On a Guided Genetic Algorithm for Frequency Assignment in Non-Homogeneous Cellular Networks

Avi Raymond, Vladimir Lyandres, Raúl Chávez Santiago
Ben-Gurion University of the Negev
Electrical and Computer Engineering Department
P.O.B. 653, Beer-Sheva, 84105, ISRAEL
e-mail: [raymonda,lyandres,rachavez]@ee.bgu.ac.il

Abstract: As the demand for bandwidth in cellular communications grows rapidly, fast and effective frequency assignment algorithms are becoming a very critical item in the design process. We consider a realistic radio network with non equal cells and non uniform traffic and propose a stochastic algorithm for the search of a quasi optimal solution for the corresponding frequency assignment problem. The algorithm represents certain modification of the known Guided Genetic Algorithm and is rather fast due to the fact that it escapes local minimums and on the other hand spends more time in the spots with the highest conflict.

Keywords: Frequency Assignment, Local Search, Genetic Algorithms, Carrier to Interference Ratio, Minimum Span

Introduction

In the last few years, when broadband services such as video, Internet, etc. enter the cellular market, a drastic growth in the additional spectrum demand is observed. One of the most economically effective ways to answer this challenge consists in an attempt to optimize the available frequency resource usage, i.e. to do the best in search of a solution of the corresponding Frequency Assignment (FA) problem, which satisfies the electromagnetic compatibility (EMC) constraints. There are three types of the EMC constraints mentioned in the literature: co-channel (CCC), co-site (CSC), and adjacent channel (ACC). As well known, the FA problem relates to the class of NP-hard problems and the only practical way to its solution consists in the use of deterministic or stochastic heuristic algorithms, such as sequential graph coloring, simulated annealing, tabu search, etc. [1]. We consider the Carrier to Interference Ratio (CIR) as the main parameter characterizing the performance of a cellular network and provide a mathematical formulation of the FA problem with the CIR minimization criteria. We propose also a new Guided Genetic Algorithm (GGA) for the FA problem solution that includes the minimum encoding scheme.

Problem definition

In real mobile radio networks cells have different sizes and channel requirements are not uniform throughout the network. Due to both these limitations, the channel reuse pattern can no longer be treated as a regular one (as the classical cellular concept supposes). Our objective is to find an optimized frequency plan to real networks existing in non-uniform environments depending on traffic demand, terrain, propagation model, etc. We suppose that the topographic map is divided in non-equal cells and in each cell a base station is located at the center with an omni directional antenna.

Assume that adjacent channel interference and noise are negligible, thus only co-channel and co-site interference are taken into consideration. Reusing the same channel in a different cell is allowed only if the CIR is satisfied. Let C_i be the carrier power of cell i, and consider the simplest propagation model [2] where the power of the transmitter P at a distance d from the base station is proportional to d^{-4} and P_0 is the power at distance 1 m from the base station, then we can write:

$$C_i = P_o \cdot d^{-\gamma} \tag{1}$$

Let I_{ij} be the co-channel interference power of cell j to cell i and β_i a set of the transmitters that use the same frequency excluding cell i. We can write CIR_i using Eq. (1) and a propagation factor $\gamma = 4$ as:

$$CIR_i = \frac{C_i}{\sum_{j \in \beta_i} I_{ij}} = \frac{P_{oi} \cdot R_i^{-4}}{\sum_{j \in \beta_i} P_{oj} \cdot d_{ij}^{-4}} \geq \alpha \tag{2}$$

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - R_i$$

where d_{ij} is the worst case distance between the interfering cell j and cell i. R is the cell radius and CIR_i is the overall interference from all the cells that use the same frequency affecting cell i. For a 30 KHz FM bandwidth a threshold $\alpha = 18$ dB should be considered [2]. Equation (2) can be rewritten with $\gamma = 1/\alpha$ to get the co-channel constraint:

$$\sum_{j \in \beta_i} \frac{p_{oj} \cdot d_{ij}^{-4}}{p_{oi} \cdot R_i^{-4}} \leq \gamma \tag{3}$$

The main problem is to minimize the total number of the used channels (the frequency plan order) or the difference between the minimum and the maximum used frequency (the frequency plan span) in the overall solution. Let be:

$F = \{1,2,\ldots,m\}$ set of available frequency channels,

$N = \{1,2,\ldots,n\}$ number of transmitters,

$d = \{d_i\}$, $i \in N$, d_i is the required links in cell i, $d_i \geq 0$ for all i,

$R = \{R_i\}$, $i \in N$, R_i is the radius of cell i,

$P = \{P_i\}$, $i \in N$, P_i is the transmitter power of cell i,

$B = \{(x_i,y_i)\}$, $i \in N$, the set of base station coordinates,

$f_{ki} = 1$ if the i-th frequency is assigned to cell k or

$f_{ki} = 0$ otherwise.

The objective is to satisfy all requirements (CIR,CCC,d_i), i.e. given F, N, d, R, P, and B:

for the Minimum Span find $\min(\max_{i \in F, k \in N} \{i \mid f_{ki} = 1\})$, and

for the Minimum Order find $\min_{k \in N}(\sum_i f_{ki})$, such that Eq. (3) holds.

Guided Genetic Algorithms

Genetic Algorithms (GA) are known to be blind search procedures that use genetic information and evolutionary functions to maximize or minimize specific functions or features in the desired population. They belong to the class of stochastic search

techniques [3]. GA maintains a population pool of some individuals called Chromosomes or Strings. Each chromosome is built out of m building stones called genes that can have one value out of a finite domain. Each chromosome has a fitness value determined by a user-defined function, which is proportional to the constraint satisfaction obtained by the chromosome. In every generation the genetic operations *Selection*, *Crossover*, and *Mutation* are performed to produce the next generation [4,5]. The major disadvantage of GA is being trapped in local minimums.

