4 Network of 63 (7x9) cells

Relatively to the test network of 63 cells and the obtained results presented in Figure 27, we concluded again that with the value 12 of LS and the variant RG the best solution was obtained. Anyway, the results with 10 LS were similar.



	1	2	4	6	8	10	12
C	106094	101601	96122	93580	93296	93082	93123
PG	107890	102554	96594	94028	93184	93130	93090
RG	103502	100235	95597	92941	91790	91383	91354
V	117408	108634	99511	93547	93314	93278	93293

Figure 27 – Adjustment of Local Search with network of 63 cells.

5.4.2.5 Comparison of results

Analysing and comparing the results obtained, we concluded that with 12 LS we achieved the lowest costs. Nevertheless, we must consider the execution time over each network, and with 12 LS that time usually surpassed the one assigned, per default, by the grid. Because of that we decided to fix the number of LS to 10 for all the following

experiments, since with this value the execution time was significantly lower and the results were similar. As for the GRASP variants, we observed that the RG variant accomplished the best costs for almost all the networks.

5.4.3 Sequential Implementation

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

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

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

5.4.3.1 Analysis over sequential results

With the sequential experiments we had the objective of determining the 5 best configurations relatively to the GRASP variants, their parameters, and bias function, when applicable. So we decided to elect the best configuration for each GRASP variant and a fifth configuration that would represent the best configuration, in global, for all the networks and variants. Considering this, and knowing that we used these five configurations to proceed with the parallel experiments, presented in subsection 5.4.4 *Parallel Implementation*, the configurations achieved were:

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

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

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

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

e) Conf_4: this is the best global configuration (among the remaining configurations), in this case, associated with the variant RG, with 100 iterations, 10 local searches, k=4 and bias=1.

5.4.4 Parallel Implementation

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

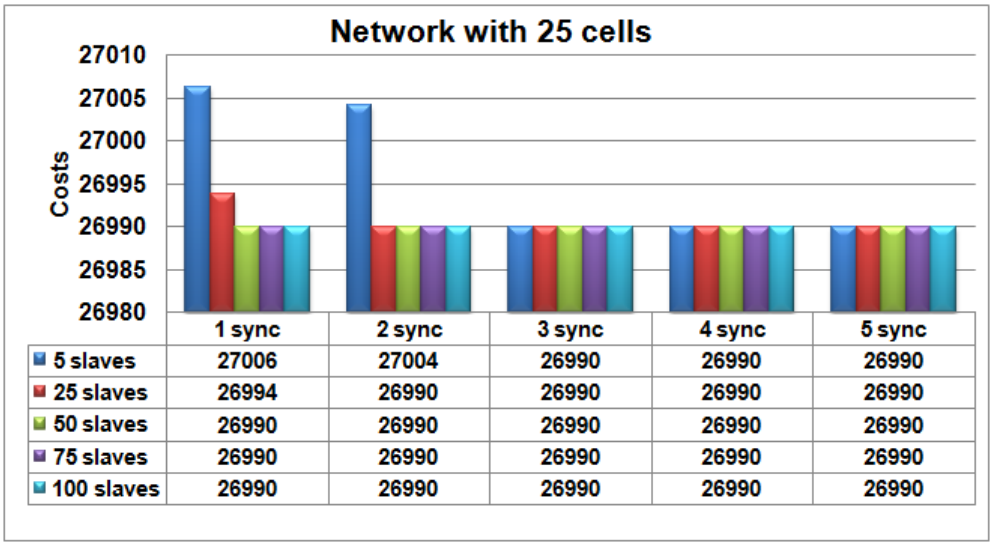
It is also important to highlight that in each experiment we tested the configuration for the number of synchronizations (between the master and slave nodes) with the values 1, 2, 3, 4 and 5.

We must take in consideration that the assignment of the configurations must be proportional to the number of slave nodes that compound the team. For example, if we had 5 slaves, each slave contained a different configuration file, if we had 25 slaves, we would have 5 slaves with equal configurations (but, of course, managing different

solutions), if we had 50 slaves we would have 10 slaves with equal configurations, and so on.

5.4.4.1 Executions with 5x5 network

In Figure 28 we present the average of the results achieved, when we used the network of 25 cells and respectively the configurations with 5, 25, 50, 75 and 100 slaves. Observing the results we concluded that, for this network, we obtained the lowest costs with the majority of configurations and that from 50 slaves, and for all the number of synchronizations we always obtained the best solution.

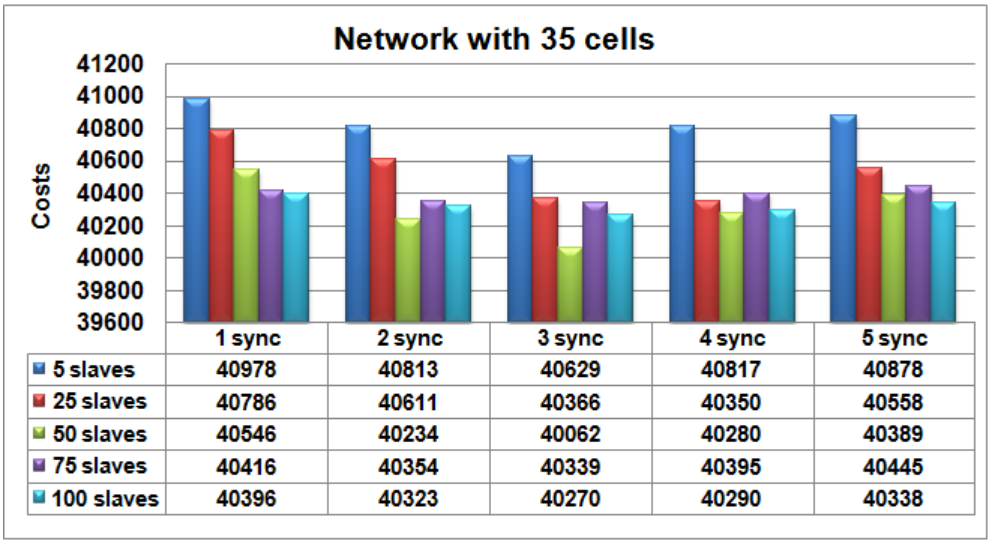


	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	27006	27004	26990	26990	26990
25 slaves	26994	26990	26990	26990	26990
50 slaves	26990	26990	26990	26990	26990
75 slaves	26990	26990	26990	26990	26990
100 slaves	26990	26990	26990	26990	26990

Figure 28 – Parallel executions with 5x5 network.

5.4.4.2 Executions with 5x7 network

Using the network of 35 cells we observed (see Figure 29) that it was with the configuration of 50 slaves and 3 synchronizations, with the master node, that we achieved the solution with the lowest cost.

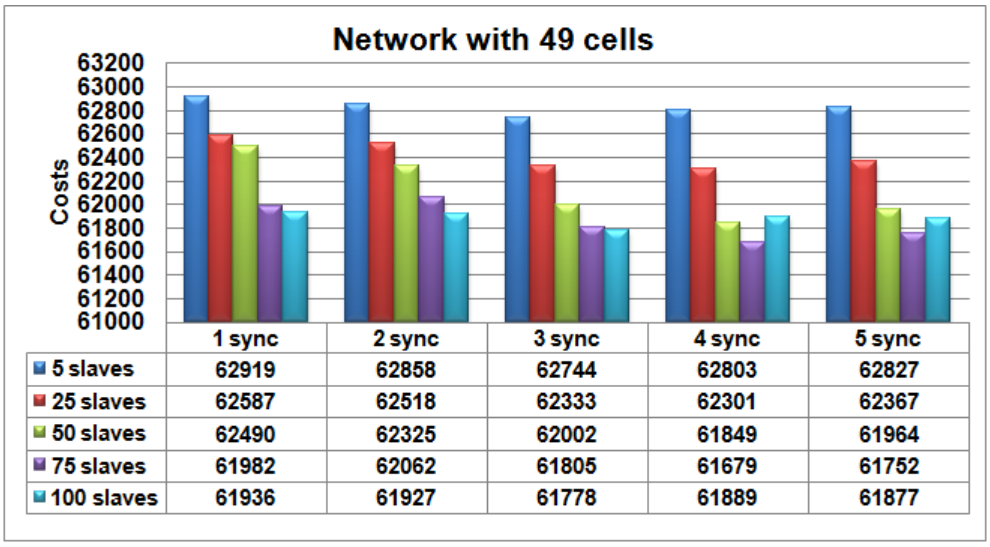


	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	40978	40813	40629	40817	40878
25 slaves	40786	40611	40366	40350	40558
50 slaves	40546	40234	40062	40280	40389
75 slaves	40416	40354	40339	40395	40445
100 slaves	40396	40323	40270	40290	40338

Figure 29 – Parallel executions with 5x7 network.

5.4.4.3 Executions with 7x7 network

The average results obtained with the network of 49 cells are presented in Figure 30. Here we concluded that it was with 75 slaves and 4 synchronizations, with the master, that the best solutions were found.

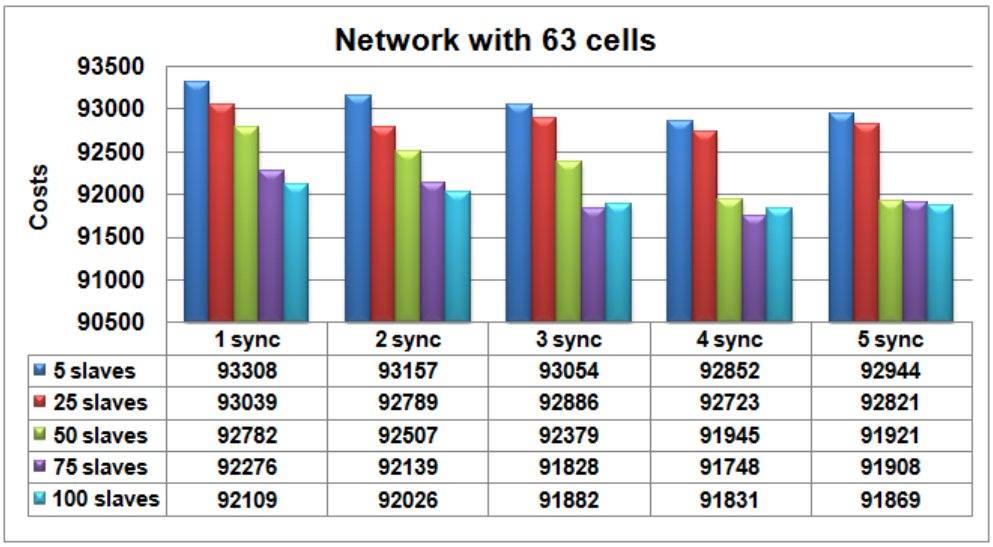


	1 sync	2 sync	3 sync	4 sync	5 sync
5 slaves	62919	62858	62744	62803	62827
25 slaves	62587	62518	62333	62301	62367
50 slaves	62490	62325	62002	61849	61964
75 slaves	61982	62062	61805	61679	61752
100 slaves	61936	61927	61778	61889	61877

Figure 30 – Parallel executions with 7x7 network.

5.4.4.4 Executions with 7x9 network

Considering the average results shown in Figure 31, for the network of 63 cells, once again (like for the 7x7 network) we observed that was with the configuration of 75 slaves and 4 synchronizations, that the best solutions (the lowest costs) were achieved.



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

Figure 31 – Parallel executions with 7x9 network.

5.4.4.5 Analysis and conclusions over the parallel executions

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

5.4.4.6 Participation of the configurations in the parallel team

In this section we present the participation of each GRASP configuration in the parallel team. For this study we used the results obtained with the best configuration of

the parallel team, i.e. 75 slave nodes and 4 synchronizations. Figure 32 shows, respectively for the networks of 25, 35, 49 and 63 cells, the participation percentage of each configuration in the parallel team. Analysing the results we noticed that the participation of the variant RG in the team, and for all the test networks, is clearly superior. This is reasonable, because it is the variant with the best results in all our experiments.

Anyway, we also concluded that the five configurations elected with the sequential executions were very useful in the parallel team. In fact, all of them participate in the results (as shown in Figure 32) and help to improve the final results, as we will see in the next subsection, 5.4.5 *Comparison between Sequential and Parallel Results*.

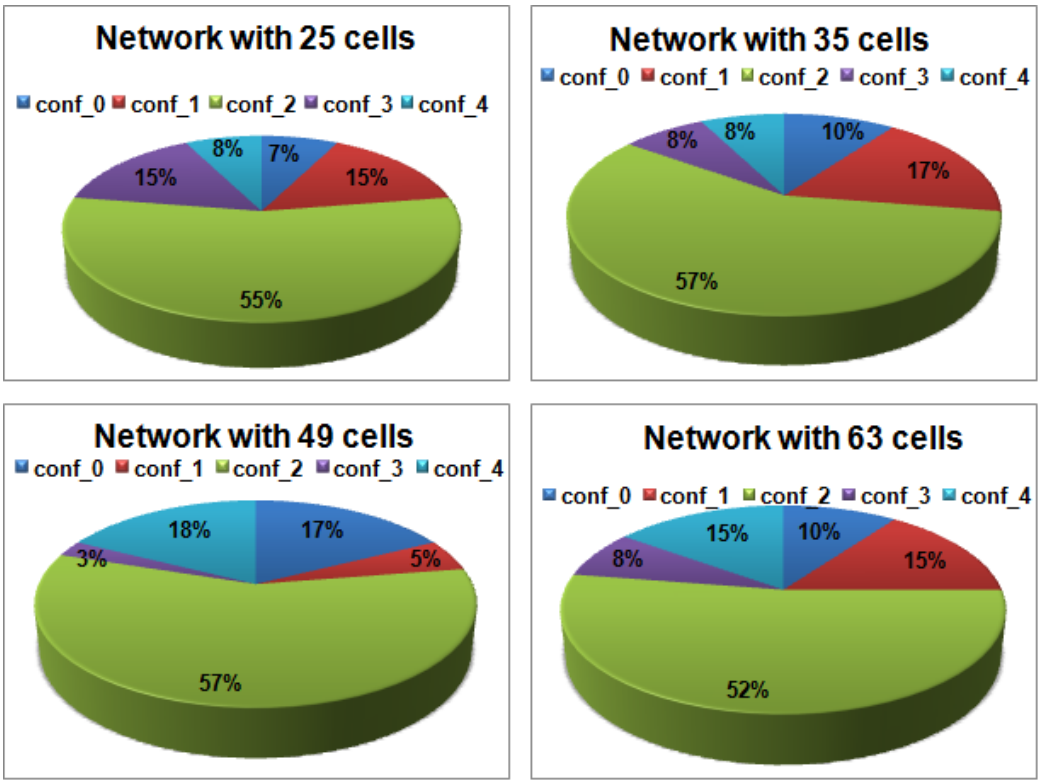


Figure 32 – Participation of GRASP configurations in the parallel team.

5.4.5 Comparison between Sequential and Parallel Results

Finally we decided to compare the results obtained with our parallel implementation of GRASP relatively to its best sequential version. Observing Table 29,

we noticed that the results of the parallel GRASP always surpass those accomplished with sequential GRASP, showing the importance of the parallel team.

Table 29 – Comparison of Sequential and Parallel Results

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

5.4.6 Analysis of the Best Solutions

After we finished our experiments and observing the obtained results, we could take some conclusions about the network configurations achieved. In Figure 33, Figure 34, Figure 35 and Figure 36 we expose, respectively, the best configuration for 5x5, 5x7, 7x7 and 7x9 networks.

Figure 33 – Best LA Configuration of 5x5 Network (Cost of 26990 units).

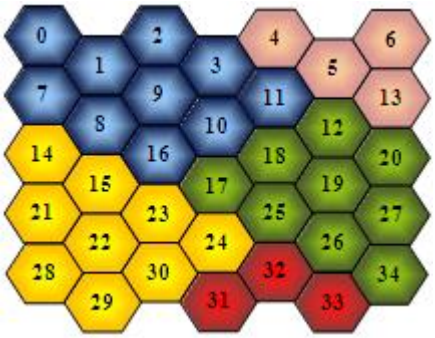


Figure 34 – Best LA Configuration of 5x7 Network (Cost of 39832 units).

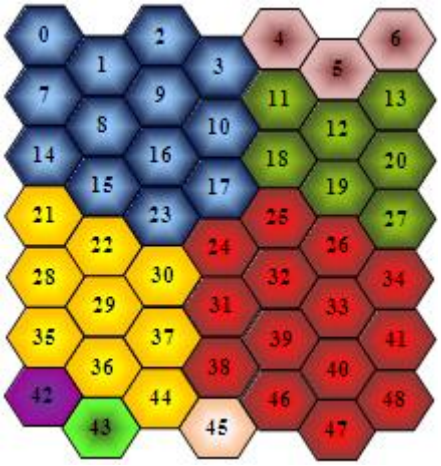


Figure 35 – Best LA Configuration of 7x7 Network (Cost of 60914 units).

