Metaheuristics for Solving a Real-World Frequency Assignment Problem in GSM Networks

Francisco Luna
U. de Málaga, Spain
flv@lcc.uma.es

César Estébanez
U. Carlos III de Madrid, Spain
cesteban@inf.uc3m.es

Coromoto León
U. de La Laguna, Spain
cleon@ull.es

José M. Chaves-González
U. de Extremadura, Spain
jm@unex.es

Enrique Alba
U. de Málaga, Spain
eat@lcc.uma.es

Ricardo Aler
U. Carlos III de Madrid, Spain
aler@inf.uc3m.es

Carlos Segura
U. de La Laguna, Spain
csegura@ull.es

Miguel A. Vega-Rodríguez
U. de Extremadura, Spain
mavega@unex.es

Antonio J. Nebro
U. de Málaga, Spain
antonio@lcc.uma.es

José M. Valls
U. Carlos III de Madrid, Spain
jvalls@inf.uc3m.es

Gara Miranda
U. de La Laguna, Spain
gmiranda@ull.es

Juan A. Gómez-Pulido
U. de Extremadura, Spain
jangomez@unex.es

ABSTRACT

The Frequency Assignment Problem (FAP) is one of the key issues in the design of GSM networks (Global System for Mobile communications), and will remain important in the foreseeable future. There are many versions of FAP, most of them benchmarking-like problems. We use a formulation of FAP, developed in published work, that focuses on aspects which are relevant for real-world GSM networks. In this paper, we have designed, adapted, and evaluated several types of metaheuristic for different time ranges. After a detailed statistical study, results indicate that these metaheuristics are very appropriate for this FAP. New interference results have been obtained, that significantly improve those published in previous research.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*; G.1.6 [**Numerical Analysis**]: Optimization—*Global optimization*

General Terms

Algorithms, Performance

Keywords

Automatic Frequency Planning, Metaheuristics

1. INTRODUCTION

Frequency planning is one of the key problems in the design of GSM systems (Global System for Mobile communications) [19]. In frequency planning, the available frequency band is slotted into channels (or frequencies) which have to be allocated to the elementary transceivers (TRXs) installed in the base stations of the network. This problem is known as Automatic Frequency Planning (AFP), Frequency Assignment Problem (FAP), or Channel Assignment Problem (CAP). FAP is a hard design task because the usable radio spectrum is very scarce and frequencies have to be reused throughout the network, and consequently, some inevitable degree of interference will occur. It is the goal of the designer to minimize the interference and satisfy other design constraints.

Tackling the FAP is crucial for today's GSM operators not only at the stage of the initial design, but also in subsequent modifications of the network aimed at solving, for instance, unpredicted interference reports or handling an increase of traffic demand in some areas. Indeed, by mid 2006 GSM services were used by more than 1.8 billion subscribers[1] across 210 countries, representing approximately 77% of the world's cellular market. It is widely accepted that the third generation mobile telecommunication system (*Universal Mobile Telecommunication System* or UMTS) [21], will coexist with the enhanced releases of the GSM standard (GPRS [11] and EDGE [9]) at least in the first phases. GSM is then expected to play an important role as a dominating technology for many years. Therefore, frequency planning in these networks will be an important task, at present as well as in the future.

Several different problem types are subsumed under the general FAP framework, and many mathematical models have been proposed since the late sixties [1]. Our work is focussed on concepts and models which are relevant for current

---

[1]http://www.wirelessintelligence.com/

real-world GSM frequency planning [7]. For these reasons, we separate ourselves from existing results, since our problem is far different from those reported in the literature with similar names (which are benchmarking-like problems). We have used a new formulation, proposed by the authors of [16], so as to take full advantage of realistic and accurate interference information from a real-world GSM network. As a generalization of the graph coloring problem, the FAP is NP-hard [12] and therefore using exact algorithms to solve real-sized instances of the problem is not practical. Metaheuristics [4, 5, 10] become the choice here.

Encouraged by the success of published work, where an ACO algorithm was applied to a real-world FAP formulation [16], we have endeavored to tackle this problem within the OPLINK coordinated research project (http://oplink.lcc.uma.es) using several metaheuristic techniques. The large effort required to design and adapt the different metaheuristics to our FAP version justifies this coordinated task. Indeed, our long term goal is to cover a wide range of types of metaheuristics on this real-world optimization problem. In the current paper, we have covered the two main types of techniques by using both population-based and trajectory-based metaheuristics.

Since the OPLINK project is composed of four Spanish research groups, two prototypical techniques were selected for each kind of metaheuristic types. Each of the research groups is in charge of the development of one optimization method, depending on its technical expertise. This way, the chosen algorithms have consisted of Genetic Algorithms and Scatter Search for the population-based category and Local Search with Heuristic Restarts (LSHR) and a (1+2) Evolutionary Algorithm for the trajectory-based metaheuristics. In order to ensure all the algorithms were programmed in the most efficient way and following common guidelines, the coordinated work has also involved the supervision of the design and development of techniques by the other groups. Great care has been put into making proper comparisons between the algorithms by using the same programming language, compiler options, and running all the algorithms under the same experimental conditions. In particular, previous work highlighted the importance of hybridizing metaheuristics with local search in this problem [16]. In order to improve the fairness of the comparison between the metaheuristics, the four algorithms tested in this paper use the same local search algorithm. Results are very promising since a new best solution for the tackled real-world GSM network has been computed.

