A Differential Evolution Based Algorithm to Optimize the Radio Network Design Problem

Silvio Priem Mendes[1], Juan A. Gómez Pulido[2], Miguel A. Vega Rodríguez[2], María D. Jaraíz Simón[2], Juan M. Sánchez Pérez[2]
(1) Polytechnic Institute of Leiria, High School of Technology, Leiria, Portugal
smendes@estg.ipleiria.pt
(2) University of Extremadura, Polytechnic School, Cáceres, Spain
jangomez@unex.es

Abstract

In this paper we present a Differential Evolution based algorithm used to solve the Radio Network Design (RND) problem. This problem consists in determining the optimal locations for base station transmitters in order to get a maximum coverage area with a minimum number of transmitters. Because of the very high amount of possible solutions, this problem is suitable to be tackled with evolutionary techniques, so in our work it has been developed an algorithm inspired on the well-known Differential Evolution algorithm, obtaining good results.

1. Introduction

The Radio Network Design (RND) problem is a study area in the cellular wireless technology domain. This field of telecommunications is very important due to the increasing number of user services. Together with other problems like assigning frequencies in radio link communications, developing error correcting codes for transmission of messages, designing the telecommunication network, etc, the RND problem belongs to this kind of network problems that need be studied as NP-hard optimization problems.

In a few words, the RND problem consists in minimizing the number and locations of transmission antennae to cover a maximum area in order to give service to the highest possible amount of terminals.

The paper is organized as follows. In the second section we define and characterize the radio network design problem. In Section 3 we will briefly describe the Differential Evolution based algorithm we have developed to adapt this evolutionary algorithm to the RND problem. In Section 4 we will provide the results of the tests performed to study the efficiency of the algorithm. Finally, some concluding remarks and future research lines are drawn in Section 5.

2. The RND problem

A base station (BS) transmitter is a radio signal transmitting device, with a determined type of coverage. In this work we consider two possible types: square and circle coverage. Other types will be considered in future works, like sector-directional coverage. In the figure 1 we can see the considered coverage models for BS transmitters.

We consider a digitalized model of the terrain. The area is divided in sectors or locations (atomic bits of terrain). In this work we use a rectangular area modeled by a grid. Each coordinate (x,y) of the grid represents a possible BS transmitter location. In the figure 2 we want to represent this model, showing how three BS transmitters are located in available coordinates of the grid (a) and how some locations are under the influence of these BS with different coverage degrees (b).

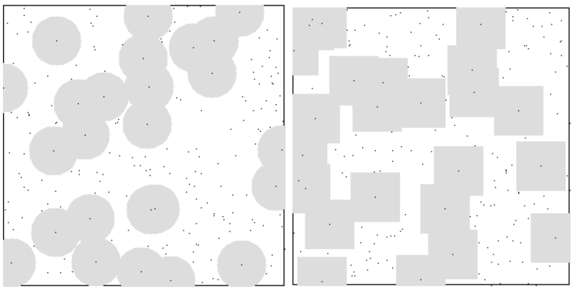


Figure 1. Base station transmitters with circle and square coverage

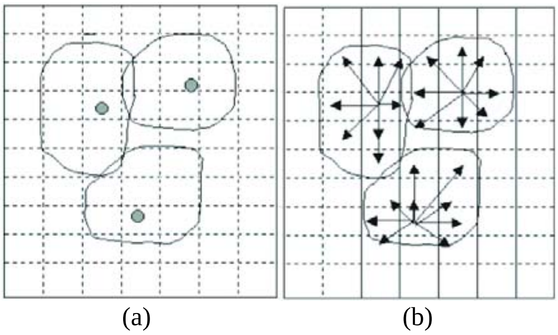


Figure 2. (a) The terrain is digitalized by means of a grid. (b) Three BS locations are shown, together with the locations under their influence

If two or more BS transmitters are enough near one another, their influence areas are overlapped, and then the locations inside these areas can have different degrees of coverage (for example, one location can be under the influence of two BS transmitters and other location can be inside the area of only one transmitter; in this case the second location has a lower level of intensity of the received signal). For this reason, the information stored in every position of the grid must take into account the following data:

- The degree of coverage.
- If it is an available location for a transmitter.
- If BS transmitter is located or not.
- The kind of transmitter.