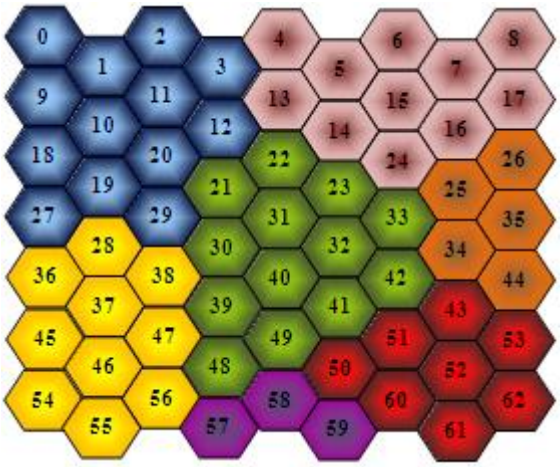


Figure 36 – Best LA Configuration of 7x9 Network (Cost of 89410 units).

Relatively to the shape of LAs the most of them present diverse forms as rectangular or triangular, instead of having a circular shape, as in the actual GSM systems.

With respect to the ideal number of cells, which compound each LA, we also obtained LAs which group different number of cells instead of being all grouped into a predetermined and equal number of cells.

Referring to the number of neighbors for each LA, we noticed that usually each location area has two to four adjacent location areas, although in the GSM networks they

have six. We consider that this happen due to a reduction of the number of location updates and respectively the reduction of the location management costs of the network.

Finally, we also noticed that usually boundary cells of a LA present lower traffic (i.e. less location update and incoming calls) than the other cells in that LA. Considering Table 2 and Figure 33, we observe that cells 17 and 18 have higher traffic while cells 22, 23 and 24 have lower traffic.

5.4.7 Computation Time

After finishing this study, we could notice that the results obtained by using the grid technology are very interesting considering the computation time. With the grid technology it is possible to solve complex problems, which in other way would be compromised and conditioned by the time of execution. In Table 30 we expose the average time of execution, in seconds, to obtain one solution for each network, and also the description of total time consumed by all our experiments.

With these values, we can observe that if we sum all the execution times of all the experiments, we have a total of 992 hours for the sequential experiments and 7389 hours for the parallel experiments, which means a total of 8381 hours for all the executions.

Converting this total into days, we have 349.21 days, which is approximately 1 year. Knowing this, we can understand the real potential of the grid computing, because if we had used a conventional computer to perform the experiments, it had to be working 24 hours per day, during 1 year.

Table 30 – Execution Time of the Experiments Performed

	Network	***Executions***	***Average Time of Resolution (sec.)***	***Total Execution Time***
Sequential	25	1710	28	47880
	35	1710	110	188100
	49	1710	800	1368000
	63	1710	1150	1966500

	Network	***Number of Slave Nodes in the Team***	***Total Execution Time***
Parallel	25	5	6600
		25	33000
		50	66000
		75	99000
		100	132000
	35	5	27500
		25	137500
		50	275000
		75	412500
		100	550000
	49	5	200000
		25	1000000
		50	2000000
		75	3000000
		100	4000000
	63	5	287500
		25	1437500
		50	2875000
		75	4312500
		100	5750000

5.4.8 Summary

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

Relatively to the grid technology, we may say that the results obtained are very satisfactory, because with it we can solve complex problems which, in other way would be conditioned by the time of execution. If we sum all the execution time of all the GRASP experiments we get a total of 8381 hours, or more specifically 349.21 days (about 1 year). This leads us with the real notion of the importance of the grid and parallel computing.

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

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

After analyzing the best solutions we noticed that, on contrary of current GSM systems, which have circular LAs and with equal number of cells, our solutions present diverse shapes like triangular or rectangular ones and are compound by distinct number of cells.

5.5. Results Comparison

In this section we will compare the results obtained when we have applied the three approaches based respectively on SS, DE and GRASP algorithms.

After obtaining the results by using the SS based approach, applied to the LAs problem, we wanted to compare them with our previous work using a DE based approach [103]. We also wanted to compare the results of DE and SS with those obtained in the sequential and parallel implementations of GRASP [105].

With this comparison we noticed that SS performs better than DE and GRASP, because it always obtains equal or better solutions, as shown in Table 31.

Table 31 – Comparison of LM Costs with the LA strategy using SS, DE and GRASP algorithms.

Test Network	SS	DE	Sequential GRASP	Parallel GRASP
5x5	26990	26990	27056	26990
5x7	39832	39859	40611	40062
7x7	60685	61037	62900	61679
7x9	89085	89973	92312	91748

5.6. Comparison of Results with Other Authors

In this section we will analyse the results obtained with studies of other authors, which use different algorithms and compare those result with the ones achieved by our approaches.

After obtaining the results achieved by SS, DE and GRASP based approaches we decided to compare them with the results presented by Taheri and Zomaya in [7, 20, 89, 90] and obtained with very different algorithms as GA (Genetic Algorithm), HNN (Hopfield Neural Network), SA (Simulated Annealing) and GA-HNNx (different combinations of Genetic Algorithm and Hopfield Neural Network [90]).

Observing the results obtained, which are exposed in Table 32, we may conclude that the results achieved with the sequential experiments of GRASP are very close to those accomplished with the algorithms implemented by other authors.

Table 32 – Comparison of Network Costs Achieved by Different Algorithms.

Test Network	SS	DE	GRASP	GA	HNN	SA	GA-HNN1	GA-HNN2	GA-HNN3
5x5	26990	26990	27056	28299	27249	26990	26990	26990	26990
5x7	39832	39859	40611	40085	39832	42750	40117	39832	39832
7x7	60685	61037	62900	61938	63516	60694	62916	62253	60696
7x9	89085	89973	92312	90318	92493	90506	92659	91916	91819

Comparing the results reached by the approach based on DE algorithm with studies of other authors (see Table 32), we can state that they are very similar and in some cases even better.

Furthermore, comparing the SS results with the ones presented by Taheri and Zomaya in [90], our results are always equal or even better, as it is also possible to observe in Table 32. In particular, for the bigger networks (7x7 and 7x9), that is, the more complex ones, SS obtains the best results.

6. Development over Reporting Cells Problem

The Reporting Cell strategy is one of the most common schemes of Location Management problem whose major goal is to minimize the involved costs, which are directly associated to the user movements and their tracing in order to make or receive calls. This will be also our major goal.

Relatively to the work based on the RCs strategy we have developed two main approaches, respectively based on Differential Evolution (DE) and Scatter Search (SS) algorithms. Our main objective is to understand the best configuration of DE and SS when applied to the RCs strategy.

In this chapter, we will detail the decisions taken about the implementation of DE and SS when they are applied to the RC scheme. First we will present the test networks used, then we explain the respective fitness function, defined to evaluate each potential solution and after that, we will expose the main considerations about the initial parameters definition.

For each of the approaches developed we will expose and explain the most important experiments performed and respective results achieved. The experiments executed and the conclusions exposed always take in consideration the goal of tuning the specific parameters of each algorithm and reaching its most adequate configuration.

To perform the experiments used to study the RCs strategy we have used twelve distinct test networks, which will be presented in section 6.1.1 *Networks Configuration*, based on realistic data and patterns.

After that we also want to compare the results obtained by both strategies and also with those achieved by other authors, as Alba et al. in [10], which expose two approaches based on Hopfield Neural Network with Ball Dropping (HNN-BD) and Geometric Particle Swarm Optimization (GPSO).

6.1. Implementation Considerations

In the following section we will expose the experiments performed with the goal of solving the reporting cells problems, by implementing two approaches, respectively based on DE and SS algorithms. However, first we need to explain and present some common implementation considerations. In the following subsection we will expose the configuration of the networks used by both approaches. After that, and also because it was common for both approaches, we will explain the fitness function applied to evaluate each potential solution generated.

6.1.1 Networks Configuration

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 [10] 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 [21] as benchmark.

Considering the objective of testing our DE and SS based approaches, and comparing the results accomplished, we decided to use this set of twelve test networks.

In Table 33 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. The information of the other 11 networks is available in [21].

Table 33 – Test Network 1 – NLU and NP Values

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

6.1.2 Fitness Function

In the study of the reporting cells problem, like in the study of the location area problem, the fitness function is used for measuring the total location management cost of each potential solution, which is defined according to equation (6), presented in section 2.4.2. This fitness function will be used to calculate the cost value of each network configuration (potential solution generated) by means of reporting cells and non-reporting cells.

6.2. DE Based Approach

With the goal of solving the LM problem through the reporting cells strategy, we started designing a new approach based on the DE algorithm. This decision was taken considering the good results achieved with the DE based algorithm, when applied to the LA strategy and also because, for the best of our knowledge, it was the first that this algorithm was applied to the RC strategy. The results obtained are very promising when compared with those achieved by other authors' approaches.

We will start this section explaining the implementation details specific for the DE based implementation. The specific implementation details correspond to the initial definition of DE parameters. Following, we expose and explain the major results of all the experiments which we have performed over the twelve test networks. These experiments were performed with the aim of adjusting each of DE parameters and defining the best DE configuration when applied to the RCs problem. Finally, we analyse and compare the results accomplished, trying to obtain the best conclusions over the DE based approach when applied to the RC strategy.

6.2.1 Implementation Details

In this section we are committed to explain the specific considerations that must be taken before the implementation of the DE based approach and respective execution of experiments.

Considering that the network configuration and the fitness function are two common processes for both approaches, which have already been presented, in this section we only will expose the main decisions and adjustments of the DE algorithm implementation, which consist in the respective definition of the initial values of DE parameters.

6.2.1.1 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.

In the RCs problem 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.

The initial population has been defined by setting the type of each cell as reporting cell, or non-reporting cell, with a probability of fifty percent.

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. Relatively to the terminal condition, we have set 1000 generations for the algorithm execution.

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

6.2.2 Experiments and Analysis

In this section we explain the four distinct experiments executed, respectively for adjusting the population size *NI*, crossover value *Cr*, mutation factor *F* and the DE scheme, applied to each test network with the objective of studying in more detail the best configuration of DE, applied to the reporting cells problem. For each experiment, and for

all combination of parameters, 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.

As said, with the aim of assuring the statistical relevance of the results obtained, we have executed 30 independent runs, for each experiment and combination of parameters. In addition, we have also performed a statistical analysis, over all the experiments executed, by using the ANOVA test. Like for the LA strategy, we have considered a confidence level of 95% (i.e. significance level of 5% or p-value under 0.05).

Following, and considering the results achieved with our experiments, we analyze and compare the results obtained with the goal of presenting the respective configuration for best solutions.

After that we also want to compare the results obtained by DE based approach with those achieved by other authors, as Alba et al. in [10], which expose two approaches based on Hopfield Neural Network with Ball Dropping (HNN-BD) and Geometric Particle Swarm Optimization (GPSO).

6.2.2.1 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 6.2.1.1 *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 [98, 99].

With this experiment we have concluded that, increasing *NI* value, it is possible to observe a positive evolution of the results (see fitness results in Table 34), 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.

Table 34 – Experiment 1: Determining the Best NI

Test Network	NI – Fitness Evaluation									
N. (Dim)	**10**	**25**	**50**	**75**	**100**	**125**	**150**	**175**	**200**	**225**
1 (4x4)	99137	100881	98535	98535	98535	98535	98535	98535	98535	98535
2 (4x4)	101250	98879	97156	97156	97156	97156	97156	97156	97156	97156
3 (4x4)	98106	101403	95038	95038	95038	95038	95038	95038	95038	95038
4 (6x6)	195283	185092	176800	173701	173701	173701	173701	173701	173701	173701
5 (6x6)	205859	192426	185937	182331	182331	182331	182331	182331	185059	182331
6 (6x6)	193540	186423	175321	174519	174519	174519	174519	174519	174519	174519
7 (8x8)	338196	321575	315097	310888	308853	309342	308401	308401	308991	308401
8 (8x8)	319912	304750	294548	287149	289051	287149	287149	287149	289935	287149
9 (8x8)	292467	276299	270171	265272	264204	264204	264204	264316	264204	264204
10 (10x10)	425866	405127	394168	388206	386775	387551	386695	386474	387543	386893
11 (10x10)	390793	377363	361299	360210	361581	359224	358778	359224	359697	358944
12 (10x10)	401704	385022	380846	375233	376631	374711	375001	375722	373733	374220

6.2.2.2 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.

Table 35 – Experiment 2: Determining the Best Cr

Test Network	Cr – Fitness Evaluation						
N. (Dim)	**0.1**	**0.15**	**0.20**	**0.25**	**0.50**	**0.75**	**0.90**
1 (4x4)	98535	98535	98535	98535	98535	98535	98535
2 (4x4)	97156	97156	97156	97156	97156	97156	97258
3 (4x4)	95038	95038	95038	95038	95038	95038	98216
4 (6x6)	173701	173701	173701	173701	173701	177647	177889
5 (6x6)	182331	182331	187990	183264	184679	183991	185966
6 (6x6)	174519	174519	174519	175321	175182	175321	178255
7 (8x8)	308401	308401	308401	311646	313378	313607	319069
8 (8x8)	287149	287149	289573	289051	293248	302812	309609
9 (8x8)	264204	264204	265452	264786	272249	266876	275489
10 (10x10)	387318	386681	388357	386959	393510	393492	420650
11 (10x10)	360262	358669	360072	360128	360596	367508	374405
12 (10x10)	373695	374966	374554	374921	377190	383782	391001

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 35, we could conclude that *Cr*=0.15 is the one that performs better.

6.2.2.3 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 experiment 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 36) we conclude that *F*=0.5 is the one that permits to obtain the best results.

Table 36 – Experiment 3: Determining the Best F

Test Network	**F – Fitness Evaluation**				
N. (Dim)	**0.1**	**0.25**	**0.50**	**0.75**	**0.90**
1 (4x4)	98535	98535	98535	98535	98727
2 (4x4)	97156	97156	97156	97156	97156
3 (4x4)	95038	95038	95038	95038	95038
4 (6x6)	174112	173701	173701	173701	176530
5 (6x6)	182331	182331	182331	182331	182331
6 (6x6)	174519	175321	174519	174519	174519
7 (8x8)	310162	310426	308401	311492	308401
8 (8x8)	293093	304911	287149	292913	295557
9 (8x8)	265494	264643	264204	268312	265750
10 (10x10)	388849	389438	386681	389125	387533
11 (10x10)	359221	360072	358669	358167	361441
12 (10x10)	373298	375087	374966	371829	375232

6.2.2.4 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 3.1.5. *DE Schemes*.

Once finished all the executions, and observing the respective results shown in Table 37, 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.

Table 37 – Experiment 4: Determining the Best DE Scheme

Test Network	**DE Scheme – Fitness Evaluation**									
	Exponential Crossover					**Binomial Crossover**				
N. (Dim)	**Best1**	**Rand1**	**RTB1**	**Best2**	**Rand2**	**Best1**	**Rand1**	**RTB1**	**Best2**	**Rand2**
1 (4x4)	98535	98535	98535	98535	99008	98535	98535	98535	98727	98535
2 (4x4)	97156	97156	97156	97156	97156	97156	97156	97156	97156	97156
3 (4x4)	95038	95038	95038	95038	95038	95038	95038	95038	95038	95038
4 (6x6)	173701	173701	173701	176530	178038	176041	173701	173701	173701	173701
5 (6x6)	182331	182331	187801	190779	191279	182331	182331	182331	182331	182331
6 (6x6)	174519	175182	183992	177276	177892	181850	174519	174519	174519	174519
7 (8x8)	322973	319772	328327	323391	332472	320236	308401	308401	309855	308730
8 (8x8)	304214	307139	313010	310708	316849	305236	287149	287149	287149	287149
9 (8x8)	277408	279177	290646	291684	289936	269984	264204	264316	265164	264353
10 (10x10)	420701	423017	420452	421353	425228	394176	386681	386951	393471	386695
11 (10x10)	385950	380824	384661	387273	380241	366156	358167	359486	367202	359517
12 (10x10)	394636	388468	395290	395404	394767	379227	371829	376015	379544	376165

6.2.2.5 Statistical Analysis Using ANOVA

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 38 we show the results obtained with this test, using the distinct values for each DE parameter.

Table 38 – 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

Once again we can observe that the fitness differences have been found significant in most of the cases.

6.2.2.6 Analysis and Comparison of Results

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.

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 [10] 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 386951 and in [10] the one obtained by the HNN+BD is 386351; and for the test-network-12 our fitness value is 371829 and in [10] the value obtained by HNN+BD and GPSO is 370868. Relatively to the average values it is possible to say that they are very similar.

In Figure 37, Figure 38 and Figure 39 the configuration for each test-network solution is shown and it is possible to observe that most of them split each one in subnetworks.