The organization of the paper is as follows. In Section 2, we present the background of our frequency assignment problem and the mathematical formulation used for its solution. In Section 3 we outline these four chosen metaheuristics. Finally, in Section 4 we present the experimental evaluation of the algorithms, and then offer some conclusions and an outline of future work in Section 5.

2. FREQUENCY ASSIGNMENT IN GSM NETWORKS

The frequency planning is the last step in the layout of a GSM network. Prior to tackling this problem, the network designer has to address some other issues: where to install the BTSs (Base Transceiver Station) or how to set configuration parameters of the antennae (tilt, azimuth, etc.), among others [18]. Once the sites for the BTSs are selected and the sector layout is decided, the number of TRXs to be installed per sector has to be fixed. This number depends on the traffic demand that the corresponding sector has to support. Frequency planning lies in the assignment of a channel (a frequency) to every TRX [7]. The optimization problem arises because the usable radio spectrum is generally very scarce and, consequently, frequencies have to be reused by many TRXs in the network.

However, the multiple use of a same frequency may cause interferences that may reduce the quality of service (QoS) down to unsatisfactory levels. Indeed, significant interference may occur if the same or adjacent-channels are used in neighboring, overlapping cells. The point here is that computing this level of interference is a difficult task which depends not only on the channels, but also on the radio signals and the properties of the environment. The more accurate the measure of the interference in a given GSM network, the higher the quality of the frequency plan that can be computed for this network. Several ways of quantifying this interference exist, ranging from theoretical methods to extensive measurements [14]. They all result in a so-called *interference matrix*, denoted by M. Each element $M(i,j)$ of M indicates the degradation of the network quality if cells i and j operate on the same frequency. This is called *co-channel interference*. Apart from co-channel interference it may exist a so-called *adjacent-channel interference*, which occurs when two TRXs operate on adjacent channels (i.e., one TRX operates on channel f and the other on channel $f + 1$ or $f - 1$). An accurate interference matrix is therefore an essential requirement for frequency planning because the ultimate goal of any frequency assignment algorithm will be to minimize the sum of the interferences.

In real-life situations, additional complicating factors such as separation constraints among cells, or advanced interference reduction techniques such as frequency hopping or dynamic power control, etc., may be considered. The interested reader is referred to [7] for a more detailed description of frequency planning in actual GSM networks.

2.1 Mathematical Formulation

Let $T = \{t_1, t_2, \ldots, t_n\}$ be a set of n *transceivers*, and let $F_i = \{f_{i1}, \ldots, f_{ik}\} \subset \mathbb{N}$ be the set of valid *frequencies* that can be assigned to a transceiver $t_i \in T$, $i = 1, \ldots, n$. Note that k —the cardinality of F_i— is not necessarily the same for all the transceivers. Furthermore, let $S = \{s_1, s_2, \ldots, s_m\}$ be a set of given *sectors* (or cells) of cardinality m. Each transceiver $t_i \in T$ is installed in exactly one of the m sectors. Henceforth we denote the sector in which a transceiver t_i is installed by $s(t_i) \in S$. Finally, given a matrix $M = \{(\mu_{ij}, \sigma_{ij})\}_{m \times m}$, called the *interference matrix*. The two elements μ_{ij} and σ_{ij} of a matrix entry $M(i,j) = (\mu_{ij}, \sigma_{ij})$ are numerical values greater than or equal to zero. In fact, μ_{ij} represents the mean and σ_{ij} the standard deviation of a Gaussian probability distribution describing the carrier-to-interference ratio (C/I) [24] when sectors i and j operate on a same frequency. The higher the mean value, the lower the interference and thus the better the communication quality. Note that the interference matrix is defined at sector (cell) level, because the transceivers installed in each sector all serve the same area.

A solution to the problem is obtained by assigning to each transceiver $t_i \in T$ one of the frequencies from F_i.

A solution (or frequency plan) is henceforth denoted by $p \in F_1 \times F_2 \times \cdots \times F_n$, where $p(t_i) \in F_i$ is the frequency assigned to transceiver t_i. The objective is to find a solution p that minimizes the following cost function:

$$C(p) = \sum_{t \in T} \sum_{u \in T, u \neq t} C_{\text{sig}}(p, t, u) \ . \tag{1}$$