In this way, we can establish the fitness function (F) to measure the efficiency of a BS set disposed in a determined manner in the grid. F is described [1] by means of the cover rate measure and the amount of BS transmitters used (1):

$$F = \frac{(cover\ rate)^2}{used\ transmitters} \qquad (1)$$

The measure of the fitness is important in the algorithms performed in the optimization problem. In the figure 3 we show a pseudo code that evaluates F.

The work to do is to build a BS network giving the maximum coverage to an area. First, in the cell planning design, we have to determine the set of available locations for the BS. Then, the objective is to obtain a high percentage of the area covered using the lowest amount possible of BS. This must be done in an efficient manner.

The problem has one important constraint: the list of available location sites. BS can only be placed in a predefined set of available locations (sectors). For example, antennas can be placed in some roofs, and they can not be placed in public recreation areas, etc.

This set of available locations is modeled with an array of coordinates of the grid. Then, the size of the array is the size of the problem instance.

```
Initialize grid (set grid positions to ‘0’)
for(all the available locations)
   if(transmitter is placed)
      (x,y)=location_coordinates;
      Count one more transmitter
      Mark the transmitter
      for(all sectors belonging to transmitter
                                 coverage (x,y))
         (x1,y1)=sector_coordinates;
         if( there is no BS in (x1,y1))
            Increase the coverage
            if( new coverage in (x1,y1))
               increase covered points
cover rate=(100*covered points)/GRID SIZE
fitness=(cover rate*cover rate)/used trans
```

Figure 3. Pseudocode of the fitness function

This is a typical NP-hard combinatorial optimization problem. For this reason we’ve wanted to try an evolution based algorithm, because algorithms like genetic algorithms (in sequential and parallel implementations) and other evolutionary techniques have been employed with success in some works [1][2][3].

3. A Differential Evolution based algorithm

Differential Evolution (DE) is an algorithm created by Ken Price and Rainer Storn [4]. From 1994, DE has been used for many optimization problems, with satisfactory results. This is the reason to use it in our research, with the goal of comparing it with other works. DE code is available [5] to the researchers.

DE is a very simple population based stochastic function minimizer/maximizer, used in a wide range of optimization problems, including multi-objective optimization [6]. It has been modified in this work to be adapted because the grid structure of the problem is a binary description.

3.1. Algorithm structure

The following points describe the algorithm performance (figure 4). This is an iterative algorithm, where the successive generations try to get an optimal solution, stopping when the maximum number of generations is reached or when the fitness of the current solution is greater than a predetermined value.

1. Establish the parent population size (NP individuals). Each individual is a BS set, for example randomly generated, with its fitness (also named cost) function evaluated.

2. Choose target individual. For example, we can choose the first individual (0) to be always targeted.
3. Choose randomly 3 population members (for example, individuals with indexes ia, ib, ic).
4. Build weighted difference vector with individuals ia and ib. This is performed by means of a determined crossover function.
5. Add the third randomly chosen individual (ic) to the last result. Also, this is done with a crossover function (it could be a different crossover function that the used before).
6. Do crossover with target individual to get trial individual.
7. Greater fitness value survives into next generation. Then, the offspring population is the parent population where the target individual is substituted with the trial individual if its fitness value is greater than the target fitness value. Then, the next generation begins in the above second point.