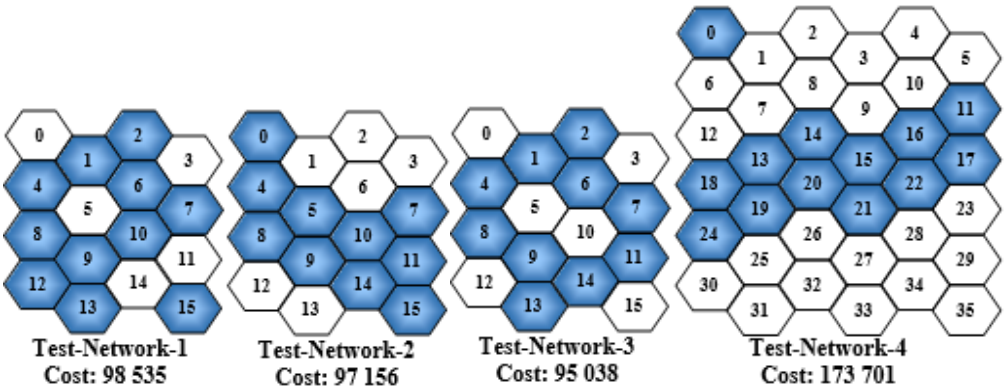


Figure 37 – Test Network Solutions with Reporting Cells Configuration and Applying the DE Algorithm (Part 1)

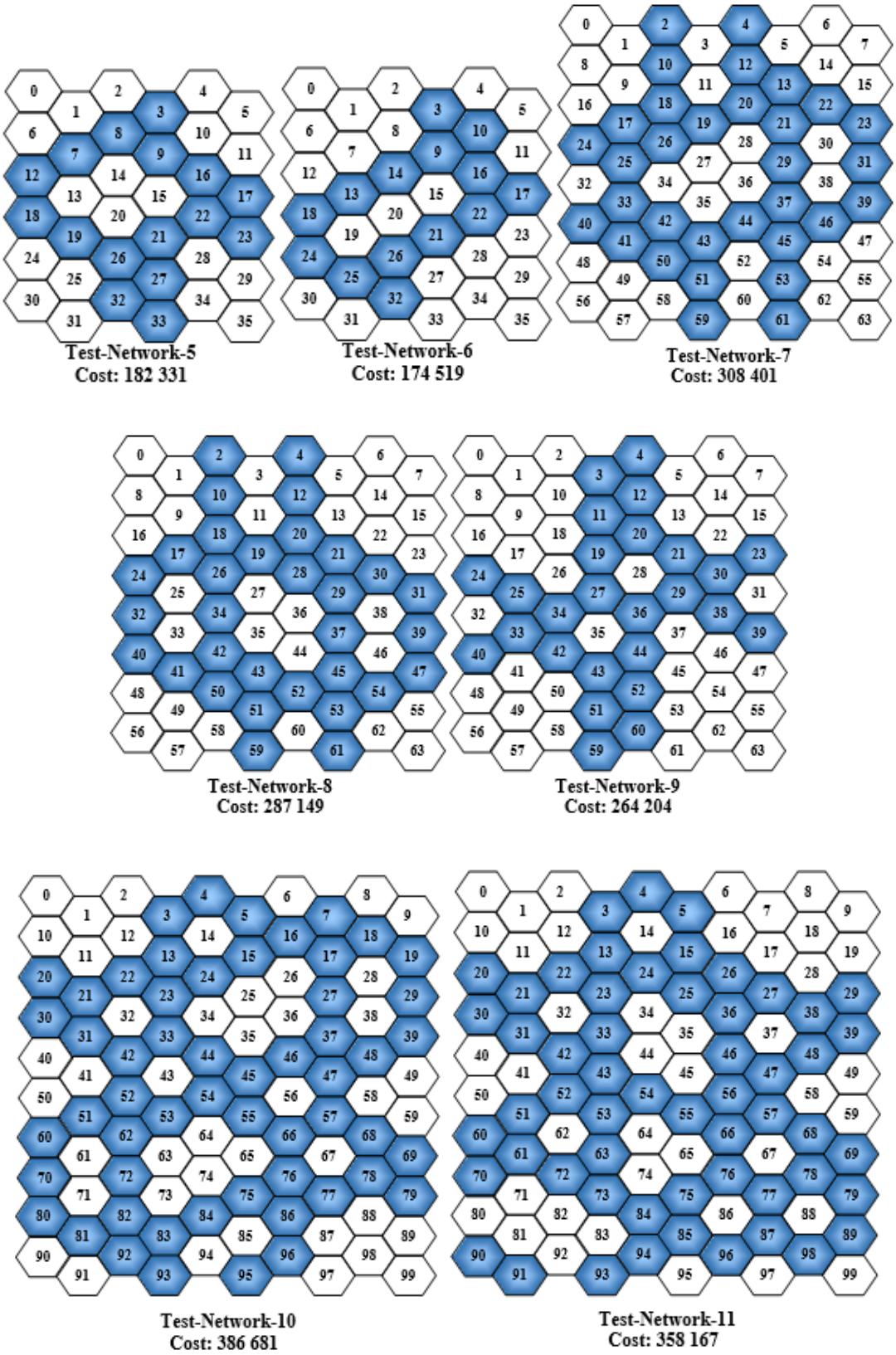


Figure 38 – Test Network Solutions with Reporting Cells Configuration and Applying the DE Algorithm (Part 2)

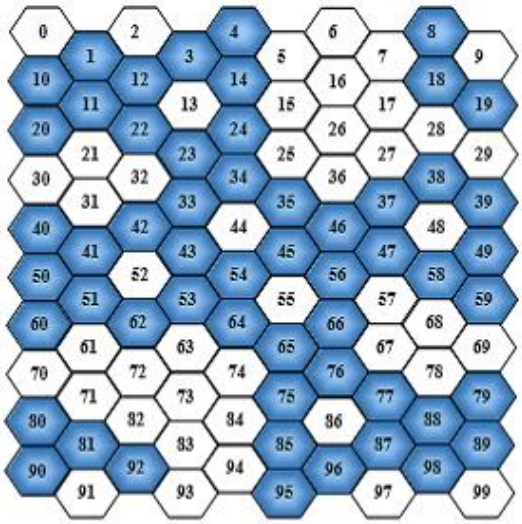


Figure 39 – Test Network Solutions with Reporting Cells Configuration and Applying the DE Algorithm (Part 3)

The convergence curves of DE relatively to each test-network solution are presented in Figure 40. In all the graphics we can observe the best solution (blue line), the worst (green line) and the average solution (red line). In all the cases we can see a good convergence.

Table 39 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.

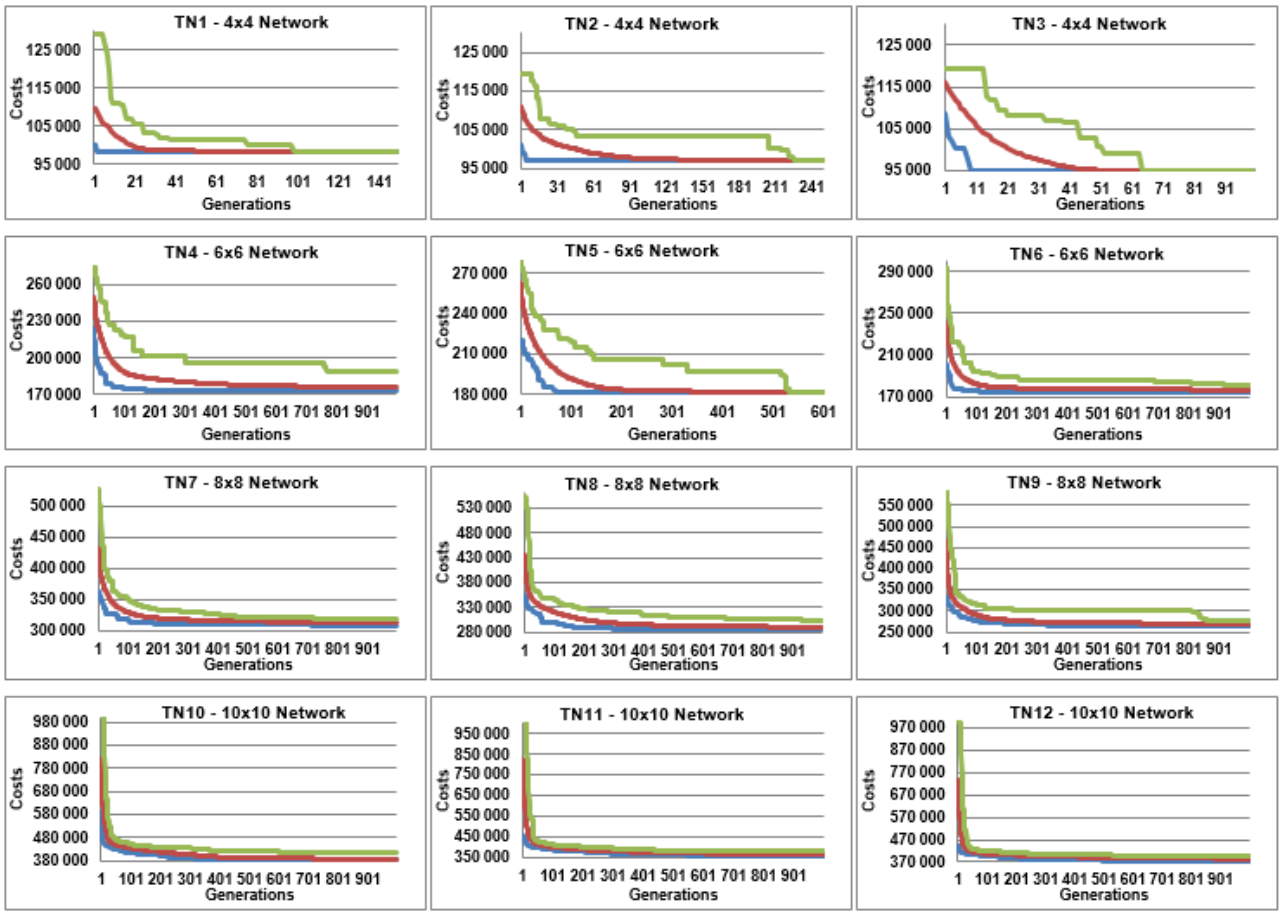


Figure 40 – DE convergence curves: LM cost values (Y axis) vs. generations (X axis). Each graphic shows the evolution of the best (blue line), the worst (green line) and the average solution (red line).

Table 39 – CPU time(s) of DE over RC experiments.

Network	Average	Best
TN1 – 4x4	19	16
TN2 – 4x4	21	16
TN3 – 4x4	20	15
TN4 – 6x6	110	91
TN5 – 6x6	68	58
TN6 – 6x6	77	71
TN7 – 8x8	140	133
TN8 – 8x8	139	125
TN9– 8x8	163	148
TN10 – 10x10	187	169
TN11 – 10x10	203	192
TN12 – 10x10	232	211

6.2.3 Summary

In this section we have discussed the use of differential evolution algorithm applied to the reporting cells problem, which, for the best of our knowledge, is the first time that DE is employed for this task.

We have detailed the implementation considerations made to develop the DE based approach. After that we exposed the experiments performed, studying in detail the best configuration of DE including parameters and scheme. After more than 10000 runs, they are *NI*=175, *Cr*=0.15, *F*=0.5 and *DE/rand/1/bin* as the best DE scheme.

Analyzing the experimental results achieved we could conclude that with this approach it is possible to obtain promising results, considering that those results include the same minimum fitness values as the ones obtained in [10] 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. Observing the average values it was possible to say that they are also very similar.

After we have observed the configuration for each test-network solution, we noticed that the most of the networks are partitioned in subnetworks.

Considering the convergence curves relatively to each test network and after analysing the graphics that include the best and average solutions, we noticed that in all cases we observed a good convergence.

6.3. SS Based Approach

Finding an optimal set of reporting cells is an NP-complete problem. So, after the implementation of the approach based on the DE algorithm, we decided to study and implement a new Scatter Search based approach applied to the reporting cells planning as a cost optimization problem. The major goal of the RC problem is to optimize the configuration planning of mobile networks, by means of reporting cells and non-reporting cells, in a process of minimizing the involved costs.

Reflecting the good results achieved when we have applied the SS to the LA problem we considered that would be a good direction of our investigation applying it to the RC problem. After finishing this work we intended to compare the results accomplished with those obtained with the approach based on the DE algorithm and also with the results achieved by other authors.

The results achieved show that our approach can be successfully applied to the RC problem, because, comparing with the results accomplished by other authors that use Geometric Particle Swarm Optimization (GPSO) and Hopfield Neural Network with Ball Dropping (HNN-BD) we obtain equal or even better fitness costs.

Our first step in this section will be the specification of the implementation details that are specific for the implementation based on the SS algorithm. This specification is characterised by the initial definition of the SS parameters. After that, we detail and present the main experiments that we have accomplished and the respective results achieved over the twelve test networks. Considering the aim of determining the most adequate configuration of SS, when applied to the RCs problem, each experiment was executed to adjust the best value of each one of those SS parameters.

Finally, we will analyse the results accomplished with the goal of understanding the viability of the SS based approach when applied to the RC strategy. To complement and reinforce the conclusions taken we need to compare the results achieved with those obtained by the DE based approach and also by other authors.

6.3.1 Implementation Details

In this section we expose the decisions taken about the implementation details that are specific of SS when applied to the RC scheme and which must be taken before the implementation of the SS based approach.

Like we have explained for the DE based approach, the network configuration and the fitness function definition are common processes for both approaches and have been presented previously in 6.1 *Implementation Considerations*.

Considering the common processes, the specific implementation details of the SS algorithm will only correspond to the major decisions and adjustments for the definition of initial values of SS parameters.

6.3.1.1 Parameters Definition

As we already mentioned earlier, the initial definition of parameters is one of the most important steps of the approach definition because it represents the basis for the algorithm evolution. First it is defined the initial population of candidate solutions, where each solution corresponds to an individual.

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

Considering the outline of the SS algorithm, exposed in Algorithm 2, we implemented the *diversification generation* method, which is applied to the generation of the initial population, considering that will be set a RC or nRC to each cell, with a probability of 50%. Relatively to the *improvement* method we decided to apply a local search, characterized by switching a RC with one of their neighbors that are nRC. Regarding the *subset generation* method we decided to use subsets of size 2. Respectively to the *combination* method we implemented a crossover that may be applied from one to four crossover points, considering a predetermined probability.

Concluding, our implementation of SS, applied to the RC problem, considers four core parameters: initial population size *PSize*; reference set size *RSSize*; probability of combination (crossover) *Cr*; and also, the number of iterations of local search *nLS*. The reference set size is also divided between the set size of the quality solutions (*nQrs*) and the diversity solutions (*nDrs*).

To start the initial experiments, we have set the following initial values of parameters: *PSize*=100; *RSSize*=10 (respectively divided between *nQrs*=5 and *nDrs*=5); *Cr*=0.2; and *nLS*=1, taking into consideration the suggestion of several authors [15, 16].

During the course of the different experiments, the parameters values have been tuned with the specific objective of obtaining the best results.

6.3.2 Experiments and Analysis

Considering that the scatter search had given good results when applied to the LA problem we also decided to apply it to the RC problem.

With the objective of understanding the behaviour of SS when applied to the RC problem, we have performed, and will explain in this section, five main experiments, over the twelve test networks, in order to adjust the parameters of the SS algorithm. This means that each of these experiments corresponds respectively to the tuning of the four core parameters: initial population size *PSize*, reference set size *RSSize*, probability of combination (crossover) *Cr* and the number of iterations of local search *nLS*. Furthermore, after determining the best *RSSize*, the following experiment will correspond to its respective division between the size of quality solutions *nQrs* and the size of the most diversity solutions *nDrs*, this is, the two parameters which compound the *RSSize*.

For each experiment we set the values of each parameter as the initial ones, referred in section 6.3.1.1 *Parameters Definition* (while the respective experiment was not performed) or as the one achieved and defined in its respective experiment.

Similar to the DE based approach, and with the objective of assuring the statistical relevance of the results obtained, we have performed 30 independent runs, for each experiment, including each parameters' configuration. Also, in addition, we have performed a statistical analysis, over all the experiments executed, by using the ANOVA test. Analogous to the LA strategy, we have considered a confidence level of 95% (i.e. significance level of 5% or p-value under 0.05).

After that, considering the results accomplished by using our SS based approach, we analyse and discuss the results achieved, determining the best SS configuration and present the network configuration for the best solutions.

Finally we want to compare the results obtained with our previous work, based on Differential Evolution (DE) algorithm, and also with those accomplished by Alba et al.

in [10], which use two approaches based on Hopfield Neural Network with Ball Dropping (HNN-BD) and Geometric Particle Swarm Optimization (GPSO).

Table 40 – Results of Experiment 1 to Determine the Best Population Size – Parameter *PSize* (A – Average Cost; B – Best Cost).