In order to define the function $C_{\text{sig}}(p, t, u)$, let s_t and s_u be the sectors in which the transceivers t and u are installed, that is, $s_t = s(t)$ and $s_u = s(u)$, respectively. Moreover, let $\mu_{s_t s_u}$ and $\sigma_{s_t s_u}$ be the two elements of the corresponding matrix entry $M(s_t, s_u)$ of the interference matrix with respect to sectors s_t and s_u. Then, $C_{\text{sig}}(p, t, u) =$

$$\begin{cases} K & \text{if } s_t = s_u, |p(t) - p(u)| < 2 \\ C_{\text{co}}(\mu_{s_t s_u}, \sigma_{s_t s_u}) & \text{if } s_t \neq s_u, \mu_{s_t s_u} > 0, |p(t) - p(u)| = 0 \\ C_{\text{adj}}(\mu_{s_t s_u}, \sigma_{s_t s_u}) & \text{if } s_t \neq s_u, \mu_{s_t s_u} > 0, |p(t) - p(u)| = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

$K \gg 0$ is a very large constant defined by the network designer so as to make it undesirable allocating the same or adjacent frequencies to transceivers serving the same area. Furthermore, function $C_{\text{co}}(\mu, \sigma)$ is defined as follows:

$$C_{\text{co}}(\mu, \sigma) = 100 \left(1.0 - Q\left(\frac{c_{\text{SH}} - \mu}{\sigma} \right) \right) \tag{3}$$

where

$$Q(z) = \int_z^\infty \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} dx \tag{4}$$

is the tail integral of a Gaussian probability distribution function with zero mean and unit variance, and c_{SH} is a minimum quality signalling threshold. Function Q is widely used in digital communication systems because it characterizes the error probability performance of digital signals [23]. This means that $Q\left(\frac{c_{\text{SH}} - \mu}{\sigma} \right)$ is the probability of the C/I ratio being greater than c_{SH} and, therefore, $C_{\text{co}}(\mu_{s_t s_u}, \sigma_{s_t s_u})$ computes the probability of the C/I ratio in the serving area of sector s_t being below the quality threshold due to the interferences provoked by sector s_u. That is, if this probability is low, the C/I value in the sector s_t is not likely to be degraded by the interfering signal coming from sector s_u and thus the communication quality yielded is high. (Note that this is compliant as to defining a minimization problem.) On the contrary, a high probability —and consequently a high cost— causes the C/I mostly to be below the minimum threshold c_{SH} and thus incurring in low quality communications.

As function Q has no closed form for the integral, it has to be evaluated numerically. For this purpose we use the complementary error function E:

$$Q(z) = \frac{1}{2} E\left(\frac{z}{\sqrt{2}} \right) \tag{5}$$

In [20], a numerical method is presented that allows the value of E to be computed with a fractional error smaller than $1.2 \cdot 10^{-7}$. Analogously, function $C_{\text{adj}}(\mu, \sigma)$ is defined as

$$\begin{aligned} C_{\text{adj}}(\mu, \sigma) &= 100 \left(1.0 - Q\left(\frac{c_{\text{SH}} - c_{\text{ACR}} - \mu}{\sigma} \right) \right) \\ &= 100 \left(1.0 - \frac{1}{2} E\left(\frac{c_{\text{SH}} - c_{\text{ACR}} - \mu}{\sigma \sqrt{2}} \right) \right). \end{aligned} \tag{6}$$

The only difference between functions C_{co} and C_{adj} is the additional constant $c_{\text{ACR}} > 0$ (*adjacent channel rejection*) in the definition of function C_{adj}. This hardware specific constant measures the receiver's ability to receive the wanted signal in the presence of an unwanted signal at an adjacent channel. Note that the effect of constant c_{ACR} is that $C_{\text{adj}}(\mu, \sigma) < C_{\text{co}}(\mu, \sigma)$. This makes sense, since using adjacent frequencies (channels) does not provoke such a strong interference as using the same frequencies.

The new feature of this mathematical model is to be found in the definition of the interference matrix information, which is directly imported from real world GSM frequency planning as currently conducted in the industry (and not generated in a computer by sampling random variables). This definition allows not only the computation of high performance frequency plans, but also the prediction of QoS. Indeed, both the definition of the interference matrix and the subsequent computations carried out to obtain the cost values are motivated by real-world GSM networks since they are related to the computation of the BER (Bit Error Rate) performance of Gaussian Minimum Shift Keying (GMSK), the modulation scheme used for GSM [23].

3. PROPOSED METAHEURISTICS

The aim of this section is to present the algorithms used in this work for solving the proposed FAP. Both population-based (Genetic Algorithms and Scatter Search) and trajectory-based metaheuristics (Evolutionary Algorithms and Local Search with Heuristic Restarts) have been evaluated. All of them use the same local search algorithm, which will be described in this section as well. ACO will also be briefly described, in order to compare the four new metaheuristics with previous work in the literature [16].

3.1 Local Search

The application of local search methods allows admissible solutions to be achieved in relatively short times. This is a typical requirement within commercial tools, the context in which the tackled problem resides. In order to perform a fair comparison of the implemented approaches, the local search strategy (see Algorithm 1) has been defined as a common element for all the proposed metaheuristics. The local search strategy has been specifically designed to deal with our version of FAP.