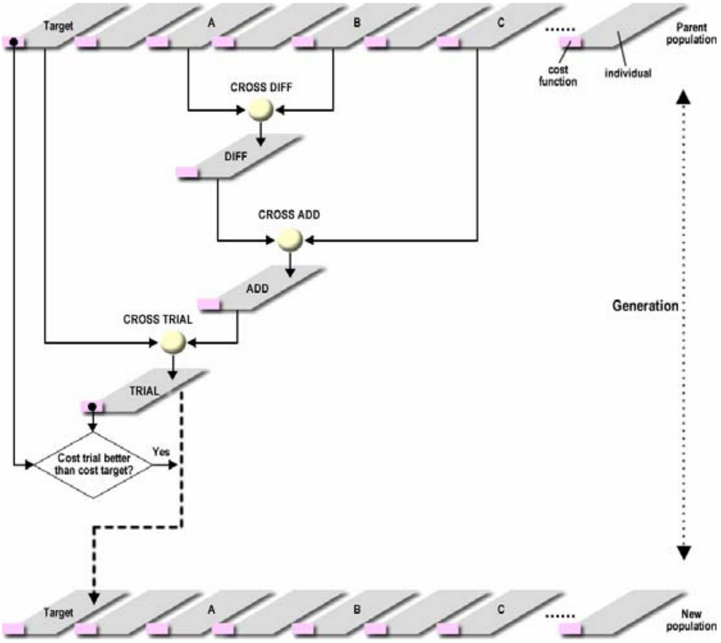


Figure 4. The Differential Evolution based algorithm scheme considered for the RND problem

3.2. Crossover functions

The crossover function is very important in any evolutionary algorithm. It also should be noted that there are evolutionary algorithms that use mutation as their primary search tool as opposed to crossover operators. In our case we focus our effort in creating and choosing carefully the crossover function in order to obtain better results. In this work we have started developing two simple functions, whose characteristics are explained as follows: Let be two set of BS locations (individuals), named A and B (the parents). Let be the S individual (the offspring) obtained from the application of the crossover function to A and B. We consider two possibilities for the crossover:

- Crossover FA/2+FB/2+SA/2 (shortly named FA). Choose the first half of A to build the first half of the offspring. The second half of the offspring is then built with the first half of B, but if a repeated location appears, successive locations of the second halves of B and A are taken (see figure 5).
- Crossover SA/2+SB/2+FA/2 (shortly named SA). Choose the second half of A to build the first half of the offspring. The second half of the offspring is then built with the second half of B, but if a repeated location appears, successive locations of the first halves of B and A are taken (see figure 6).

Any crossover function must satisfy two important conditions:

1. All the BS locations in the crossed individual must be available locations in the grid.
2. It can not be two or more repeated locations. All locations in any individual must have different coordinates.

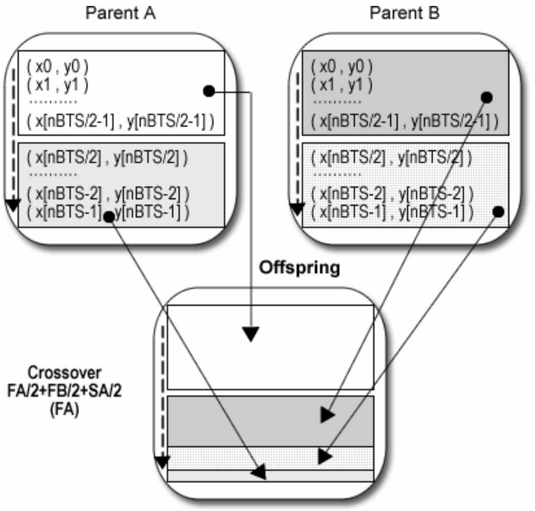

Parent A
Parent B
(x0 , y0)
(x1 , y1)
....
(x[nBTS/2-1] , y[nBTS/2-1])
(x[nBTS/2] , y[nBTS/2])
(x[nBTS-2] , y[nBTS-2])
(x[nBTS-1] , y[nBTS-1])
(x0 , y0)
(x1 , y1)
....
(x[nBTS/2-1] , y[nBTS/2-1])
(x[nBTS/2] , y[nBTS/2])
(x[nBTS-2] , y[nBTS-2])
(x[nBTS-1] , y[nBTS-1])
Offspring
Crossover
FA/2+FB/2+SA/2
(FA)