	Fitness Evaluation								
PSize	**10**	**25**	**50**	**75**	**100**	**125**	**150**	**175**	**200**
TN1 – B	98535	98535	98535	98535	98535	98535	98535	98535	98535
TN1 – A	98637.4	98605.4	98611.8	98579.8	98599	98592.6	98605.4	98611.8	98592.6
TN2 – B	97156	97156	97156	97156	97156	97156	97156	97156	97156
TN2 – A	97326.3	97251.13	97275.27	97251.53	97237.6	97225.57	97207.07	97230.93	97227.53
TN3 – B	95038	95038	95038	95038	95038	95038	95038	95038	95038
TN3 – A	95140.27	95150.9	95038	95140.27	95038	95038	95140.27	95038	95038
TN4 – B	175241	174190	174311	173753	174190	174112	173753	173701	173753
TN4 – A	178790.57	176928.67	177297.63	178082.87	177891.37	176544.5	177057.03	178250.87	176519.7
TN5 – B	182331	182331	182331	182331	182331	182331	182331	182331	182331
TN5 – A	185486.6	184408.7	184476.03	185273.4	184571.6	184962.8	184421.8	184572.43	184251.1
TN6 – B	174519	174519	174519	174519	174519	174519	174519	174519	174519
TN6 – A	177144.07	177166.5	177425.8	176673.93	176663.1	176906.93	177150.47	176279.13	176913.47
TN7 – B	311687	309557	308899	311621	311515	309566	309566	310252	309566
TN7 – A	315602.57	313969.73	314039.6	314325.93	313894.3	313360.1	314039.37	313580.4	313827.67
TN8 – B	288313	289935	287149	287149	287149	287149	287149	287149	287149
TN8 – A	296981.1	297828.13	295751.4	295483.07	294816	295963.6	295533.7	296250	294877.13
TN9 – B	265000	265291	264563	264466	265120	264353	264716	264716	264409
TN9 – A	272963.2	273316.07	270484.8	270561.2	271217.77	269335.6	269832.9	271397.3	270087.1
TN10 – B	387668	389225	389276	389118	388129	388049	389043	389368	388121
TN10 – A	394202.27	395621.6	395321.53	395534.7	395079.07	394606.67	394470.8	393892.2	394286
TN11 – B	360243	360348	360247	359814	359795	359936	359891	359420	360200
TN11 – A	365480	364756.3	364694.6	364008.6	362992.47	364356.63	365071.07	364224.5	364036.63
TN12 – B	376521	376383	374720	371521	375285	374289	371905	372624	377257
TN12 – A	380686.8	380301.57	380213.97	380202.8	379672.27	379601.7	380603.9	380117.2	379896.23

6.3.2.1 Determining the Population Size

The first experiment includes the first step of the SS algorithm, which is the generation of an initial population with a predetermined number of distinct solutions (*PSize* parameter). Due to that, the goal of this experiment is defining the ideal number of solutions, which should compound the initial population. Using the twelve test networks and considering the initial values of each one of the other parameters, we have tested *PSize* with the following number of solutions: 10, 25, 50, 75, 100, 125, 150, 175 and 200. Analysing the results, exposed in Table 40, we noticed that the lower LM costs, for the average and best fitness, were achieved with *PSize*=175, setting it for the following experiments.

Table 41 –Results of the Experiment 2 to Determine the Best Reference Set Size – Parameter *RSSize* (A – Average cost; B – Best Cost).

	Fitness Evaluation									
RSSize	**2**	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
TN1 – B	98535	98535	98535	98535	98535	98535	98535	98535	98535	98535
TN1 – A	99813.27	98786.97	98618.2	98567	98573.4	98579.8	98554.2	98586.2	98541.4	98535
TN2 – B	97156	97156	97156	97156	97156	97156	97156	97156	97156	97156
TN2 – A	98789.17	97754.17	97543.77	97303.13	97217.27	97193.47	97196.87	97183.27	97156	97166.2
TN3 – B	95038	95038	95038	95038	95038	95038	95038	95038	95038	95038
TN3 – A	98064.47	96097.57	95300.27	95038	95038	95038	95038	95038	95038	95038
TN4 – B	183432	175241	175241	174190	173753	174311	173753	173701	173701	173701
TN4 – A	191388.67	183766.2	178819.2	176325.77	177120	176143.93	175467.5	175052.73	175093.2	174827.07
TN5 – B	187094	183264	182331	182331	182331	182331	182331	182331	182331	182331
TN5 – A	196779.1	188745.83	185869	184663.97	184568.8	183926.57	183349.03	182973.03	182567.13	182641.63
TN6 – B	177618	176265	174519	174519	174519	174519	174519	174519	174519	174519
TN6 – A	186901.97	179982.37	177218.57	176785.47	176182.77	175942.53	175861.9	175619.67	175180.77	174914.77
TN7 – B	318307	313845	310246	311105	309126	308853	309557	308972	308702	307695
TN7 – A	327857.9	318913.73	315954.13	313455.23	313699.37	313142.93	313248.77	312221.67	312152.8	311714.9
TN8 – B	290975	294649	289041	289935	287149	287149	287149	287149	287149	287149
TN8 – A	310557.7	305220.87	297995.83	296857.03	294393.37	294438.5	292068	291865.23	290389.4	290390.6
TN9 – B	274492	267679	264563	266412	265258	264316	264204	264204	264204	264204
TN9 – A	291753.53	277611.33	271994.73	272535.6	271017.4	268968.13	267376.17	266318.4	266305.37	265392
TN10 – B	395966	392875	389516	388271	387154	387273	387124	386474	386782	386782
TN10 – A	415672.07	402481.2	397391.13	393453.23	391968.53	392501.63	391717.77	391977.2	391253.37	390292.87
TN11 – B	363911	362300	360277	359194	359886	359194	359778	358915	358795	358521
TN11 – A	379343.47	368615.03	365595.6	363824.63	363716.03	363075.5	363157.6	362114.27	361713.1	360995.37
TN12 – B	381268	380348	378173	375871	375497	371366	373595	371156	371943	370868
TN12 – A	393521.83	383851	382583.4	380045.2	379565.57	378376.93	378436.17	377623.47	377730.63	376626.53

6.3.2.2 Determining the *RefSet* Size

The second experiment was performed with the objective of determining the size of the reference set (*RSSize* parameter), which will include the best and the most diverse solutions from these ones. Setting *PSize*=175, we executed the following configurations

for *RSSize*: 2, 4, 8, 10, 12, 14, 16, 18 and 20. Observing the statistical results, shown in Table 41, we verified that they were improving with the increase of the *RSSize*, and because of that, we decided to elect *RSSize*=20 to proceed for the next experiments. Furthermore, we also tested and noticed that improving the *RSSize*, with higher values than 20, was not significant.

Table 42 –Results of the Experiment 3 to Determine the Best Division of the RSSize in Quality (*nQrs*) and most Diverse (*nDrs*) Solutions (A – Average cost; B – Best Cost).

	Fitness Evaluation										
nQrs/nDrs	**10/10**	**11/9**	**12/8**	**13/7**	**14/6**	**15/5**	**16/4**	**17/3**	**18/2**	**19/1**	**20/0**
TN1 – B	98535	98535	98535	98535	98535	98535	98535	98535	98535	98535	98535
TN1 – A	98535	98547.8	98541.4	98535	98535	98535	98535	98535	98535	98535	98547.8
TN2 – B	97156	97156	97156	97156	97156	97156	97156	97156	97156	97156	97156
TN2 – A	97169.6	97156	97166.2	97156	97156	97162.8	97156	97156	97159.4	97156	97166.2
TN3 – B	95038	95038	95038	95038	95038	95038	95038	95038	95038	95038	95038
TN3 – A	95038	95038	95038	95038	95038	95038	95038	95038	95038	95038	95038
TN4 – B	173701	173753	173701	173701	173701	173701	173701	173701	173701	173701	173701
TN4 – A	174954	175084.77	175149.57	174851.33	174986.83	174727.83	175117.2	174983.27	174862	174848.43	174884.83
TN5 – B	182331	182331	182331	182331	182331	182331	182331	182331	182331	182331	182331
TN5 – A	182461.9	182455.63	182542.9	182567.5	182443.1	182374.63	182399.47	182581.33	182387.17	182443.1	182480.47
TN6 – B	174519	174519	174519	174519	174519	174519	174519	174519	174519	174519	174519
TN6 – A	175151.23	175007.17	175004.67	174662.33	174695.17	174717.9	174696.8	174969.7	174861.6	174695.17	174628.87
TN7 – B	308401	307695	308401	307695	307695	307695	307695	307695	308853	307695	308401
TN7 – A	311592.3	311860.53	311623.43	311399.2	311318.67	311803.2	311377.2	311751.33	312069	311530.37	311778.5
TN8 – B	287149	287149	287149	287149	287149	287149	287149	287149	287149	287149	287149
TN8 – A	290483.27	289551.87	290125.23	289512.67	288398.77	291084.83	291533.63	289347	291536.73	290282.8	289261.73
TN9 – B	264204	264204	264204	264204	264204	264204	264204	264316	264204	264204	264204
TN9 – A	265047.73	265411.7	265645.9	266062.07	265274.8	265753.73	266046.07	265450.1	266262.07	265976.07	264925.63
TN10 – B	386659	386351	386782	386662	385927	387146	386357	386659	386351	386474	385927
TN10 – A	389852.97	391308.4	389890.2	390415.57	390720.2	390319.47	390199.83	389903	389657.63	389917.53	389235.43
TN11 – B	358167	358635	358431	359609	358259	358456	358795	358661	358772	358489	358870
TN11 – A	362226.5	361573.7	361486.2	361934.27	361409	361795.6	361901.37	361109.97	361476.77	361156.2	361005.57
TN12 – B	371239	371341	371337	371534	370966	371735	370966	370966	371534	371156	372152
TN12 – A	377520.8	376977.3	377194.47	376207.13	376990.4	376224.77	376772.03	376455.23	376672.7	377230.77	376655.47

6.3.2.3 Dividing the *RefSet* between *nQrs* and *nDrs*

After determining the most adequate value for *RSSize*, it was necessary to determine its best division between quality solutions (*nQrs*) and most diverse solutions (*nDrs*). Considering *PSize*=175 and *RSSize*=20 (from the previous experiments), we checked all the possible combinations of *nQrs* and *nDrs*, knowing that their sum has to be 20 (and also that the number of diversity solutions must be equal to or lower than the number of quality solutions). Analysing the statistical results obtained (see Table 42), we

concluded that the division of *RSSize* between *nQrs*=14 and *nDrs*=6 was the one that performed better.

6.3.2.4 Determining the Combination Probability

The fourth experiment had the purpose of defining the bet crossover probability (*Cr* parameter) to be applied in the solution combination method. In order to perform this experiment, we used all the values of parameters already determined, and executed the following configurations for *Cr*: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9. Evaluating the best and average fitness costs, exposed in Table 43, we noticed that *Cr*=0.6 was the most adequate to proceed for the last experiment.

Table 43 –Results of the Experiment 4 to Determine the Best Crossover Probability – Parameter *Cr* (A – Average cost; B – Best Cost).

	Fitness Evaluation								
Cr	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
TN1 – B	98535	98535	98535	98535	98535	98535	98535	98535	98535
TN1 – A	98541.4	98535	98535	98535	98535	98535	98535	98535	98535
TN2 – B	97156	97156	97156	97156	97156	97156	97156	97156	97156
TN2 – A	97169.6	97162.8	97162.8	97159.4	97156	97156	97156	97156	97156
TN3 – B	95038	95038	95038	95038	95038	95038	95038	95038	95038
TN3 – A	95038	95038	95038	95038	95038	95038	95038	95038	95038
TN4 – B	173701	173701	173701	173701	173701	173701	173701	173701	173701
TN4 – A	175906.83	175112.83	174978.53	174751.77	174891.93	174636.87	174821.77	174869.4	174666.03
TN5 – B	182331	182331	182331	182331	182331	182331	182331	182331	182331
TN5 – A	183292.67	182643.77	182418.27	182469.23	182418.33	182399.47	182368.37	182399.47	182362.1
TN6 – B	174519	174519	174519	174519	174519	174519	174519	174519	174519
TN6 – A	175153.6	174838.5	174739.37	174729.7	174585.3	174573.93	174606.77	174519	174640.23
TN7 – B	309489	309050	307695	308401	308702	307695	307695	307695	307695
TN7 – A	312574.23	312202.17	311478.7	311612.2	311881.5	311057.67	311300.87	311296.4	310791.5
TN8 – B	287149	287149	287149	287149	287149	287149	287149	287149	287149
TN8 – A	291981.53	289716.47	291371.33	289179.97	289637.53	288541.77	288720.1	288752.17	288007.4
TN9 – B	264466	264204	264204	264204	264204	264204	264204	264204	264204
TN9 – A	266386.37	265550.47	264932.27	265046.33	265336.63	264767.27	265017.27	264610.6	264807.9
TN10 – B	386474	386816	386357	386474	385927	385927	385927	386474	385927
TN10 – A	390990.07	390688.43	389253.5	389397.2	389588.73	388974.37	388985.13	389425.3	389483.9
TN11 – B	358506	359377	358778	359286	358870	357823	358521	358259	358448
TN11 – A	361492.57	361601.77	361162.5	361785.03	361327.33	360801.27	361198.13	360863.87	360959.4
TN12 – B	372174	371606	372534	371164	371164	370868	370868	370868	371310
TN12 – A	377951.7	376673.93	376637.67	376409.73	375988.73	375526.4	375648.07	376282.4	375803.27

6.3.2.5 Determining the Ideal Local Search Number

In the last experiment, we had the objective of achieving the most adequate number of local search iterations (*nLS* parameter), which should represent the best performance of the improvement method. To execute this experiment, we assigned all the values of parameters defined in the previous experiments and tested the next values of *nLS*: 1, 2, 3, 4 and 5.

Analysing the results achieved (Table 44 shows the best and average costs obtained for the twelve test networks), we noticed that *nLS*=4 was the configuration that performs better for all the networks used, because with it we obtained the lowest (best) fitness costs and the majority of the best average results. Furthermore, we also noticed that with *nLS*=5 the statistical results became worst, so we elected *nLS*=4 as the most adequate for the final configuration of SS.

Table 44 –Results of the Experiment 5 to Define the Best Number of Local Search Iterations – Parameter *nLS* (A – Average Cost; B – Best Cost).

	Fitness Evaluation				
nLS	**1**	**2**	**3**	**4**	**5**
TN1 – B	98535	98535	98535	98535	98535
TN1 – A	98535	98535	98535	98535	98535
TN2 – B	97156	97156	97156	97156	97156
TN2 – A	97156	97156	97156	97156	97156
TN3 – B	95038	95038	95038	95038	95038
TN3 – A	95038	95038	95038	95038	95038
TN4 – B	173701	173701	173701	173701	173701
TN4 – A	174566.87	174647.27	174703.47	174593.03	174629.03
TN5 – B	182331	182331	182331	182331	182331
TN5 – A	182368.37	182331	182331	182331	182331
TN6 – B	174519	174519	174519	174519	174519
TN6 – A	174541.1	174596.03	174585.3	174563.2	174563.2
TN7 – B	307695	307695	307695	307695	307695
TN7 – A	311735.57	310910.67	310573.97	310411.17	310769.07
TN8 – B	287149	287149	287149	287149	287149
TN8 – A	289925.73	287543.37	287821.53	287500.2	287705.87
TN9 – B	264204	264204	264204	264204	264204
TN9 – A	264705.23	264688.53	264589.47	264506.7	264439.53
TN10 – B	386688	385927	385927	385927	385927
TN10 – A	389568.97	388144.97	387399.07	387142.77	387626.9
TN11 – B	358167	358397	358167	358033	357714
TN11 – A	361119.67	360333.07	359586.9	359079.3	359106.6
TN12 – B	371990	370868	370868	370868	370868
TN12 – A	376435.9	375344.33	373451.77	373194.77	372415.367

6.3.2.6 Statistical Analysis Using ANOVA

In addition, with the objective of reinforcing the conclusions achieved over the results of each experiment, we have performed a statistical analysis using the ANOVA test. We have considered a confidence level of 95% (this is, a 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%. In Table 45, we expose the results achieved over all the experiments, by using this test, where we have concluded that the fitness differences, when we use distinct values for each SS parameter, have been found as significant in most of the cases.

Table 45 – ANOVA analysis over SS parameters in the RCs problem, considering 12 test networks.

***PSize* Parameter**												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
***p*-Value**	4.94E-1	5.01E-1	7.56E-1	1.12E-2	3.30E-1	4.65E-1	1.20E-2	3.84E-1	2.04E-2	7.98E-1	4.58E-2	8.63E-1
***RSSize* 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
***nQrs/nDrs* Parameter**												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
***p*-Value**	1.30E-1	2.08E-2	<1E-15	4.29E-1	4.69E-1	1.86E-4	7.71E-1	1.69E-2	3.06E-2	1.46E-1	4.05E-1	6.43E-1
***Cr* Parameter**												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
***p*-Value**	4.36E-1	3.34E-2	<1E-15	3.840-7	4.9E-10	1.21E-9	9.68E-5	6.49E-5	1.09E-4	2.33E-2	4.20E-1	8.41E-3
***nLS* Parameter**												
Network	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
***p*-Value**	<1E-15	<1E-15	<1E-15	9.74E-1	2.35E-1	8.04E-1	1.26E-2	2.62E-4	4.87E-1	3.91E-7	2.99E-8	3.9E-13

6.3.2.7 Analysis and Comparison of Results

Finally, after we finished all the experiments, we had achieved the best configuration for the SS parameters when applied to the reporting cells problem. This configuration is: *PSize*=175, *RSSize*=20, which is divided in *nQrs*=14 and *nDrs*=6, *Cr*=0.6 and *nLS*=4.