The aim of the designed local search is to optimize the assignment of frequencies to TRXs in a given sector, without changing the remaining network assignments. Candidate solutions (or plans) are encoded as arrays of integer values p, where $p(t)$ is the frequency assigned to TRX t. The neighbours of a candidate solution are obtained by replacing the TRXs frequencies of each sector. Therefore, the neighbourhood size is the number of sectors in the network. The reassignment of frequencies inside a sector is performed in the following way: first, the available frequencies for the sector are sorted by their involved cost. Then, two possibilities are considered, either assign the frequency with lowest associated cost to a TRX that is allowed to use that frequency, or assign its two adjacent frequencies to two different TRXs (if they are allowed to use these frequencies). For each of the new generated partial solution the same process is repeated until all TRXs in the sector have been assigned a frequency. The complete solution with lowest associated cost is considered as the new neighbour.

When all TRXs in a given sector are allowed to use the same frequency ranges, the neighbour generation process ensures the achievement of the optimal frequency assignment inside the analyzed sector, considering the remaining network fixed. Also, it has to be noted that this local search will never assign the same frequency or adjacent frequencies to two TRXs within the same sector (these co-channel and adjacent-channel interferences are the highest-cost kind of interference, as shown in Eq. 2).

The order in which neighbours are analyzed is determined in a random way (line 7 of Algorithm 1), but trying to avoid the generation of neighbours that are not going to improve the current solution. For such purpose a set called *currentSectors* containing the sectors that might improve the current solution is mantained. Initially all sectors are introduced in *currentSector* (line 2). For the generation of a new neighbour, a sector *sec* is randomly extracted from *currentSector* (line 7) and its frequencies reassigned (line 8). The local search moves to the first new generated neighbour that improves the current solution (lines 9-10), adding all the sectors that interfere or are interfered by *sec* to the set of the next sectors (*nextSectors*) to considere (lines 11-12). Each time *currentSectors* set is emptied (line 6) sectors in the next set are transferred to the current set (line 4) and the *nextSectors* set is cleared (line 5). The local search stops when none of the neighbours improves the current solution (line 3).

Algorithm 1 Pseudocode for Local Search

1: Input: current solution S
2: $nextSectors \leftarrow \{1, ..., numberOfSectors\}$
3: **while** ($nextSectors != \emptyset$) **do**
4:     $currentSectors \leftarrow nextSectors$
5:     $nextSectors \leftarrow \emptyset$
6:     **while** ($currentSectors != \emptyset$) **do**
7:         $sec \leftarrow$ extract a random sector from $currentSectors$
8:         $neighbour \leftarrow$ reassign frequencies of S in sector sec
9:         **if** ($neighbour$ improves S) **then**
10:             $S \leftarrow neighbour$
11:             $nextSectors += $ sectors interfered by sec
12:             $nextSectors += $ sectors that interfere sec
13:         **end if**
14:     **end while**
15: **end while**
16: return S

3.2 Ant Colony Optimization (ACO)

Since we want to compare with previously published results on the problem, we have included the algorithm which reported the best results for the problem so far, the ACO presented in [16]. As far as we know, this is the only instance of metaheuristics applied on this version of FAP. To summarize this approach, it works as any other ACO algorithm: at each iteration candidate solutions are constructed in a probabilistic way. The probabilistic solution construction is based on a so-called pheromone model (denoted by $\mathcal{T}$), which is a set of numerical values that encode the algorithms' search experience. After the construction phase, some of the generated solutions are used to update the pheromone values in a way that aims at biasing the future solution construction towards good solutions found during the search process. The particular approach implemented is known as

$\mathcal{MMAS}$ algorithm in the so-called hyper-cube framework (HCF); see [3].

3.3 Steady-State Genetic Algorithm (ssGA)

This algorithm is a standard steady state GA (ssGA). The tentative solutions managed by ssGA are encoded as arrays of integer values, p, where $p(t_i) \in F_i$ is the frequency assigned to transceiver t_i. That is, the solutions manipulated are tentative frequency plans of the given FAP problem instance.

Algorithm 2 Pseudocode for ssGA

1: population $\leftarrow \emptyset$
2: initialize(population)
3: **while** not time-limit **do**
4:     parents $\leftarrow$ binaryTournament(population)
5:     offspring $\leftarrow$ UX(parents,p_c)
6:     offspring $\leftarrow$ randomMutation(offspring,p_m)
7:     offspring $\leftarrow$ localSearch(offspring,localSearchSteps)
8:     population $\leftarrow$ insert(population,offspring)
9: **end while**

An outline of the algorithm is shown in Algorithm 2. The algorithm starts by creating a population of random individuals, so that all the TRXs of each individual are randomly assigned with one of their valid frequencies. As to the genetic operators, ssGA uses binary tournament as selection scheme (line 4). This operator works by randomly choosing two individuals from the population and the one having the best (lowest) fitness is selected. The algorithm applies then uniform crossover (UX) in which every allele of the offspring (i.e., the frequency of each TRX) is chosen randomly from one of the two parents (line 5). The mutation operator used is the random mutation in which the frequencies of a set of randomly chosen TRXs of the solution are reassigned with a random valid frequency. Next, the offspring undergoes a local search phase (line 7) which finally replaces the worst individual in the population, if the newly generated one is better (lower FAP cost).