Figure 5. Crossover function FA

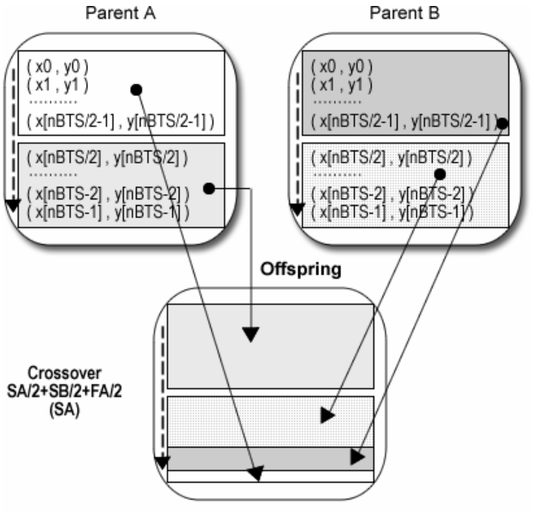

Parent A
Parent B
(x0 , y0)
(x1 , y1)
....
(x[nBTS/2-1] , y[nBTS/2-1])
(x[nBTS/2] , y[nBTS/2])
(x[nBTS-2] , y[nBTS-2])
(x[nBTS-1] , y[nBTS-1])
(x0 , y0)
(x1 , y1)
....
(x[nBTS/2-1] , y[nBTS/2-1])
(x[nBTS/2] , y[nBTS/2])
(x[nBTS-2] , y[nBTS-2])
(x[nBTS-1] , y[nBTS-1])
Offspring
Crossover
SA/2+SB/2+FA/2
(SA)

Figure 6. Crossover function SA

4. Experimental results

We can perform experiments with a high degree of variability because the developed tool allows selecting the value of many parameters. However, we establish the following common values:

- Size of the grid: 287*287 locations.
- Maximum number of location sites: 349.
- BS types: square or circle.

In the following subsections we show several experimental aspects in the testing of our algorithm.

4.1. RNDWin

RNDWin is the application developed to perform the experiments with the Differential Evolution based algorithm. Future versions of this software will have other optimization algorithms built inside, such as Genetic algorithms and Parallel Genetic Algorithms. This application runs under Windows operating system (Win98 / ME / XP / 2000 / 2003).

4.2. BS predefined locations

In order to compare the results we've obtained from the application, with other optimization methods and techniques, it is necessary to apply the experiments on a predefined set of BS available locations. With this purpose we have built four predefined sets.

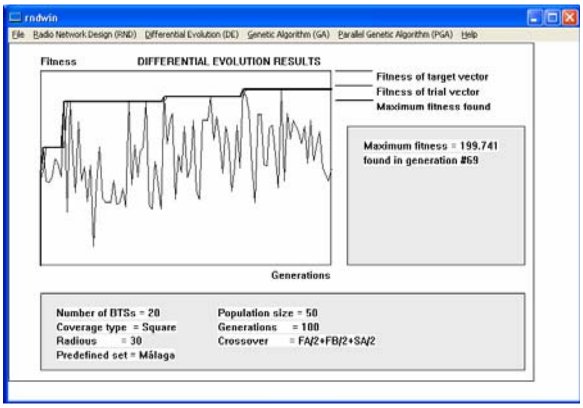
4.3. DE tests looking for the maximum coverage for a considered number of BS

We have programmed two types of experiments. In the first type, we try to find the optimal set of locations for a fixed amount of BS transmitters. The parameters of the evolutionary algorithm to be initially selected are: population size, crossover function and maximum number of generations.

In the figure 7 we can observe how the fitness value is decreasing while the generations are running. The final fitness value (corresponding with the last target individual) can be reached quite before the algorithm arrives to its end. Obviously, if we have more BS to be located, the final fitness will be better, but this is not the objective of this experiment.

4.4. DE tests looking for the minimum number of BS for a wanted coverage area

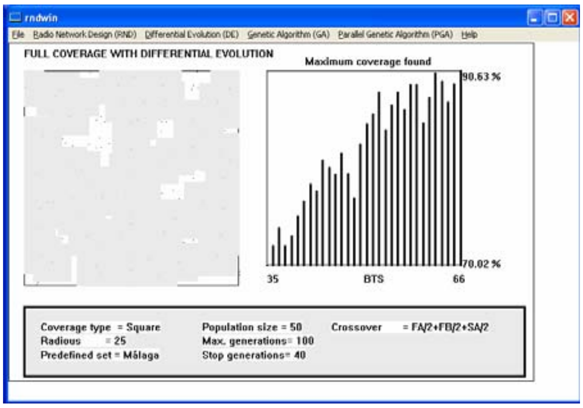
The second type of experiments tries to look for the minimum number of transmitters to be located in the grid in order to reach a determined amount of cover rate. This experiment is also performed with the Differential Evolution based algorithm, where a programmed loop goes through from a starting amount of transmitters to a final amount for which the wanted coverage is obtained (see figure 8). For every number of BS in this loop, DE is applied as in the first type of experiments above explained.