Like for the DE based approach, we have decided to compare our SS results with those presented by Alba et al. in [10], which have used a Hopfiel Neural Network with Ball Dropping (HNN-BD) and a Geometric Particle Swarm Optimization (GPSO). Comparing the results, we noticed that our SS approach outperforms the results achieved

by HNN-BD and GPSO for the networks 7, 10 and 11, and equals the lowest fitness costs for all the other networks.

In Figure 41 we expose the configuration for the best solution for the networks 7, 10 and 11, those in which SS always surpasses the results achieved by the other approaches.

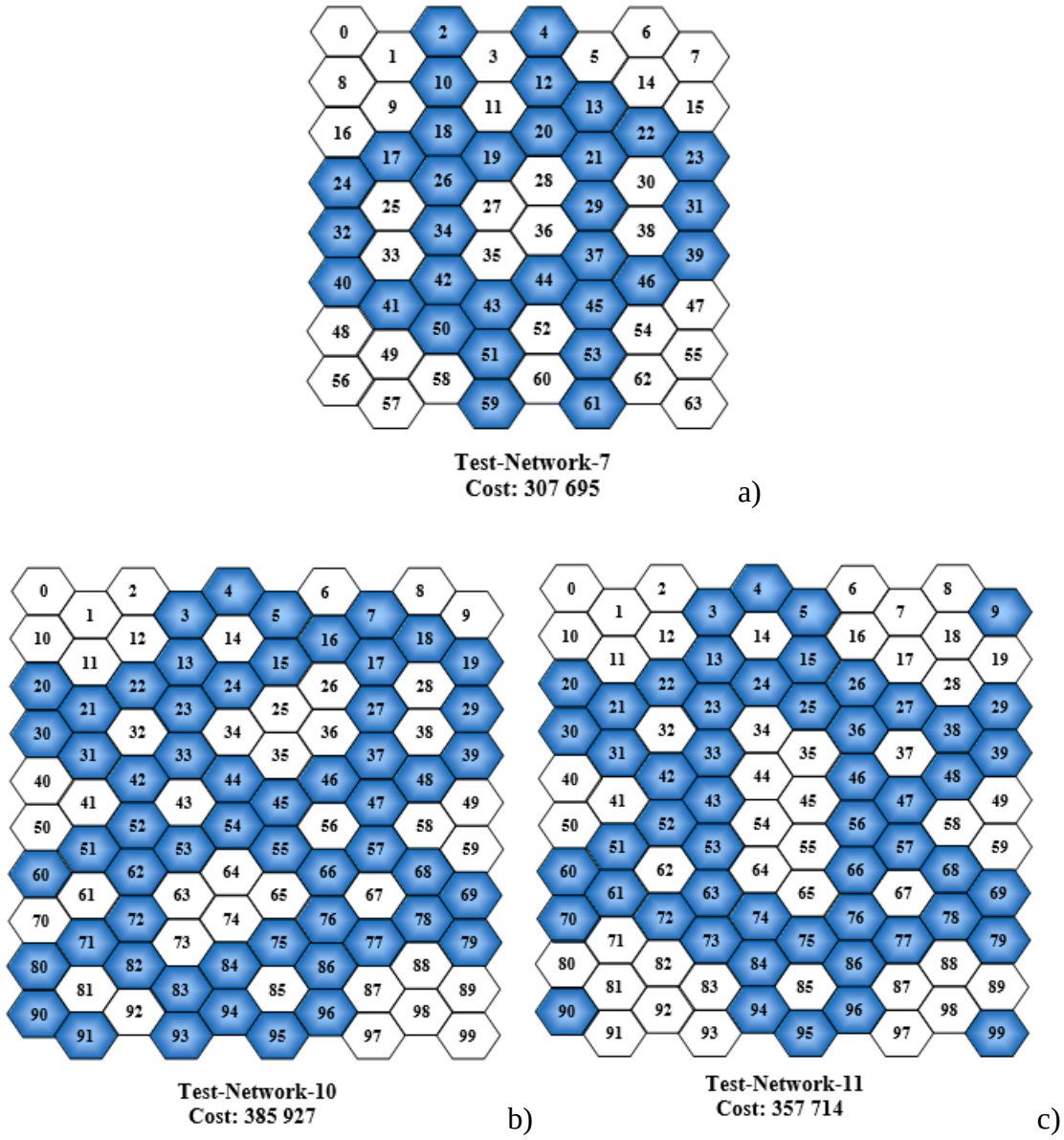


Figure 41 – Configuration of the best RCs configuration achieved with the SS based approach for the test networks 7 (a), 10 (b) and 11 (c).

6.3.3 Summary

In this section we have exposed and discussed a new approach based on SS algorithm applied to the RCs problem. Like for the DE based approach, for the best of our knowledge, it was the first time that the SS algorithm is used with the aim of solving the RCs problem.

We started with the explanation of the decisions taken about the implementation details specific of SS when applied to the RC scheme. After that, we have implemented the SS based approach following with the execution and exposition of all the experiments performed.

Considering the RCs problem and using the SS based approach, the best configuration achieved, after finished all the experiments with the purpose of tuning all the parameters, is: *PSize*=175, *RSSize*=20 divided in *nQrs*=14 and *nDrs*=6, *Cr*=0.6 and *nLS*=4.

When we have compared the results obtained by the SS based approach with those accomplished with the DE based approach, again we concluded that SS performs better because it obtains equal or better results for all the networks tested. Finally, comparing the results achieved by our SS based approach with those presented by Alba et al. in [10], which have used a Hopfiel Neural Network with Ball Dropping (HNN-BD) and a Geometric Particle Swarm Optimization (GPSO), we noticed that our SS approach outperforms the results achieved by HNN-BD and GPSO for the networks 7, 10 and 11, and equals the lowest fitness costs for all the other networks.

With these results we may conclude that our SS based approach applied to the RCs problem is very promising and that is a big encouragement to follow with our work.

6.4. Results Comparison

Another goal of this study was to compare the results obtained by the SS approach, when applied to the RCs problem, with our previous work in which we have used an approach based on the DE algorithm, described earlier and also presented in [103]. With

this comparison we noticed that SS presents the best results (for test networks 7, 10, 11 and 12) when compared with DE results, as it is possible to observe in Table 46.

Table 46 – Comparison of the Best Location Management Costs, for the 12 Test Networks, obtained by DE and SS based approaches.

	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
SS	98535	97156	95038	173701	182331	174519	307695	287149	264204	385927	357714	370868
DE	98535	97156	95038	173701	182331	174519	308401	287149	264204	386681	358167	371829

After that, we also decided to compare our SS results with those presented by Alba et al. in [10], which have used a Hopfiel Neural Network with Ball Dropping (HNN-BD) and a Geometric Particle Swarm Optimization (GPSO). Observing

Table 47, we noticed that our SS approach outperforms the results achieved by HNN-BD and GPSO for the networks 7, 10 and 11, and equals the lowest fitness costs for all the other networks.

Table 47 – Comparison of the Best Location Management Costs, for the 12 Test Networks.

	TN1	TN2	TN3	TN4	TN5	TN6	TN7	TN8	TN9	TN10	TN11	TN12
SS	98535	97156	95038	173701	182331	174519	307695	287149	264204	385927	357714	370868
DE	98535	97156	95038	173701	182331	174519	308401	287149	264204	386681	358167	371829
HNN-BD	98535	97156	95038	173701	182331	174519	308929	287149	264204	386351	358167	370868
GPSO	98535	97156	95038	173701	182331	174519	308401	287149	264204	385972	359191	370868

6.5. Comparison of Results with Other Authors

After obtaining the best DE and SS configurations applied to the reporting cells planning problem we decide to apply them in other test networks, and analyze the respective performance with the objective of comparing with other artificial life techniques.

We started this comparison selecting 3 distinct test networks presented in [91] and also used in [10] and [103]. Using these networks, which represent respectively 4x4, 6x6 and 8x8 instances, we have the goal of comparing our DE and SS results with those achieved by Genetic Algorithms (GA), Ant Colony algorithm (AC) and Tabu Search (TS). With these additional experiments, we have noticed the good performance of DE and SS approaches because, for all the 3 networks (executing 30 runs for each one), we always obtain equal or better fitness solutions (lower LM costs), as it is possible to observe in Table 48.

Table 48 – Comparison of LM Costs for Additional Networks using SS, DE, GA, AC and TS.

Test Network	SS	DE	GA	AC	TS
4x4	92833	92833	92833	92833	92833
6x6	211278	211278	229556	211291	211278
8x8	436269	436269	436283	436886	436283

Using the Test-Network-1 (4×4 instance provided in [91]) we obtain the same fitness value as the GA, TS and AC, that is, 92883 with a total of 10 reporting cells.

Our approach generates a better solution when solving the Test-Network-2 (6×6 instance provided in [91]), comparing with GA and AC. The fitness value obtained by the GA is 229556 with a total of 26 reporting cells in the network [10, 91], while, the cost obtained by DE and SS in this work is 211278 with 24 reporting cells. TS presents the same cost 211278 and the fitness value obtained by AC is 211291.

For the Test-Network-3 (8×8 instance in [91]) DE an SS again surpass the results obtained in [91] by the GA, TS and AC. Specifically, the fitness value obtained by the GA and TS is 436283, the one obtained by AC is 436886, while the cost obtained by DE and SS in this work is 436269 with a total of 39 reporting cells.

Finally, we have selected two bigger networks, respectively 7x9 and 9x11 instances, provided in [92], which use a combination of Hopfield Neural Network (HNN) and Ball Dropping Technique (BDT); and also referred in [103, 104]. 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 [92].

Observing Table 49, when we applied our SS approach to the 7x9 instance, we noticed that the results achieved surpass the LM costs obtained by all the other approaches. That is, with SS we achieved the fitness value of 120052, considering 27 RCs, while with DE [103] the best solution corresponds to a fitness value of 120904 with 28 RCs, and the HNN-BDT [92, 104] obtained the fitness value of 123474 with 27 RCs.

Table 49 – Comparison of LM Costs for Additional Networks using SS, DE and HNN-BD.

Test Network	SS	DE	HNN -BD
7x9	120052	120904	123474
9x11	242924	243957	243414

Relatively to the 9x11 instance (see Table 49), with the SS approach we obtained a fitness value of 242924 cost units, with 44 RCs, which surpasses the lowest fitness values achieved by the DE approach (which represents 243957 cost units, with 47 RCs) and also by HNN-BDT (which is of 243414 cost units, with 43 RCs).

After finishing all the additional experiments, it is also possible to conclude that our SS based approach, when applied to the RCs problem, is very competitive, considering that it outperforms the results accomplished by our DE based approach and also using other artificial life techniques.

6.6. Location Areas vs. Reporting Cells Strategies

LAs and RCs are two common strategies of Location Management, which both consider update cost and paging cost for the measurement of respective LM costs.

Considering this, we decided to proceed with new experiments, with the goal of comparing the results of these two strategies.

6.6.1 Comparison of LM Costs

To accomplish the goal of comparing the results obtained by these two strategies, we used the four test networks exposed in section 5.1.1 *Networks Configuration*, those already used with the experiments of LA approaches, because these are the ones that give us the necessary data for fitness evaluation of each solution generated.

After selection of the appropriate test networks we decided to apply the best configuration achieved for each approach, based on the algorithms DE and SS, respectively, and analyse the results obtained.

In Table 50, we expose the best and average fitness results obtained by each approach. Comparing the results achieved by the LAs based approaches with those achieved by the RCs based approaches, we noticed that the results obtained using the LA strategy are more adequate and accomplish the lowest LM costs. This means that, considering the update costs and the paging costs, the common ones in these two strategies of location management, the configurations obtained by the LA strategy are more cost efficient than those obtained by the RC strategy.

Table 50 – Comparison of LM Costs: LAs vs RCs Strategies

Strategy	**LA**		**RC**	
Algorithm	**DE**	**SS**	**DE**	**SS**
5x5 Network				
Best	26990	26990	35844	35844
Average	27077.77	26990	36802.9	35867.8
5x7 Network				
Best	40205	39832	52225	52225
Average	41261.77	39832	53464.3	52225
7x7 Network				
Best	63307	60685	79168	78284
Average	65737.13	60708.13	80779.5	78595.5
7x9 Network				
Best	94841	89085	116404	116404
Average	97895.13	89485.4	120099.1	116864.7

Considering the algorithms used, as we have noticed earlier in the results comparison of each approach, we concluded that the SS always equals or surpasses the results obtained by DE.

7. Conclusions

In this chapter we will describe the main results achieved through our work considering all the investigation, implementation and execution phases. After that we outline some future lines of work that may be followed for solving the location management problem, considering the Location Area and Reporting Cell strategies. Finally, we will expose the scientific work produced along this work, including the listing of publications and scientific events.

7.1. Main Contributions of this Research

Current mobile networks, namely and more specifically the personal communication networks (PCN), must support communications that enable all the users to make or receive calls at any time of the day and for, or from, any location. In order these networks support the mobility of users and being able to find them, also when they change their location, it is necessary to consider mobility management and more precisely Location Management (LM). In conclusion, the Location Management is a very important and critical task for current (and future) 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 was also our major goal. There exist several strategies of location management and we have developed several approaches over the Location Area (LA) and Reporting Cell (RC) schemes, considering that they are two of the most common ones.

The location management is an NP-hard optimization problem, therefore, the use of evolutionary algorithms is a very interesting approach. Due to this, we decided to apply Differential Evolution (DE) and Scatter Search (SS) algorithms to both strategies and the Greedy Randomized Adaptive Search Procedure based (GRASP) algorithm to the Location Areas strategy.

Our work was based on the development of distinct approaches for solving the LAs and RCs problems, which principally consider the location update and paging costs, with the objective of determining the best configuration of each of these evolutionary algorithms. To accomplish these objectives we have executed many experiments using test networks, and realistic networks, as well as compared our results with the ones obtained by other authors with the goal of surpassing those other results.

Considering the LAs based approaches, we have shown that all of them improve the results obtained with other classical location management strategies as always-update and never-update.

Our first approach developed was based on DE applied to the LAs strategy using realistic and test networks. When the implementation results were compared with the ones of other authors, it was possible to conclude that they are considered interesting because they are equal or better, when applied to the same test networks. Also when we applied our approach to realistic networks, based on SUMATRA data, using static LAs, the results were very similar when compared with the ones of other authors, which are using dynamic LAs.

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

With the goal of achieving the best configuration for DE parameters we have executed a big number of experiments (more than 5000 independent runs) with four distinct test networks. After this tuning of the algorithm parameters we have determined the best configuration as: NI=250, Cr=0.1; F=0.5; and the DE scheme as *DE/rand/1/bin*.

Furthermore, we have also studied the best configuration of DE using networks based on realistic data with the goal of understanding the influence of the DE parameters and schemes over this type of networks. The best configuration achieved was NI=300, Cr=0.05, F=0.5 and DE scheme as *DE/rand/1/bin*, which is similar to the one obtained for test networks.

Analysing the best DE configurations accomplished it is also possible to conclude that in general the binomial schemes perform better than the exponential ones.

Relatively to the second approach developed, we have applied a SS based algorithm with the objective of finding the best configuration for the LAs strategy.

With this approach, we have studied in detail the best configuration of SS, applied to the LAs problem. The best parameters, after a big number of experiments with four distinct test networks, are *PSize*=100, *RSSize*=18 with *nQrs*=9 and *nDrs*=9, *Cr*=0.2 and *nLS*=5. Following these experiments, we have used the scatter search (SS) algorithm to solve the location area problem applied to SUMATRA networks, with the goal of determining the best configuration of SS using these networks based on realistic data. The best values for the SS parameters are *PSize*=175, *RSSize*=16, *nQrs*=15, *nDrs*=1, *Cr*=0.1 and *nLS*=7.

Comparing these two approaches we concluded that SS performs better than DE, because it always obtains equal or better fitness solutions. Furthermore, if we compare the SS results with those achieved using other algorithms as Genetic Algorithm (GA), Hopfield Neural Network (HNN), Simulated Annealing (SA) and different combinations of genetic algorithm and Hopfield neural network (GA-HNNx), it also presents a good performance because always achieves equal or even better solutions, which means solutions with lower LM costs.

After the previous work presented, we have studied and analysed the implementation of a new approach based on the GRASP algorithm for solving the LA problem taking in consideration that all the experiments were executed using grid computing.

Considering the grid technology, we may say that the results obtained are very satisfactory, because with it we have the possibility of solving complex problems which, in other way would be conditioned by the time of execution. The sum of all the execution time of all the GRASP experiments corresponds to a total of 8381 hours, or more specifically 349.21 days (about 1 year). This values of execution time leads us with the real notion of the importance of the grid and parallel computing.

Relatively to the results achieved, we have defined the best configuration of the GRASP based approach when applied to the LA problem. The results achieved with the configuration of sequential GRASP are very similar to those obtained by other authors. The parallel version of GRASP always surpasses the sequential results and the best

configuration is compound by 75 slave nodes and 4 synchronizations with the master node.

Finally, taking in consideration the distribution of the five best distinct configurations of GRASP, inside the parallel team, we concluded that the most adequate variant is RG, which is characterized by being first random and latter greedy.