3.4 Scatter Search (SS)

Scatter Search (*SS*) [17] [15] works with a quite small set of solutions called *RefSet* (solutions are encoded by using the same representation as ssGA, that is, arrays of integer values). This set is made with the most representative solutions from the population. *RefSet* is divided into quality solutions (the best frequency plans for the FAP problem) and diverse solutions (the most different ones). The number of individuals for each subset has been specially configured to solve the FAP problem. Thus, in our case, both subsets in the RefSet have the same size.

A brief description of the algorithm can be seen in Algorithm 3.

As we can observe, we have followed the typical structure of SS. The algorithm starts with the generation of the population through the assignment of a valid frequency to each single TRX in each solution (line 1). These frequencies are assigned randomly, but by using some heuristic doing impossible to assign the same frequency or adjacent frequencies to two TRXs within the same sector (this avoids the highest-cost interferences). Then, an Improvement Method fixed to the FAP problem is applied to each population individual to try to improve it (line 2). This method is the local search

Algorithm 3 Pseudocode for Scatter Search

1: initialize(population)
2: population ← localSearch(population)
3: RefSet ← generateFrom(population)
4: **while** not time-limit **do**
5:   SubSet ← subSetGenerator(RefSet)
6:   SubSet ← combinationMethod(SubSet)
7:   RefSet ← localSearch(SubSet)
8:   RefSet ← generateFrom(RefSet, population)
9: **end while**

we have explained previously (Section 3.1). After generating *RefSet* (line 3) we use a Subset Generation Method (line 5) to create all possible subsets from the *RefSet*. The next step is to apply the Solution Combination Method (line 6) to the solutions in each subset. The solutions are combined in a pair-wise way. Finally, the local search will be applied again (line 7) to try to improve a frequency planning obtained as a result of the combination method. Frequency plannings are replaced in the *RefSet* so that the best solutions to the FAP problem keep in there. When all combinations have been made, the b/2 best solutions are saved in the *RefSet* and a new population is generated to select the b/2 most diverse solutions (line 8). The distance used to measure the diversity among frequency plans is the sum of the absolute values of the difference between all the frequencies assigned to TRX. With this new *RefSet* the algorithm restarts a new iteration until the time limit for the experiment is expired (line 4).

3.5 Evolutionary Algorithm (EA)

This algorithm is a $(\mu + \alpha)$ EA approach, with a mutation operator specifically designed to face the FAP problem. The selected values for μ and α are 1 and 2, respectively. The EA approach (Algorithm 4) has been combined with the local search method to improve the algorithm convergence. Individuals are encoded as in ssGA (see Section 3.3). The individual of the first population is heuristically generated (line 1). In each generation, two new individuals are obtained by means of the mutation operator (line 4). The best individual among the new generated ones and the parent is selected to survive (line 6). After every creation of an individual the local search is applied to it (lines 2, 5).

Algorithm 4 Pseudocode for (1+2) EA

1: initialize(population)
2: population ← localSearch(population)
3: **while** not time-limit **do**
4:   offspring ← randomMutation(population)
5:   offspring ← localSearch(offspring)
6:   population ← best(population, offspring)
7: **end while**

The heuristic that generates the initial individual is based on avoiding the strongest interferences. For this purpose, interferences between sectors are considered in a decreasing order, and the TRXs involved in each interference are reassigned. If an assignment that avoids such and previously considered interferences can't be done, next interferences involving any of these sectors are not considered, i.e. their frequency assignation is unchangeable.

The variation of individuals in the population is based on a single mutation operator. First, the operator randomly selects one sector, and then, the TRXs in the selected sector, and also the ones in the interferer and victim sectors are randomly reassigned.

3.6 Local Search with Heuristic Restarts (LSHR)

This algorithm extends the local search algorithm described in Section 3.1, by adding periodically guided perturbations, to avoid local minima. In short, it periodically restarts local search by means of a probability distribution F learned during the search process. F is a matrix with dimension $M sectors \times N frequencies$ and it is updated in a similar fashion as PBIL [2]. The algorithm is summarized in Algorithm 5.

Algorithm 5 Pseudocode for Local Search with Heuristic Restarts

1: initialize(probability matrix F)
2: $S^* \leftarrow S \leftarrow$ generateFrom(F)
3: **while** not time-limit **do**
4:   $S \leftarrow$ localSearch(S)
5:   $S^* \leftarrow$ best(S,S^*)
6:   $F \leftarrow$ update(F,S)
7:   $S \leftarrow$ generateFrom(F)
8: **end while**

Matrix F is a probability distribution representing a memory of the search process so far. F_{ij} is the probability that frequency j is assigned to sector i. According to Algorithm 5, the probability distribution F is initialized in line 1 with uniform probabilities ($F_{ij} = 1/(M \times N)$). An initial solution S is obtained from F by means of roulette-wheel (line 2). The best-so-far solution S^* is also initialized. Then, a loop is entered where local search is applied on current solution S (line 4). S is considered for replacing S^* in line 5. Finally, F is updated with the improved solution S (line 6). In order to update F, F_{ij} is reinforced if the frequency j is assigned to sector i in solution S. More specifically, $F'_{ij} = (1 - f_r) * F_{ij} + f_r * P_{ij}$, where $P_{ij} = 1$ if frequency j is assigned to sector i in the best solution, and $P_{ij} = 0$ otherwise. f_r is the learning rate. The loop is repeated until time is exhausted. The `Generate` procedure in line 7 represents the heuristic restart step required so that the local search jumps out of local minima. LSHR returns the best plan S^* found during the search.