rndwin
File Radio Network Design (RND) Differential Evolution (DE) Genetic Algorithm (GA) Parallel Genetic Algorithm (PGA) Help
Fitness
DIFFERENTIAL EVOLUTION RESULTS
Fitness of target vector
Fitness of trial vector
Maximum fitness found
Generations

```
Maximum fitness = 199.741
found in generation #69

Number of BTSs = 20          Population size = 50
Coverage type  = Square      Generations     = 100
Radious        = 30          Crossover       = FA/2+FB/2+SA/2
Predefined set = Málaga
```

Figure 7. DE tests looking for the maximum coverage for a considered number of BS


rndwin
File Radio Network Design (RND) Differential Evolution (DE) Genetic Algorithm (GA) Parallel Genetic Algorithm (PGA) Help
FULL COVERAGE WITH DIFFERENTIAL EVOLUTION
Maximum coverage found
90.63 %
70.02 %
35
BTS
66

```
Coverage type  = Square      Population size = 50       Crossover    = FA/2+FB/2+SA/2
Radious        = 25          Max. generations= 100
Predefined set = Málaga      Stop generations= 40
```

Figure 8. DE tests looking for the minimum number of BS for a wanted coverage area

4.5. Efficiency measures of DE capabilities

Finally, we present the most important experimental task, because it tells us about the efficiency of the algorithm in comparison with randomly generated sets of BS locations. The experiments we present now also inform us what crossover function offers the best precision.

We have performed 10 DE tests (as explained in subsection 4.3) with the following common parameters:

- Number of BS transmitters: 50
- Coverage type: square
- Radius of coverage: 30
- Predefined available locations set: malaga.dat
- NP: 100 individuals
- Maximum number of generations*:* 1000
- Crossover function: FA

For every test we have annotated the maximum fitness found and the last generation for which this maximum was reached. Then we calculate the average of these values for the 10 experiments. These tests were repeated for the other crossover function (SA).

In order to compare the effectiveness of DE, we have also performed 10 experiments where BS sets were randomly generated and their fitness evaluated. The average of these fitness values was compared with the obtained in the last described DE tests. These results are graphically shown in the figure 9. We can see how DE methodology is better than a random search. Also, we can observe how the FA crossover offers better fitness results than the SA crossover, at the same time these results are obtained in a lesser number of generations.

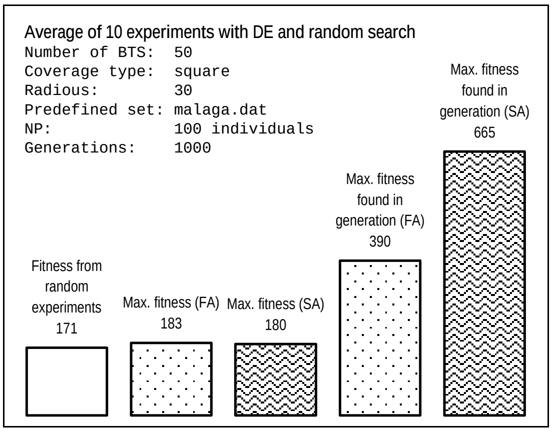


Figure 9. Experimental results where DE is compared with solutions randomly generated

5. A distributed architecture to perform massive calculations

At present we are using the software system BOINC [7] (Berkeley Open Infrastructure for Network Computing) in order to perform massive computations/experiments of the RND problem in a parallel way. BOINC is a system for “Volunteer Computing” (also called “Public-Resource Computing”, “Peer-to-Peer Computing” or “Global Computing”). Volunteer computing [8] uses computers volunteered by the general public to do distributed scientific computing. Volunteer computing is being used in high-energy physics, molecular biology, medicine, astrophysics, climate study, and other areas. These projects have attained unprecedented computing power. For example, SETI@home has sustained a processing rate of about 60 TeraFLOPS for several years [9].