Following the study and implementations over the LAs strategy we decided to explore the Reporting Cells strategy. Considering this, we have implemented two more approaches based on DE and SS algorithms, with the goal of finding their best configuration when applied to the RCs and that minimizes the LM costs. For the best of our knowledge, it was the first time that these algorithms were applied with the aim of solving the RCs problem.

After all the experiments performed, more than 10000 runs, to study in detail the best configuration of DE including parameters and scheme, they are *NI*=175, *Cr*=0.15, *F*=0.5 and *DE/rand/1/bin* as the best DE scheme. Observing the configuration for each test-network solution, we noticed that the most of the networks were partitioned into subnetworks.

Using the SS based approach over the RCs problem, the best configuration achieved, after finished all the experiments with the purpose of tuning all the parameters and minimizing the LM costs, was: *PSize*=175, *RSSize*=20 divided in *nQrs*=14 and *nDrs*=6, *Cr*=0.6 and *nLS*=4.

Comparing the results obtained by the SS based approach with those accomplished with the DE based approach, again we concluded that SS performs better because it obtains equal or better results for all the networks tested.

Furthermore, after executing additional experiments, it was possible to conclude that our SS based approach, when applied to the RCs problem, is very competitive, considering that it outperforms the results accomplished using other artificial life techniques as Hopfiel Neural Network with Ball Dropping (HNN-BD), Geometric Particle Swarm Optimization (GPSO), Genet Algorithms (GA), Ant Colony algorithm (AC) and Tabu Search (TS).

A final goal of this work was to compare directly LA and RCs strategies to determine the most adequate to accomplish the lowest LM costs, which are the update costs and the paging cost. Considering these costs we concluded that the LAs strategy is the most adequate because its solution obtained (this is, the network configurations achieved) are more cost efficient than those achieved by the RC strategy. Relatively to the algorithms used, as we have noticed earlier in the results comparison of each approach, we concluded that SS always equals or surpasses the results obtained by DE.

7.2. Future Work

Concerning the investigation and work described in this PhD Thesis, there is a set of directions that may be taken as following research work considering the optimization of the Location Management problem.

As future work we consider that the application of other evolutionary algorithms to LA and RC strategies, with the objective of making the comparison of their results with the ones accomplished by the DE and SS based approaches would be an interesting step.

The exploration of other static location management strategies, as well as the analysis and investigation of dynamic LM strategies could be an interesting direction of this investigation.

Furthermore, the execution of distinct experiments, using bigger test networks, and mainly using new networks based on real GSM systems (or other mobile systems), is also an opportunity of deepening this line of investigation.

Finally, the formulation of the LA and the RC respective problems as multi-objective optimization problems may be also a good course to following with this investigation. This multi-objective implementation could be implemented defining two fitness functions, each one with the goal of minimizing, respectively, the location update and location paging costs, as well as other possible multi-objective formulations.

7.3. Publications

Considering the investigation and experiments performed to accomplish this dissertation, several publications have been published in international journals and conference proceedings as follows:

Impact Factor Journals

1. Sónia M. Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: Differential Evolution for Solving the Mobile Location Management. Applied Soft Computing, Volume 11, Issue 1, Elsevier Science, Amsterdam, Netherlands, 2011, pag:410-427, ISSN:1568-4946. (Impact factor = 2.612, Quartile Q1, Journal Rank = 13 of 111)

2. David L. González-Álvarez, Álvaro Rubio-Largo, Miguel A. Vega-Rodríguez, Sónia M. Almeida-Luz, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: Solving the Reporting Cells Problem by Using a Parallel Team of Evolutionary Algorithms. Logic Journal of the IGPL, Volume 20, Issue 4, Oxford University Press, Oxford, England, 2012, pag:722-731, ISSN:1367-0751. (Impact factor = 1.136, Quartile Q1, Journal Rank = 1 of 19)

Other International Journals

3. Sónia M. Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A., Gómez-Pulido, Juan M., Sánchez-Pérez, Solving the Location Area Problem by Using Differential Evolution. Journal of Communications Software and Systems. Volume 4, Issue 2, pag: 131-141 (2008). ISSN: 1845-6421

International Book Chapters

4. Sónia Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: Defining the Best Parameters in a Differential Evolution Algorithm for Location Area Problem in Mobile Networks. In: "New Trends in Artificial Intelligence". J. Neves, M.F. Santos, J.M. Machado (Eds). APPIA, Associação Portuguesa Para a Inteligência Artificial, Braga, Portugal, 2007, pag: 219-230. ISBN: 978-98-995-6180-9.

LNCS-Springer

5. Sónia M. Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: Applying Scatter Search to the Location Areas Problem in: "Intelligent Data Engineering and Automated Learning. LNCS, vol. 5788". Springer-Verlag, Berlin Heidelberg, Germany, 2009, pag: 791-798. ISBN: 978-3-642-04393-2.

6. Sónia M. Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez.: Solving the Reporting Cells Problem Using a Scatter Search Based Algorithm. In: "Rough Sets and Current Trends in Computing. LNAI, vol. 6086". Springer-Verlag, Berlin Heidelberg, Germany, 2010, pag: 534-543. ISBN: 978-3-642-13528-6.

7. Álvaro Rubio-Largo, David L. González-Álvarez, Miguel A. Vega-Rodríguez, Sónia M. Almeida-Luz, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: A Parallel Cooperative Evolutionary Strategy for Solving the Reporting Cells Problem. In: "Soft Computing Models in Industrial and Environmental Applications. Advances in Intelligent and Soft Computing, vol. 73". Springer-Verlag, Berlin Heidelberg, Germany, 2010, pag: 71-78. ISBN: 978-3-642-13160-8.

8. Sónia M. Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: A Scatter Search Based Approach to Solve the Reporting Cells Problem. In: "Soft Computing Models in Industrial and Environmental Applications. Advances in Intelligent and Soft Computing, vol. 73". Springer-Verlag, Berlin Heidelberg, Germany, 2010, pag: 145-152. ISBN: 978-3-642-13160-8.

IEEE Conferences

9. Sónia Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez- Pulido, Juan M. Sánchez-Pérez.: Applying Differential Evolution to the Reporting Cells Problem. In Proceedings of the International Multiconference on Computer Science and Information Technology (IMCSIT 2008), Polskie Towarzystwo Informatyczne (Poland). IEEE Computer Society, Wisla, Poland, 2008, pag:65-71. ISBN: 978-83-60810-14-9.

10. Sónia Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan 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 2008), IEEE Computer Society, Valencia, Spain, 2008, pag:170-175. ISBN: 978-0-7695-3369-8.

11. Sónia Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: Solving a Realistic Location Area Problem Using SUMATRA Networks with the Scatter Search Algorithm. In Proceedings of the 9th International Conference on Intelligent Systems Design and Applications, IEEE Computer Society, Pisa, Italy, 2009, pag:689-694. ISBN: 978-0-7695-3872-3.

12. Sónia Almeida-Luz, Manuel M. Rodríguez-Hermoso, Miguel A. Vega-Rodríguez, Juan A. Gómez- Pulido, Juan M. Sánchez-Pérez: GRASP and Grid Computing to Solve the Location Area Problem. In Proceedings of the World Congress on Nature and Biologically Inspired Computing, IEEE Computer Society, Coimbatore, India, 2009, pag:164-169. ISBN: 978-1-4244-5612-3.

Other International Conferences

13. Sónia Almeida-Luz, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: A Differential Evolution Algorithm for Location Area Problem in Mobile Networks. In Proceedings of 15th International Conference on Software, Telecommunications and Computer Networks, FESB, University of Split, Split, Croatia, 2007, pag:1-5. ISBN: 953-6114-95-X.

14. David L. González-Álvarez, Álvaro Rubio-Largo, Miguel A. Vega-Rodríguez, Sónia Almeida-Luz, Juan A. Gómez-Pulido, Juan M. Sánchez-Pérez: Resolución del Problema Reporting Cells mediante Computación Clúster y un Equipo Paralelo de Algoritmos Evolutivos. XX Jornadas de Paralelismo, Servizo de Publicacións, Universidade da Coruña, A Coruña, Spain, 2009, pag:69-74. ISBN:84-9749-346-8.

7.4. Scientific Events

Along the time working to develop this Thesis, I have been member of the technical program committee or reviewer of the following international conferences and impact factor journals:

- The Third International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2009), October 11-16, 2009 – Sliema, Malta. See http://www.iaria.org/conferences2009/ComADVCOMP09.html (ADVCOMP 2009 Technical Program Committee).

- The Fourth International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2010), October 25-30, 2010 – Florence, Italy. See http://www.iaria.org/conferences2010/ComADVCOMP10.html (ADVCOMP 2010 Technical Program Committee).

- The 11th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL2010). University of the West of Scotland, Glasgow, Scotland, UK, from 1st to 3rd September 2010. See http://ideal2010.ucc.ie/Committees.html (Program committee of the 11th IDEAL).

- ITBI 2010, International Conference on Information Technology & Business Intelligence (ITBI-10). Institute of Management Technology, from 12th to 14th November 2010, Nagpur – India. See http://imtnagpur.ac.in/itbi10/index.html (ITBI 2010 International Program committee).

- The 12th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL2011). University of East Anglia, Norwich, UK, from 7th to 9th September 2011. See http://www.uea.ac.uk/ideal2011 (Program committee of the 12th IDEAL).

- The Fifth International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2011), November 20-25, 2011 - Lisbon,

Portugal. See http://www.iaria.org/conferences2011/ComADVCOMP11.html (ADVCOMP 2011 Technical Program Committee)

- The Sixth International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2012), September 23-28, 2012 - Barcelona, Spain. See http://www.iaria.org/conferences2012/ComADVCOMP12.html (ADVCOMP 2012 Technical Program Committee).

- The Seventh International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2013), September 29 - October 3, 2013 - Porto, Portugal. See http://www.iaria.org/conferences2013/ComADVCOMP13.html (ADVCOMP 2013 Technical Program Committee).

- PBio 2013: International Workshop on Parallelism in Bioinformatics, 17 September 2013, to be held as part of ACM EuroMPI 2013, 15 – 18 September 2013 – Madrid, Spain. (see http://www.eurompi2013.org and http://arco.unex.es/mavega/pbio2013) (PBio 2013 Program committee).

- The Eighth International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2014), August 24 – 28, 2014 - Rome, Italy. See http://www.iaria.org/conferences2014/ComADVCOMP14.html (ADVCOMP 2014 Technical Program Committee).

- PBio 2014: Second International Workshop on Parallelism in Bioinformatics, 26 September 2014, to be held as part of IEEE Cluster 2014, 22 – 26 September 2014 – Madrid, Spain. (see http://www.cluster2014.org and http://arco.unex.es/mavega/pbio2014) (PBio 2014 Program committee).

- Applied Soft Computing – The Official Journal of the World Federation on Soft Computing (WFSC), Elsevier Science. ISSN: 1568-4946 (Impact factor = 2.679 in 2013, Quartile Q1, Journal Rank = 14 of 102). See http://ees.elsevier.com/asoc/. (Reviewer from 2009).

- AEÜ - International Journal of Electronics and Communications, Elsevier Science. ISSN: 1434-8411 (Impact factor = 0.696 in 2013, Quartile Q3, Journal Rank = 177 of 248). See http://ees.elsevier.com/aeue/. (Reviewer from 2014).

Bibliography

1– K. Pahlavan, A.H. Levesque: Wireless Information Networks. John Wiley & Sons, 2nd edition, 2005.

2– GSM World. Jan 2011, URL http://www.gsmworld.com/newsroom/.

3– M. Mouly and M.-B. Pautet: The GSM System for Mobile Communications, Telecom Publishing, 1992.

4– D. Plassmann: Location Management Strategies for Mobile Cellular Networks of 3rd Generation. Proc. of 44th IEEE Vehicular Technology Conference, pp. 649-653, June 1994.

5– 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.

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

7– 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, November 2005, pp. 568-577.

8– J. Cowling: Dynamic Location Management in Heterogeneous Cellular Networks, MIT Thesis, 2004, URL: http://people.csail.mit.edu/cowling/.

9– A. Bar-Noy, I. Kessler: Tracking mobile users in wireless communications networks. IEEE Transactions on Information Theory, 1993, vol. 39, no. 6, pp. 1877-1886.

10– E. Alba, J. García-Nieto, J. Taheri, A. Zomaya: New Research in Nature Inspired Algorithms for Mobility Management in GSM Networks. EvoWorkshops, Springer, LNCS 4974, March 2008, 1-10.

11– K. Price, R. Storn: Web Site of Differential Evolution (on July 2014), the URL of which is: http://www.icsi.berkeley.edu/~storn/code.htm.

12– R. Storn, K. Price: Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces, Journal of Global Optimization, Kluwer Academic Publishers, December 1997, vol. 11, no. 4., pp. 341-359.

13– K. Price: An Introduction to Differential Evolution. New Ideas in Optimization, D. Corne, M. Dorigo and F. Glover, Eds. London: McGraw-Hill, 1999, pp. 79-108.

14– F. Glover: Heuristics for integer programming using surrogate constraints. Decision Sciences, vol. 8, no. 1, pp. 156-166, 1977.

15– R. Martí, M. Laguna, F. Glover: Principles of Scatter Search. European Journal of Operational Research, vol. 169, no.2, pp. 359-372, 2006.

16– M. Laguna, K.P. Hossell, R. Martí: Scatter Search: Methodology and Implementation in C. Kluwer Academic Publishers, Norwell, 2002.

17– M. Resende, C. Ribeiro: Greedy Randomized Adaptive Search Procedures. AT&T Labs Research Tech. Report TD-53RSJY, version 2, 2002, pp. 1–29.

18– L. S. Pitsoulis, M. G.C. Resende: Greedy Randomized Adaptive Search Procedures. Handbook of Applied Optimization, Oxford University Press, 2002, pp. 168-183.

19– T. A. Feo, M. G. C. Resende: Greedy randomized adaptive search procedures. Journal of Global Optimization, vol. 6, pp. 109-133, 1995.

20– J. Taheri, A. Y. 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, pp.194–201, April 2005.

21– Test networks benchmark: http://oplink.lcc.uma.es/problems/mmp.html (on July 2014).

22– SUMATRA, 1998. Stanford University Mobile Activity TRAces (online). InfoLab - Stanford University, Stanford. Available from: http://infolab.stanford.edu/sumatra/ (on July 2014).

23– D. Lam, J. Jannink, D. C. Cox, J. Widom: Modelling location management for personal communication services. In: ICUPC'96: Proceedings of 1996 IEEE International Conference on Universal Personal Communications, vol. 2, pp. 596-601, 1996.

24– R. Subrata, A. Zomaya: Dynamic Location Management for Mobile Computing. Telecommunication Systems, vol. 22, pp. 169-187, 2003.

25– EIA/TIA: Cellular radio telecommunications intersystem operations. Technical Report, July 1991.

26– M. Rahnema: Overview of the GSM system and protocol architecture. IEEE Communications Magazine, vol. 31, no. 4, pp. 92-100, 1993.

27– T. Keshav: Location Management in Wireless Cellular Networks, on Student Reports on Recent Advances in Wireless and Mobile Networking. (2006) (on July 2014): http://www.cs.wustl.edu/~jain/cse574-06/cellular_location.htm

28– B. Sidhu, H. Singh: Location Management in Cellular Networks. In Proceedings of World Academy of Science, Engineering and Technology, pp. 314-319, 2007.

29– S. G. Glisic: Advanced Wireless Networks - 4G Technologies. Jonh Wiley & Sons, 2006.

30– P.G. Escalle, V.C. Giner, J.M. Oltra: Reducing location update and paging costs in a PCS network. IEEE Transactions on Wireless Communications, vol. 1, no. 1, pp. 200-209, Jan, 2002.

31– A. Bar-Noy, I. Kessler, M. Sidi: Mobile users: To update or not to update?, Wireless Networks, vol. 1 no. 2, pp. 175-185, 1995.

32– S. Ramanathan, M. Steenstrup: A survey of routing techniques for mobile communication networks. Mobile Networks and Applications, vol. 1, pp. 89-104, 1996.

33– J. Zhang: Location Management in Cellular Networks. Handbook of Wireless Networks and Mobile Computing, pp. 27-47, Ivan Stojmenovic (Editor), John Wiley & Sons, Inc. New York, NY, USA, 2002.

34– J. Taheri, A. Y. Zomaya: Clustering techniques for dynamic location management in mobile computing. Journal of Parallel Distributed Computing, vol. 67, no. 4, pp. 430-447, 2007.

35– C. Rose: Minimizing the average cost of paging and registration: A timer-based method. ACM Wireless Networks, vol. 2, no. 2, pp. 109–116, 1996.1

36– Z. Naor, H. Levy: Minimizing the wireless cost of tracking mobile users: an adaptive threshold scheme. In Proceedings of INFOCOM ’98, IEEE, vol.2, pp. 720-727, 1998.

37– Z. Naor, H. Levy: LATS: A Load-Adaptive Threshold Scheme for Tracking Mobile Users. IEEE/ACM Transactions on Networking, vol. 7, no. 6, pp. 808-817, Dec. 1999.