4. EXPERIMENTAL EVALUATION

This section describes the problem instance used in this paper, and presents the empirical results of the four algorithms tested.

4.1 Problem Instance

Here, we want to provide the reader with details on the FAP instance which is being tackled. The GSM network used has 711 sectors with 2,612 TRXs to be assigned a frequency; the constants used in the mathematical formulation [16] were set to $K = 100,000$, $c_{SH} = 6$ dB, and $c_{ACR} = 18$ dB, respectively. Each TRX has 18 available channels (from 134 to 151). Figure 1 displays the network topology, every triangle representing a sectorized antenna in

Table 1: Empirical results of the metaheuristics for 3 different time limits. The best, average, and standard deviation of 30 executions are presented.

Time limit	120 seconds			600 seconds			1800 seconds		
	Best	Average	Std.	Best	Average	Std.	Best	Average	Std.
ACO	90736.3	93439.5	1318.9	89946.3	92325.4	1092.8	89305.9	90649.9	727.5
ssGA	87477.3	**89540.4**	991.1	86755.7	**87850.8**	573.6	85720.3	**86908.9**	379.8
SS	91216.7	94199.6	1172.3	91069.8	93953.9	1178.6	91069.8	93820.4	1192.3
(1+2) EA	87763.9	92294.0	1407.6	86064.8	89669.8	1164.8	85607.3	88574.3	1100.3
LSHR	88543.0	92061.7	585.3	88031.0	89430.9	704.2	87743.0	88550.3	497.0
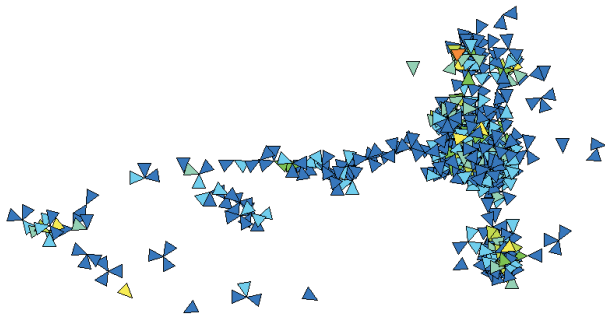
Figure 1: Topology of the GSM instance used.

which operate several TRXs. This GSM network is currently operating in a U.S. 400 km^2 city with more than 500,000 people, so its solution is of great practical interest. The data source to build the interference matrix based on the C/I probability distribution uses thousands of Mobile Measurement Reports (MMRs) [14] rather than propagation prediction models. MMRs are a more accurate data source, as they capture the call location pattern in the network and do not rely on predictions. These properties make our GSM problem more realistic than standard available benchmarks [8]. Indeed, the most similar available instances are the COST 259 benchmark, but the basic traffic load is drawn at random according to an empirically observed distribution, and signals are predicted with several propagation models. The Philadelphia, CELAR and GRAPH instances [8] are even simpler.

4.2 Parameterization

This section summarizes the parameter settings for each of the algorithms. Experiments were carried out to find the best set of parameter values for each algorithm. These parameter settings are:

- **ssGA:** Population size = 10, uniform crossover with $p_c = 1.0$, random mutation with $p_m = 0.2$, selection with binary tournament, replacement = worst individual

- **SS:** Population size = 40, RefSet size = 10, Solution combination method = uniform crossover

- **EA:** $\mu = 1$, $\alpha = 2$

- **LSHR:** Learning rate $f_r = 0.1$

4.3 Empirical Results

Experiments for all the algorithms have been carried out under exactly the same conditions: a PC with a 3GHz processor and 2GB RAM has been used for this purpose[2]. We have used gcc version 3.2.3 on a Red Hat Linux 2.4.21-4. Since we are dealing with stochastic algorithms, we have carried out 30 independent runs for each metaheuristic.