In our case, we use BOINC in order to perform many different executions of our evolutionary algorithm in parallel. In this way, we can do a deep survey about which are the best parameters and combinations for solving the RND problem. People interested in learning more about our platform RND-BOINC, or that want to join in this project (RND@home), can access it via the website [10].

6. Conclusions and future works

In this work we have tried to perform a first approximation to the optimization of the Radio Network Design problem by means of a special type of evolutionary algorithm, based on Differential Evolution.

From the modeling point of view, we have seen this algorithm is capable of tackling the RND problem from, carefully built, determined crossover functions.

The results we show in this paper prove that it is possible to obtain a satisfactory locations set to cover a maximum area for a determined number of base transmitting stations, in comparison with a randomly generated set. These results encourage us to follow working on the code.

It is possible the DE algorithm might become biased towards either the cover rate or used transmitters terms in the fitness function if one of these values rapidly and radically improved. For this and other reasons could be interesting to apply an inherently multi-objective algorithm (e.g. NSGA-II, SPEA, MOGA) to reveal the relationship between these parameters and to allow for a more detailed analysis.

We observe that the developed code needs more enhancements, like new and more efficient crossover functions, for example. So, we hope that a carefully written enhanced code will produce better results than those shown in this paper. Also, it is needed to introduce new BS coverage types, like sector or directional antennas. Other interesting experience could be applied to the problem considering BS locations from real city maps.

Finally, the results found with our algorithm must be compared with those obtained from other evolutionary algorithms, such as genetic algorithms, parallel genetic algorithms, etc. We believe this work is quite necessary for successful further research.

7. Acknowledgment

This work has been developed thanks to OPLINK project (TIN2005-04498-C04-03, Ministerio de Ciencia y Tecnología, Spain, 2006-2008).

8. References

[1] P. Calegari, F. Guidec, P. Kuonen, and D. Kobler. Parallel island-based genetic algorithm for radio network design. Journal of Parallel and Distributed Computing, 47:86–90, 1997.

[2] E. Alba. Evolutionary Algorithms for Optimal Placement of Antennae in Radio Network Design, IEEE Proceedings of NIDISC'04, Santa Fe, New Mexico, USA, pp. 168, 2004.

[3] S. Khuri and T. Chiu. Heuristic algorithms for the terminal assignment problem. In Proceedings of the 1997 ACM Symposium on Applied Computing, pages 247–251. ACM Press, 1997.

[4] Price, K. and R. Storn (1997), “Differential Evolution – A simple evolution strategy for fast optimization”. Dr. Dobb’s Journal, 22 (4), 18 – 24 and 78.

[5] Price, K. and R. Storn (2006), Web site of DE as on July 2006, the URL of which is: http://www.ICSI.Berkeley.edu/~storn/code.html

[6] Hussein A. Abbass and Ruhul Sarker. The Pareto Differential Evolution Algorithm, Int. Journal on Artificial Intelligence Tools, Vol. 11, No. 4, pp. 531--552, 2002.

[7] Anderson, D.P. “BOINC: A System for Public-Resource Computing and Storage”. 5th IEEE/ACM International Workshop on Grid Computing, pp. 365-372, Nov. 8 2004, Pittsburgh, PA, USA.

[8] Anderson, D.P., Fedak, G. “The Computational and Storage Potential of Volunteer Computing”. 6th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2006), pp. 73-80, May 16-19 2006, Singapore.

[9] Anderson, D.P., Cobb, J., Korpela, E., Lebofsky, M., Werthimer, D. “SETI@home: An Experiment in Public-Resource Computing”. Communications of the ACM, Vol. 45, No. 11, pp. 56-61, November 2002.

[10] Vega, M.A., Vega, D., Gómez, J.A., Sánchez, J.M. “RND@home”. http://arcoboinc.unex.es/rnd/, 2006.