38– V. Bhatt and D. S. Khatri: Timer Based Dynamic Location Management. IEEE, Wireless Telecommunications Symposium, pp. 55-60, 2004.

39– R. Ahirwal, A. Sahelay: Timer based Location Management in Cellular Network using Cache Memory. International Journal of Computer Applications, vol. 39, no.15, pp. 35-42, February 2012.

40– B. Liang, Z.J. Haas: Predictive distance-based mobility management for PCS networks. In Proceedings of INFOCOM '99, vol.3, pp. 1377-1384, 1999.

41– I. Han, S.-S. Choi, D.-H. Cho: An automatic drawing method for location tracking in wireless networks. IEEE 53rd Vehicular Technology Conference, VTC 2001, vol. 4, pp. 2555-2559, 2001.

42– I.F. Akyildiz, J.S.M. Ho, Y.-B. Lin: Movement-based location update and selective paging for PCS networks. IEEE/ACM Transactions on Networking, vol. 4, no. 4, pp. 629–638, Aug 1996.

43– B.-K. Lee, C.-S. Hwang: A predictive paging scheme based on the movement direction of a mobile host. IEEE Vehicular Technology Conference, vol. 4, pp. 2158-2162, 1999.

44– Y. Fang: Movement-Based Mobility Management and Trade Off Analysis for Wireless Mobile Networks. IEEE Transactions on Computers, vol. 52, no. 6, pp.791-803, June 2003.

45– J.S.M. Ho, J. Xu: History-based location tracking for personal communications networks. In Proceedings of 48th IEEE Vehicular Technology Conference, vol. 1, pp. 244-248, 1998.

46– A. Rahaman, J. Abawajy, M. Hobbs: Taxonomy and Survey of Location Management Systems. In Proceedings of 6th IEEE/ACIS Int. Conf. on Computer and Information Science, Melbourne, pp. 369-374, 2007.

47– U. Varshney: Location management for wireless networks: issues and directions. Int. Journal of Mobile Communications, vol. 1 no. 1-2, pp. 91-118, July 2003.

48– R. Subrata and A. Y. Zomaya: Evolving cellular automata for location management in mobile computing networks. IEEE Transactions on Parallel and Distributed Systems, vol. 14, no. 1, pp. 13-26, 2003.

49– I.F. Akyildiz, J. McNair, J.S.M. Ho, H. Uzunalioglu, and W. Wang: Mobility management in next-generation wireless systems. Proceedings of the IEEE, vol. 87, no. 8, pp.1347–1384, 1999.

50– C. Blum, A. Roli: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing. Surveys, vol. 35(3), pp. 268–308, 2003.

51– I. H. Osman and G. Laporte: Metaheuristics: A bibliography, Annals of Operations Research, vol. 63: pp. 513-623, 1996.

52– S. Voß, S. Martello, I. H. Osman, and C. Roucairol, editors. Meta-Heuristics - Advances and Trends in Local Search Paradigms for Optimization. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999.

53– T. Bäck, D. B. Fogel and Z. Michalewicz: Handbook of Evolutionary Computation. New York NY: Oxford University Press, 1997.

54– L. J. Fogel, A. J. Owens and M. J. Walsh: Artificial Intelligence through Simulated Evolution. New York: Wiley, 1966.

55– X. Yao, Y. Liu and G. Lin: Evolutionary Programming Made Faster. IEEE Transactions on Evolutionary Computation, vol. 3, no. 2, pp. 82-102, July 1999.

56– J.H. Holland: Adaptation in natural and artificial system, Ann Arbor, The University of Michigan Press, 1975.

57– J.H. Holland: Outline for a Logical Theory of Adaptive systems. Journal of the ACM, vol. 9, no. 3: pp. 297–314, 1962.

58– D. E. Goldberg: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

59– R. Storn and K. Price: Differential evolution - a simple evolution strategy for fast optimization. Dr. Dobb's Journal, vol. 22, no. 4, pp. 18-24, April 1997.

60– R K. V. Price, R. M. Storn, and J. A. Lampinen: Differential Evolution: A Practical Approach to Global Optimization. Natural Computing Series. Springer; 1 edition, December 2005.

61– F. Glover: Scatter search and path relinking. In New ideas in optimization, McGraw-Hill Ltd., UK, Maidenhead, UK, England, 1999, pp. 297–316.

62– F. Glover, M. Laguna, and R. Martí: Fundamentals of scatter search and path relinking. Control and Cybernetics, vol. 39: pp. 653–684, 2000.

63– T. Bäck: Evolutionary Algorithms in Theory and Practice. Oxford University Press. 1996.

64– D. C. Dracopaulos: Evolutionary learning algorithms for neural adaptive control. Springer, 1997.

65– T. A. Feo, M. G. C. Resende: Greedy randomized adaptive search procedures. Journal of Global Optimization, vol. 6: pp. 109-133, 1995.

66– F. Glover: Heuristics for integer programming using surrogate constraints. Decision Sciences, vol. 8, no. 1:156–166, 1977.

67– F. Glover: Future paths for integer programming and links to artificial intelligence. Computer and Operations Research, vol. 13, no. 5: pp. 533–549, 1986.

68– F. Glover and M. Laguna: Tabu Search. Kluwer Academic Publishers, 1997.

69– S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi: Optimization by simulated annealing. Science, vol. 220: pp. 671–680, May, 1983.

70– M. Fleischer: Simulated Annealing: past, present and future. In C. Alexopoulos, K. Kang, W.R. Lilegdon, and G. Goldsman, editors, Proceedings of the 1995 Winter Simulation Conference, pp. 155-161, 1995.

71– P. Hansen, N. Mladenovic: Variable neighborhood search: Principles and applications. European Journal of Operational Research, vol. 130, no. 3: pp. 449–467, May 2001.

72– P. Hansen, N. Mladenovic: "Variable neighborhood search". In F. Glover and G. A. Kochenberger, editors, Handbook of Metaheuristics, vol. 57 of International Series in Operations Research & Management Science, pp. 145–184. Springer New York, 2003.

73– N. Mladenovic, P. Hanson: Variable neighborhood search. Computers and Operations Research, vol. 24, no. 11: pp.1097–1100, November 1997.

74– C. Voudouris, E. P. K. Tsang: Guided Local Search. Handbook of Metaheuristics, vol. 57 of International Series in Operations Research & Management Science, pp 185-218. Springer New York, 2003.

75– C. Voudouris and E. P. K. Tsang, Guided local search joins the elite in discrete optimization. In Proceedings of the DIMACS Workshop on Constraint Programming and Large Scale Discrete Optimisation, pp. 29-40, 1998.

76– C. Voudouris, E. Tsang: Partial constraint satisfaction problems and guided local search. In Proceedings of (PACT'96) Second International Conference on Practical Application of Constraint Technology. London, April 1996, pp. 337-356.

77– H. R. Lourenço, O. Martin, and T. Stützle: Iterated local search. In F. Glover, and G. Kochenberger, editors, Handbook of Metaheuristics, vol. 57 of International Series in Operations Research & Management Science, pp. 321-353. Kluwer Academic Publishers, Norwell, MA, 2002.

78– H. R. Lourenço, O. Martin, and T. Stützle: A beginner's introduction to iterated local search. In Proceedings of 4th Metaheuristics International Conference, Porto, Portugal, pages 1–11, 2001.

79– T. Stützle: Local Search Algorithms for Combinatorial Problems - Analysis, Improvements, and New Applications. PhD thesis, Darmstadt University of Technology, Department of Computer Science. 1999.

80– K. Katayama, H. Narihisa: Iterated local search approach using genetic transformation to the traveling salesman problem. In: Proceedings of GECCO'99, vol. 1. Morgan Kaufmann, pp. 321–328. 1999.

81– T. Stützle: Iterated local search for the quadratic assignment problem. European Journal of Operational Research, vol. 174, no. 3: pp. 1519-1539, 2006.

82– S.Das, P.N. Suganthan: Differential Evolution: A Survey of the State-of-the-Art. IEEE Transactions on Evolutionary Computation, vol. 15, no. 1, pp. 4-31, Feb. 2011.

83– F. Glover, M. Laguna and R. Martí: Scatter Search. In Advances in Evolutionary Computation: Theory and Applications, A. Ghosh and S. Tsutsui Eds. New York: Springer-Verlag, 2003, pp. 519-537.

84– M.G.C. Resende, C.C. Ribeiro: Greedy randomized adaptive search procedures. In F. Glover and G. Kochenberger, editors, Handbook of Metaheuristics, pp. 219–249, Kluwer Academic Publishers, 2003.

85– P. Festa, M.G.C. Resende: GRASP: An annotated bibliography. In C.C. Ribeiro and P. Hansen, editors, Essays and Surveys on Metaheuristics, pp. 325–367, Kluwer Academic Publishers, 2002.

86– P. Demestichas, N. Georgantas, E. Tzifa, V. Demesticha, M. Striki, M. Kilanioti and M. Theologou: Computationally efficient algorithms for location area planning in future cellular systems. Computer Communications, vol. 23, no. 13, pp. 1263–1280, 2000.

87– J. Taheri, A. Y. Zomaya: Clustering techniques for dynamic mobility management. In Proceedings of 4th ACM International Workshop on Mobility Management and Wireless Access, 2006, pp. 10-17.

88– I. Demrikol, C. Ersoy, M.U. Caglayan, H. Delic: Location area planning in cellular networks using simulated annealing. In Proceedings of IEEE INFOCOM'2001, April 2001, pp. 566–602.

89– J. Taheri, A. Y. Zomaya: The Use of a Hopfield Neural Network in Solving the Mobility Management Problem. In Proceedings of The IEEE/ACS International Conference on Pervasive Services, 2004, pp. 141–150.

90– 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.

91– R. Subrata, A. Zomaya: A comparison of three artificial life techniques for reporting cell planning in mobile computing. IEEE Trans. on Parallel and Distributed Systems, vol. 14, no. 2, 2003, pp. 142-153.

92– J. Taheri, A. Y. Zomaya: A modified Hopfield network for mobility management. Wireless Communications and Mobile Computing, vol. 8, no. 3, John Wiley & Sons, 2008, pp. 355-367.

93– C. A. Baburaj, K. Alagarsamy: A review on various network results for reporting cell planning in genetic algorithm technique. International Journal of Engineering Science and Technology, vol. 2, no. 9, 2010, pp. 4088-4094.

94– C. A. Baburaj, S. T. Kannan: Effective and Efficient Mining of Data in Mobile Computing. IAENG International Journal of Computer Science, vol. 32, no. 4, 2006, IJCS_32_4_5.

95– F. Mehta, P. Swadas: A Simulated Annealing Approach to Reporting Cell Planning Problem of Mobile Location Management. International Journal of Recent Trends in Engineering, vol. 2, no. 2, November 2009, pp. 98-102.

96– A. Hac, X. Zhou: Location Strategies for Personal Communication Networks: A Novel Tracking Strategy. IEEE Journal on Selected Areas in Communications, vol. 15, no. 8, October 1997, pp. 1425-1436.

97– S. M. Almeida-Luz, M. A. Vega-Rodríguez, J. A. Gómez-Pulido and J. M. Sánchez-Pérez: Solving the location area problem by using differential evolution. Journal of Communications Software and Systems, vol. 4, no. 2, pp. 131-141, 2008.

98– 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 J.J.M. Neves (Ed.), New Trends in Artificial Intelligence APPIA, Associação Portuguesa para a Inteligência Artificial, 2007, pp. 219-230.

99– 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, 2007, pp. 1-5.

100– S. M. 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 2008), IEEE Computer Society, Oct. 2008, pp. 170-175.

101– EGEE Web, http://www.eu-egee.org (2010); http://www.stfc.ac.uk/e-Science/projects/long-term/22409.aspx (2014)

102– EELA Web, http://www.eu-eela.eu (2014)

103– S. M. Almeida-Luz, M. A. Vega-Rodríguez, J. A. Gómez-Pulido, J. M. Sánchez-Pérez: Differential Evolution for Solving the Mobile Location Management. Applied Soft Computing, Volume 11, Issue 1, Elsevier Science, Amsterdam, Netherlands, 2011, pag:410-427, ISSN:1568-4946.

104– J. Taheri, A.Y. 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.

105– S. M. Almeida-Luz, M. M. Rodríguez-Hermoso, M. A. Vega-Rodríguez, J. A. Gómez-Pulido, J. M. Sánchez-Pérez: GRASP and Grid Computing to Solve the Location Area Problem. In Proceedings of the World Congress on Nature and Biologically Inspired Computing, IEEE Computer Society, Coimbatore, India, 2009, pag:164-169.

106– Framework Visual Studio 20XX, Microsoft Development Network (on July 2014): http://msdn.microsoft.com/en-us/vstudio//default.aspx

107– C# language, Microsoft Development Network (on July 2014): http://msdn.microsoft.com/en-us/vstudio/hh341490

108– Microsoft Office Excel, Microsoft Corporation (on July 2014): http://office.microsoft.com/en-us/excel/

109– Microsoft Windows Notepad, Microsoft Corporation (on July 2014) http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/app_notepad.mspx

Appendix A – Software Development

In this section we will present the methodology and tools used to develop the application, and to treat the results obtained. It also will be explained the application developed and its adjustments to the implemented algorithms.

A.1. Tools and Languages

To develop the application was chosen the framework Visual Studio 2005, which was later updated for the most recent versions Visual Studio 2013, and the language C#. To treat, and graphically represent, the results obtained in way to compare them, the Microsoft Excel has been used and to prepare the input data the Microsoft Windows Notepad has been used.

The framework Visual Studio 2005/20XX was developed by Microsoft [106] and permits develop applications with diverse types of languages as Visual Basic, J#, C++ and C# (among others).

This framework represents a very inclusive and flexible development environment that allows the development of robust applications, based on operating system Windows or in Web applications. It also has a good online help – Microsoft Development Network (MSDN), which is very useful when some development problems appear. In our case a Windows application was developed and applied to the problem.

The language chosen to work in the framework Visual Studio 2005 was C# (C sharp) [107] because it is an object oriented language that uses a unified system of types, which means that all is viewed as an object. This language also has a direct support to component based programming as properties, events and attributes that will permit easily define and project the characteristics of our application.

Microsoft also provides online help and support using the MSDN, which is very useful during the C# programming, because it has a lot of examples and related links.

Microsoft Office Excel [108] tool is a calculus spreadsheet that allows analysing, sharing and managing information with the objective of taking more supported conclusions.

In our case the Microsoft Office Excel has been used to treat data of solution results making graphics that allows a better analyses of the solutions obtained.

Microsoft Windows Notepad [109] is a simple text editor program that comes included with all versions of Microsoft Windows. This program can be used just for creating and manipulating simple text documents or for creating Web pages. In our case it is used to create and manipulate the text files used for introducing network data in the application.

A.2. Developed Application for LA problem

In this section and subsections we will explain the developed software application and the changes for which case. For a better understanding of the choices made for definition of the application it is also explained the algorithms definition, but algorithm implementation details were already presented in chapters 5 and 6, in each respective section. Initially it was developed an application directed to the use and manipulation of differential evolution but then because of the decisions and additions in our investigation work, this application was changed to also accommodate the use of scatter search algorithm.

A.2.1 Software Application to the Use of Differential Evolution

In the initial phase of our investigation we were principally focused on applying Differential Evolution to the Location Area problem. So we decided to develop a software application that could allow us to manipulate network data applying the differential evolution algorithm and also maintain the capacity of illustrating algorithm application evolution and respective solutions.

A.2.1.1 Differential Evolution Definition

In order to work with differential evolution it is necessary to use the problem information and available network data to make the inputs to the application and it is also necessary to define what values the operators can take. Considering that differential evolution may be applied in ten different schemes, it is fundamental to define what of them will be available and how they will be used.

Considering that more parameters besides LA are associated with the definition of a chromosome to represent the location area problem [7, 20], the chromosome definition will be different that the one used with the genetic algorithms. As it is possible to see in Figure 42, for each gene (that represents a cell) will be considered the number of users that enters in the cell, the number of calls received and the neighbours' information (that includes the list of neighbours of this cell and respective number of entering users).