In order to provide the results with statistical confidence and detect differences between the algorithms within short and long time ranges, we have considered three different time limits (120, 600, and 1800 seconds) and performed statistical comparisons on them. The best and average results of the 30 executions for every algorithm are summarized in Table 1. For comparison purposes, we also provide the interference of plans generated randomly (random1), and those generated randomly, but removing the most costly interferences -cochanel and adjacent-channel within the same sector- (random 2) in Table 2. Table 2 also displays the results of the common local search algorithm used by the four metaheuristics. This local search method is very fast: it takes only 1.81 seconds (on average) to get an average interference of 105155.6. Although not shown in Table 1, it is important to remark that the best result of local search (97682.0) is worse than the worst result of the four metaheuristics within the 120 seconds time limit (96214.0). We can draw several preliminary conclusions from Tables 1 and 2: (1) all metaheuristics perform better than the common local search algorithm, (2) ssGA is the best performing algorithm for all time limits (on average), (3) (1+2) EA and LSHR perform similarly, and (4) all metaheuristics but SS improve on average over the results of previous research (ACO) [16]. These conclusions will now be qualified by statistical tests.

The following statistical analysis has been performed for every time limit [6, 22]. First a Kolmogorov-Smirnov test is performed in order to check whether the values of the results follow a normal (gaussian) distribution or not. If so, the Levene test checks for the homogeneity of the variances. If samples have equal variance (positive Levene test), an ANOVA test is done; otherwise we perform a Welch test. For non-gaussian distributions, the non-parametric Kruskal-Wallis test is used to compare the medians of the algorithms. 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%. To further analyze the results statistically, we have then included a post-hoc testing phase in Table 3 which allows for a multiple comparison of samples. We

---

[2]Experiments with ACO were run in our 3GHz computer. Therefore, results reported here differ from those in [16].

Table 2: Best, average, and standard deviation of 30 executions of random search and the common local search algorithm.

|	Best	Average	Std.
Random1	3.53E+13	3.33E+14	1.20E+14
Random2	244930.2	248587.6	2419.5
Local Search	97682	105155.6	2077.2

Table 3: Statistical Comparison between metaheuristics for the 120, 600, and 1800 seconds time limits. The matrix shows pairs of algorithms not significantly different. — means the algorithm is significantly different from all the rest.

Time limit	120	600	1800
ACO	SS	—	—
ssGA	—	—	—
SS	ACO	—	—
(1+2) EA	—	LSHR	LSHR
LSHR	—	(1+2) EA	(1+2) EA

have used the `multcompare` function provided by Matlab ©. It chooses the most appropriate type of critical value to be used in the multiple comparison, which ranges from the more conservative *HSD* or *Tukey-Kramer* method to less conservative *Scheffe's S* procedure [13]. The same confidence level has been kept for this testing phase ($\alpha = 0.05$).

Table 3 summarizes our significance test. For the 120 seconds results, all the differences between algorithms are significant, except ACO and SS, which perform equally well. In the case of the 600 and 1800 seconds time limits, only LSHR and (1+2) EA results show no significant differences. Let us remember that Table 1 indicated that ssGA was the best algorithm for all time limits, and now Table 3 shows that ssGA is statistically significantly better than the rest of algorithms. This makes ssGA the best performing algorithm. On the other hand differences between LSHR and (1+2) EA are not significant (except within the 120 seconds time limit).

Finally, we want to give a more detailed view of the evolution of interference along time for all the metaheuristics. Figure 2 displays the evolution of the average interference, every 120 seconds, from 2 to 30 minutes. Figure 2 shows that all algorithms obtain excellent results compared to local search alone. The graphic confirms that the best performing algorithm is ssGA, followed by (1+2) EA and LSHR. Both (1+2) EA and LSHR display the same behavior. It can also be noticed that in the range from 2 to 30 minutes, (1+2) EA, LSHR, and ssGA descend at a similar rate, but ssGA obtains the lowest interference values because it managed to descend faster during the first 120 seconds. The convergence of SS is the slowest one, reducing the FAP costs slightly during the time frame used. It is also interesting to note that the initialization heuristics used by (1+2) EA and SS have no apparent effects on performance. By comparing (1+2) EA (that uses initialization heuristics) and LSHR (that does not use them) in Figure 2, we can see that any improvement of these heuristics is lost after the first 120 seconds. Additionally, ssGA does not apply any initialization heuristics and obtains the lowest interference for all time ranges.
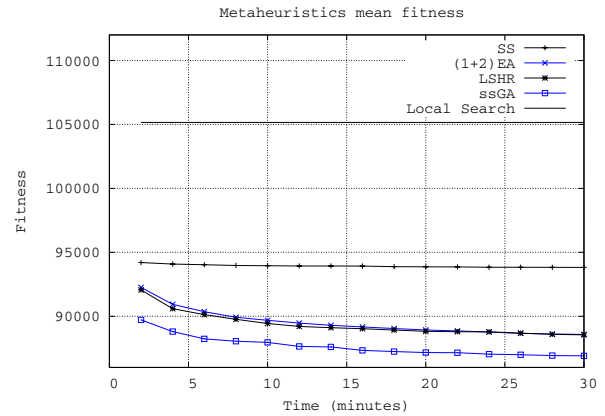
Figure 2: Average interference of the four metaheuristics every 120 seconds.

To sum up, our cooperative effort has managed to improve results beyond those found by the initial ACO algorithm. Both a population-based (ssGA) and a trajectory-based ((1+2) EA) have obtained significantly better results than the ACO. Also, in absolute terms, one of the new algorithms ((1+2) EA) has produced a best plan in 1800 seconds of 85607.3 interference units, well beyond the 89305.9 units of the initial ACO algorithm. It has to be remarked that, based on our experience, achieving differences of 3698.6 units in the long term (1800 seconds) is very difficult. Therefore, one of the results of our team effort is to set a new best absolute interference value for our real-world FAP version.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have tackled a real-world Frequency Assignment Problem (FAP) for GSM mobile networks. This work differs from previous approaches to FAP, because our new formulation focuses on real-world aspects of GSM networks, such as the use of actual interference data, measured on the field. Four research groups have collaborated in the design and adaptation of different metaheuristics, both trajectory and population-based, for our version of FAP. Following previous research advice [16], all of the algorithms have been hybridized with an efficient local search method, adapted to FAP. It has to be remarked that all the metaheuristics use the same local search method, to improve the fairness of the comparison. The techniques have been carefully evaluated and compared on a large real-world instance of FAP. It has to be remarked that this is the first time that Genetic Algorithms, Scatter Search, Evolutionary Algorithms, and Local Search with Restarts, have been applied to this version of FAP, which improve on previous results [16]. Previous work so far had only applied Ant Colony Optimization and (1+10) Evolutionary Algorithm.

To summarize, the most important contributions of the coordination of our four research groups are: the use of a FAP formulation with real-world features [16]; the design and adaptation of four different metaheuristics to this FAP; the detailed statistical comparison of all our results; and the generation of a new best result, significantly better than the ones published in the literature [16]. This result and the experience gained will guide future research.

Evaluating the algorithms by using additional real-world

instances is a matter of future work. The formulation of the FAP problem as a multiobjective optimization problem will be investigated as well.

Acknowledgements

6. REFERENCES

[1] K. I. Aardal, S. P. M. van Hoesen, A. M. C. A. Koster, C. Mannino, and A. Sassano. Models and solution techniques for frequency assignment problems. *Annals of Operations Research*, 153(1):79 – 129, 2007.

[2] S. Baluja. Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. 1994.

[3] C. Blum and M. Dorigo. The hyper-cube framework for ant colony optimization. *IEEE Trans. on System, Man, and Cybernetics — Part B*, 34(2):1161–1172, 2004.

[4] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

[5] E.K. Burke and G. Kendall. *Search methodologies: introductory tutorials in optimization and decision support techniques*. Springer, 2005.

[6] J. Demšar. Statistical comparison of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1 – 30, 2006.

[7] A. Eisenblätter. *Frequency Assignment in GSM Networks: Models, Heuristics, and Lower Bounds*. PhD thesis, Technische Universität Berlin, 2001.

[8] FAP Web. http://fap.zib.de/.

[9] A. Furuskar, J. Naslund, and H. Olofsson. EDGE – enhanced data rates for GSM and TDMA/136 evolution. *Ericsson Review*, (1), 1999.

[10] F. W. Glover and G. A. Kochenberger. *Handbook of Metaheuristics (International Series in Operations Research & Management Science)*. Springer, January 2003.

[11] H. Granbohm and J. Wiklund. GPRS – general packet radio service. *Ericsson Review*, (1), 1999.

[12] W. K. Hale. Frequency assignment: Theory and applications. *Proceedings of the IEEE*, 68(12):1497 – 1514, 1980.

[13] Y. Hochberg and A. C. Tamhane. *Multiple Comparison Procedures*. Wiley, 1987.

[14] A. M. J. Kuurne. On GSM mobile measurement based interference matrix generation. In *IEEE 55th Vehicular Technology Conference, VTC Spring 2002*, pages 1965 – 1969, 2002.

[15] M. Laguna, K. Price Hossell, and R. Marti. *Scatter Search: Methodology and Implementation in C*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[16] F. Luna, C. Blum, E. Alba, and A. J. Nebro. ACO vs EAs for solving a real-world frequency assignment problem in GSM networks. In *Genetic and Evolutionary Computation Conference (GECCO 2007)*, pages 94 – 101, 2007.

[17] R. Martí, M. Laguna, and F. Glover. Principles of scatter search. *European Journal of Operational Research*, 169(2):359–372, 2006.

[18] A. R. Mishra. *Fundamentals of Cellular Network Planning and Optimisation: 2G/2.5G/3G... Evolution to 4G*, chapter Radio Network Planning and Optimisation, pages 21 – 54. Wiley, 2004.

[19] M. Mouly and M. B. Paulet. *The GSM System for Mobile Communications*. Mouly et Paulet, Palaiseau, 1992.

[20] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[21] J. Rapeli. UMTS: Targets, system concept, and standardization in a global framework. *IEEE Personal Communications*, 2(1):30 – 37, 1995.

[22] D. J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. CRC Press, 2003.

[23] M. K. Simon and M-S. Alouini. *Digital Communication over Fading Channels: A Unified Approach to Performance Analysis*. Wiley, 2005.

[24] B. H. Walke. *Mobile Radio Networks: Networking, protocols and traffic performance*. Wiley, 2002.