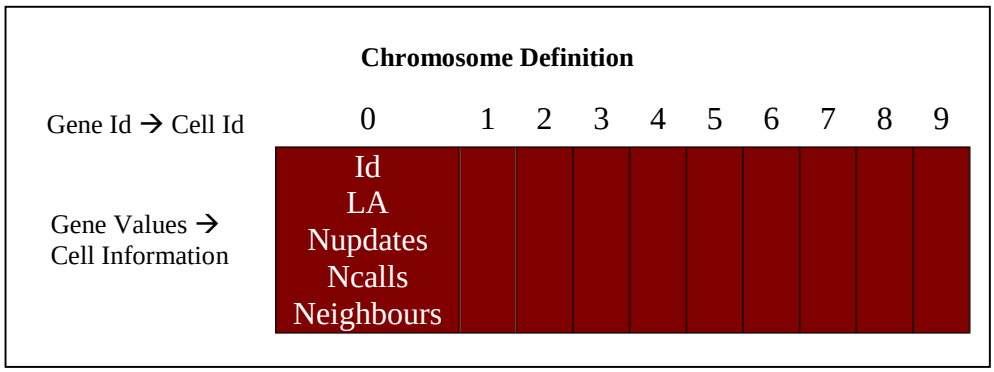


Figure 42 – Differential Evolution Chromosome Definition for LA Problem

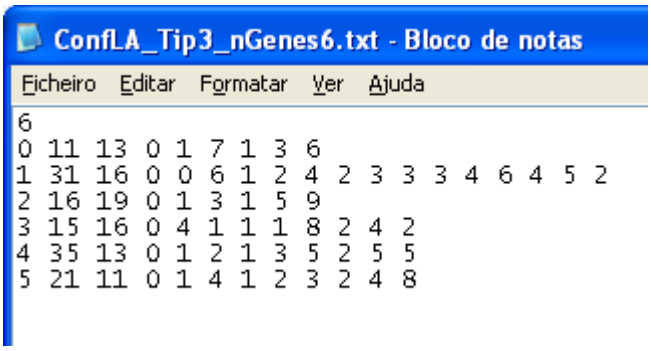
Also with differential evolution is necessary to define the initial values to the population formulation and the stop criteria to be used. These parameters have attribution by omission, but we need to consider that they must be dynamic, which means that the application should allow that also those values could be defined/changed before starting each run of the algorithm.

To the actual definition, according with problem information, it is considered that the size of a chromosome and the content of each gene depend on the input data and cannot be changed by the user, just manipulated by the algorithm. The initial number of

LAs is two, and will be changed according with the action of the algorithm, and its maximum value is limited to the number of cells that compound the chromosome. Each chromosome will correspond to an individual and the initial population must have at least ten individuals, but the omission value will be twenty individuals. The number of generations may be between one and ten thousand, but the omission value will be one thousand.

The differential evolution operators are crossover and mutation and they have associated a probability value of being applied. Relatively to the schemes that will be used, initially just *best1exp* and *rand1exp* were available, but in this moment all the ten schemes can be used.

All the omission values are kept in one structure that must be actualized every time that those parameters are changed. The input values related with the problem are in a text file, but its format is not as simple as the initial one (see Figure 43).


ConfLA_Tip3_nGenes6.txt - Bloco de notas

Ficheiro Editar Formatar Ver Ajuda

```
6
0 11 13 0 1 7 1 3 6
1 31 16 0 0 6 1 2 4 2 3 3 3 4 6 4 5 2
2 16 19 0 1 3 1 5 9
3 15 16 0 4 1 1 1 8 2 4 2
4 35 13 0 1 2 1 3 5 2 5 5
5 21 11 0 1 4 1 2 3 2 4 8
```

Figure 43 – DE Input Text File Format

In Figure 43, the first line indicates the number of cells to be considered and the other ones correspond to the values that compound the cell information, where the first value represents the cell id, the second one corresponds to the number of incoming users, the third is the number of incoming calls and the other correspond to the neighbours' information.

A.2.1.2 Software Application Layout

This new version of the application may look simpler because it does not involve so many parameters as the initial one which we have tested with genetic algorithms. But the application must continue supporting the definition and/or change of parameters values, differential evolution operators and also should permit to present the evolution of obtained results.

In Figure 44 it is possible to see the layout of the new version of the software application developed. The input data (or change of data) continues to be introduced in a text file through the menu *Ficheiro*.

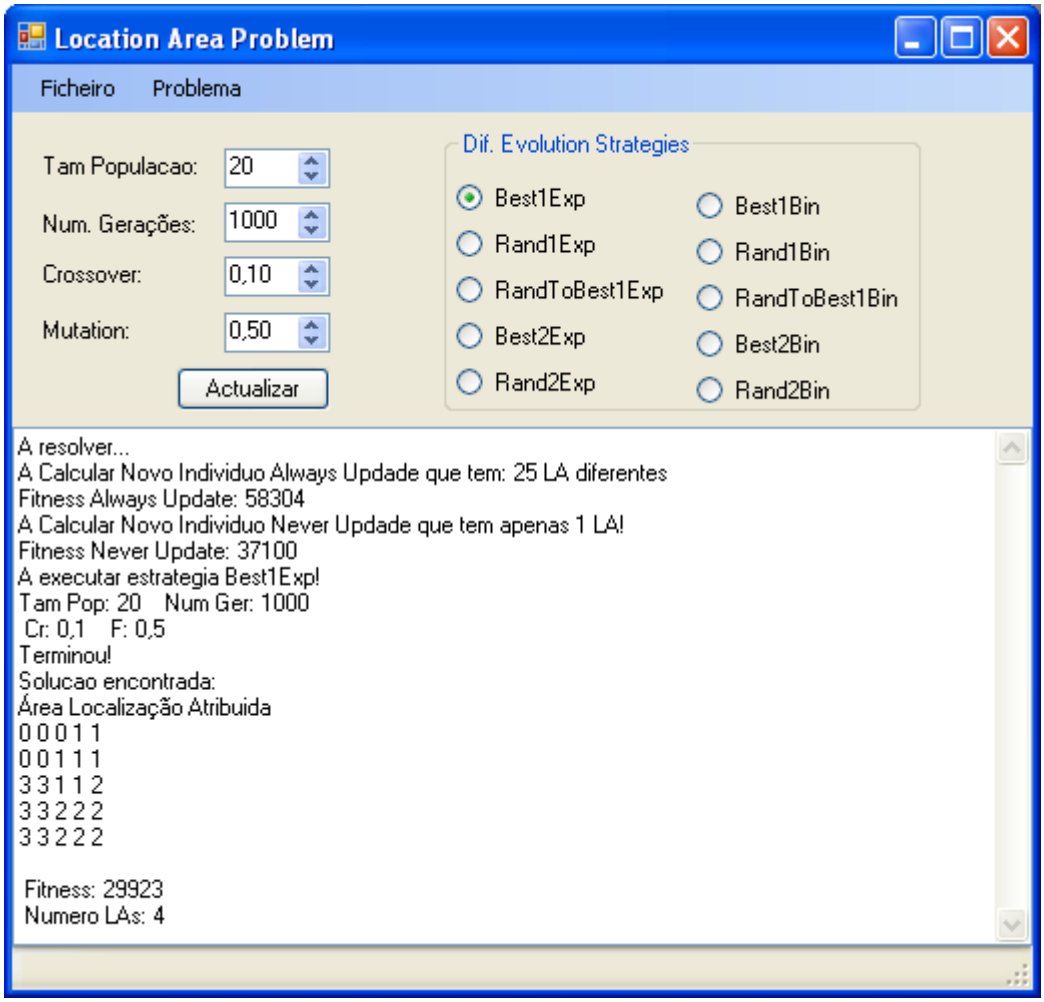


Figure 44 – DE Software Application Layout for the LA problem

In the left side of the interface is possible to see/define the values of:

- *Tam População* – population size which permits define/change the number of individuals (potential solutions) that compound the population;
- *Num. Gerações* – total number of generations that will correspond to the stop criterion when the algorithm is running;
- *Crossover* – crossover probability, which is the probability value associated to the application of crossover in a chromosome;
- *Mutation* – mutation probability that corresponds to the value associated with the application of mutation in a chromosome.

If we change some of these values we must update the associated parameters by selecting the *Actualizar* button, before starting running the algorithm. If we do not make this update selection, the changes will not be effective and the values will be discarded.

The right side of the application is used to choose what differential evolution scheme will be applied. Normally the algorithm will be applied with just one differential evolution scheme, but it is also possible to choose applying the algorithm with all the schemes and present the final solutions, by choosing the option *Resolver em Série* as it is possible to see in Figure 45.

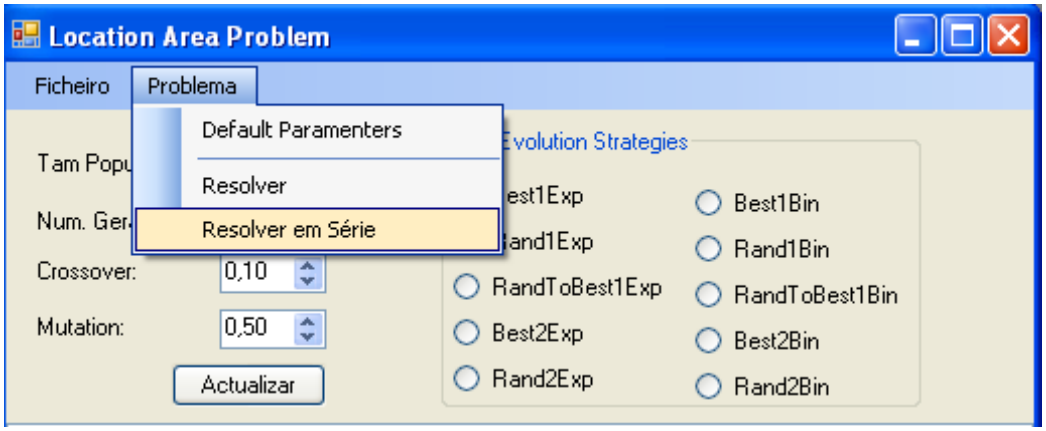


Figure 45 – DE Resolution in Series Option

And finally in the bottom side of the application (see Figure 44) the evolution of the algorithm and the final solution are presented. It starts presenting the results for the

application of always-update and never-update classical strategies, then it indicates the used parameters and finally the best solution found.

A.2.2 Software Application to the Use of Scatter Search

Considering that after the DE based approach we decided to investigate and implement a Scatter Search (SS) based approach, it was necessary to adjust the development of the software application to support it.

We needed to consider the scatter search specificities and maintaining the possibility of the software developed to be able to illustrate the algorithm evolution and showing the solutions achieved.

A.2.2.1 Scatter Search Definition

Like it was for the DE algorithm, also to work with scatter search algorithm it is necessary to consider the problem information and available network data to make the inputs to the application. It is also necessary that the software application allows us to define what values the operators can take, assuring that only valid values can be chosen.

The chromosome definition will be the same which was defined to use with the differential evolution. Concerning this, for each gene, (which represents a cell), will be considered the number of users that enter in the cell, the number of calls received and the neighbours' information (corresponding to the list of neighbours of this cell and respective number of entering users).

Considering the initial characteristics of this application, it still is possible the definition or variation of initial values for the population formulation and the respective stop criterion that is used, before starting each run of the algorithm.

Like for the DE algorithm, also for the application of the SS algorithm the size of a chromosome and the content of each gene depend on the input data, which cannot be changed by the user, only manipulated by the algorithm. The initial number of LAs is defined to two, and will change along the application of the algorithm, considering that

its maximum value is limited to the number of cells that compound the chromosome. Each chromosome corresponds to an individual and the initial population must have at least ten individuals, but the omission value is set to twenty individuals. The number of generations may be between one and ten thousand, but the omission value is defined as one thousand.

The core parameters of scatter search are the refset size, which is obtained by summing the size of the quality subset and the size of the diversity subset; the number of local searches (LS) to be considered in the improvement method; and the value of the combination probability used in the combination process.

The input values related with the problem are in a text file, and these input files are the same ones which were used with differential evolution. In Figure 43 it is exposed one example of these files' structure.

A.2.2.2 Software Application Layout

In order to allow the use and implementation of the Scatter Search algorithm, the application layout had to be adapted to accomplish these changes.

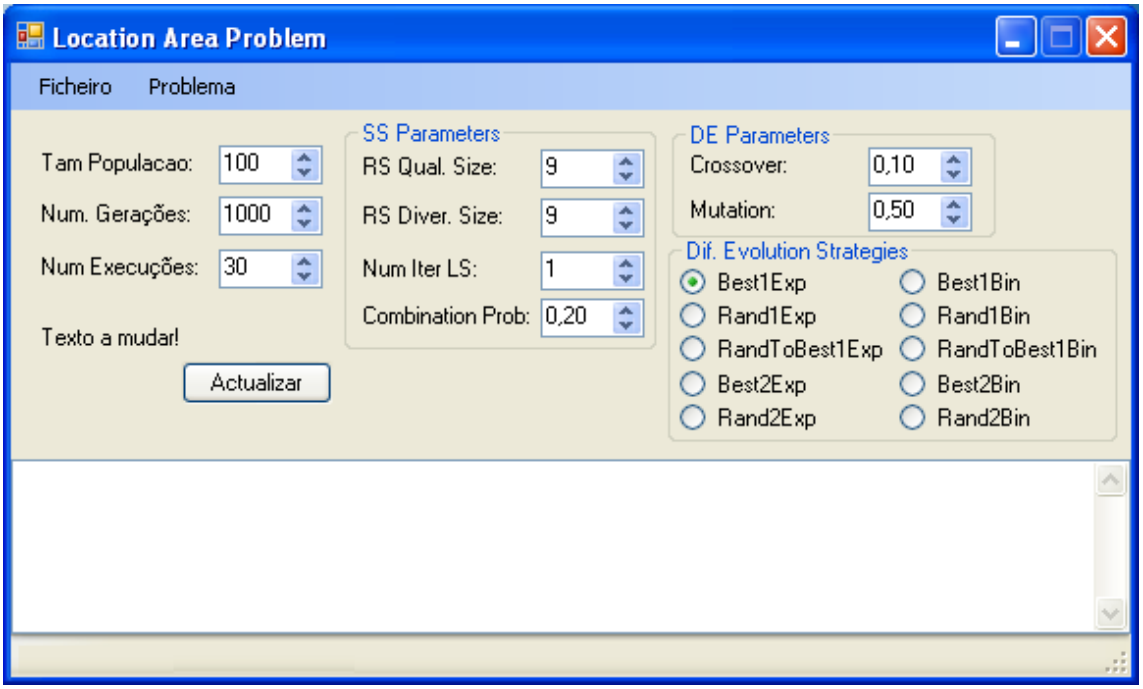


Figure 46 – DE and SS Software Application Layout, for the LA problem

The application still supports the definition or change of initial values, but now it also allows the definition or modification of scatter search parameters.

In Figure 46, we present the layout of the new version of this software application where the input data still are introduced in a text file through the menu called "*Ficheiro*".

With these improvements we have a new "section" for the scatter search parameters, where it is possible to define/change the values of:

- *RS Qual. Size* – reference set quality size which allows the definition/change of the number of quality solutions in the reference set size;
- *RS Diver. Size* – reference set diversity size, which corresponds to the number of most diverse solutions that compound the reference set;
- *Number Iter LS.* – number of iterations of local search used in the improvement method;
- *Combination Prob* – corresponds to the probability of combination (crossover) used in the solution combination method, which combines the solutions in each subset generated.

In the bottom side of the application we continue presenting the evolution of the algorithm and the final solution obtained, like it was done with the DE algorithm and considering that this application is now prepared to work with both DE and SS based approaches.

A.3. Developed Application for RC problem

The application that we have developed to the LA focused approaches is the same one that we have used to implement the approaches intended to solve the Reporting Cells (RC) problem.

The main consideration that we took is the different characterization of the chromosome. In the RC strategy the neighbours' information of each cell is not considered. Due to this, in this strategy and for each gene (which represents a cell as shown in Figure 47), we need to know the cell id (*Id*), the indication if it corresponds to a reporting cell or a non-reporting cell (*RC*), the number of users that enter in the cell

(*Nupdates*), the number of incoming calls and finally the vicinity factor (*vF*) which will be calculated, considering the configuration of the network.

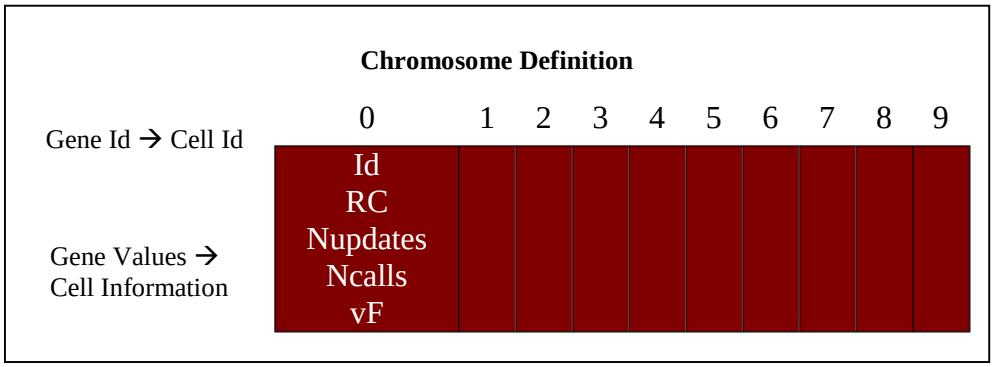


Figure 47 – Chromosome Definition for the RC Problem

Also in the RC based approach, like for the LA based approaches, in the actual definition, according with problem information, it is considered that the size of a chromosome and the content of each gene depend on the input data and cannot be changed by the user, just manipulated by the respective algorithm.

Initially, for each cell, it will be attributed a reporting cell or a non-reporting cell with a probability of fifty percent. Each chromosome will correspond to an individual and the initial population must have at least ten individuals, considering an omission value of twenty individuals. The number of generations may be defined between one and ten thousand, but the omission value will be one thousand.

All the omission values, including the respective parameters of each algorithm implementation, are kept in one structure that must be actualized every time that those parameters are changed.