Guided Local Search (GLS) is an algorithm that aims to escape local optimality by penalizing undesired features in the solution, thus guiding the algorithm towards a better solution [6]. Consider a cost function $g(s)$ that gives a numeric value to each candidate solution s with respect to its fitness. The Augmented Cost Function, $h(s)$, which will replace $g(s)$ in the local search scheme, is defined as:

$$h(s) = g(s) + \lambda \sum \left(p_i \times I_i(s)\right) \tag{4}$$

$I(s) = 1$ if s exhibits feature i, and zero otherwise.

where p_i is the penalty integer, and λ is the constant of the algorithm that relates to the gradient. The elimination of bad features is accomplished by penalizing them thus increasing the overall fitness function (augmented cost function).

Guided Genetic Algorithm (GGA) is a hybrid algorithm based on guided local search and genetic algorithms that adds two new operators: Penalty and Fitness Template, as well as a condition to activate the new operators. The overall scheme is illustrated in Fig. 1. As it is seen in the scheme, when a local minimum is reached the penalty operator increases the augmented cost function of the chromosome, $h(s)$, by increasing the penalty of undesired features in the chromosome thus guiding the algorithm to change the undesired genetic data. The algorithm uses a fitness template, δ_{pq}, for each gene, where p is the transmitter number and q is the index in the transmitter representation. The fitness template collects data from the penalty operator and uses it in the canonical GA operation as crossover and mutation. In the first step all penalties are initialized to zero and the GA searches a local optimum with respect to the augmented cost function $h(s) = g(s)$ (all $p_i = 0$); when a local optimum is reached the penalty of the undesired features is increased, $p_i = p_i + \varepsilon$, where ε is a penalty factor; the augmented cost function $h(s)$ does no longer equals $g(s)$. The fitness template is increased with respect to the penalty, $\delta_{pq} \leftarrow p_i$, and that data is being considered in the selection of the new population by all other operators. The major problem in the GGA is determining the features set, the fitness function $g(s)$, and the parameters of the augmented cost function $h(s)$. A detailed explanation of the GGA can be found in [4].

Chromosome Representation

The Intuitive Representation (IR) of a chromosome is a vector of length $M = \sum_{i=1}^{N} d_i$ where N is the number of transmitters and M is the total required frequencies. Each of the chromosome vector elements can have any value out of an F domain, providing that it does not cause any conflicts. The main drawback of this representation is the large computer time required. A more sophisticated representation is the Fixed Representation (FR) together with the Minimum Separation Encoding scheme presented in [7]. The main idea is to reduce the search space and hence, shorten the computation time of the algorithm. Chromosomes are represented by a set of 0's and 1's, the index of the 1's represent the frequency used. Each transmitter is represented by an F_{max} length vector where F_{max} is the maximum frequency available, thus the chromosome length is $N \cdot F_{max}$. By preserving the number of 1's in each transmitter, the number of channels remains constant. The minimum separation encoding is then used to shorten the search space furthermore by solving the CSC. The idea is to represent a set such as 0001 by a new 1. The number of 0's is defined by the minimum separation of channels between consecutive elements, d_{min}, to be $(d_{min} - 1)$. For example:

Original representation ➔ *Encoded representation* ($dmin = 3$)

1000100100 $\qquad\qquad$ $\tilde{1}\tilde{0}\tilde{1}\tilde{1}$

Improvement in the algorithm time is achieved due to the shorter length of the chromosome.

Implementing CIR constraint to the GGA

We will show two new implementations of the GGA that solve the FA problem with CIR constraints. The first one uses the IR of the chromosome while the later uses FR. We have named the algorithms CIR Guided Genetic Algorithm (CGGA) and CIR Guided Genetic Fixed Algorithm (CGFGA), respectively. The two algorithms differ only in the implementation of the genetic operations such as crossover and mutation. They use the same selection operator, feature template, and penalty operator. We will review all the guided and genetic operations that we have used.

Fitness features: as described in [5,6] we use feature values for minimizing the number of used frequencies (order) and values for minimizing the maximum used frequency (span).

Frequency feature for minimizing the order: a feature is defined for each frequency from the union of domains of the links [6]. Each frequency is used multiple times in the chromosome representation, by penalizing a specific frequency we force all transmitters that use the same frequency to avoid using it. When considering a minimum order problem the main objective is to use as less frequencies as possible. By penalizing the rarely used frequencies we force them to be removed, while other frequencies that are more extensively used are forced to be taken. The cost of each feature can be written as:

$$c_f(\gamma_p) = 1 / L_f(\gamma_p) \tag{5}$$

where $L_f(\gamma_p)$ is the number of transmitters that use frequency f and γ_p is the p solution candidate (chromosome) from population γ. From this it can be seen that the less a frequency is used the higher is its cost value, which leads to a higher probability of it being penalized.

Frequency feature for minimizing the span: a feature is defined to each frequency in the union of domains with a cost related to its value, i.e. the higher is the frequency the higher is the cost of it being used. The feature set θ is the union of the feature set θ_{span} and θ_{order}.

Cost function definition: by keeping fixed the number of 1's in the solution, the requirement constraints are satisfied; while the encoding scheme solves the CSC, we are left only with the CCC and ACC with respect to CIR. Thus we have:

$$g(\gamma_p) = \sum_{k=1}^{N} \sum_{i=1}^{L_k} \sum_{j=1}^{N} CIR_{ij} \tag{6}$$

where L_k is the number of elements in transmitter k, and N is the number of transmitters. The IR does not solve the CSC, thus we should add these constraints to Eq. (6).

The augmented cost function: it is defined by Eq. (4), where s must be replaced by γ_p.

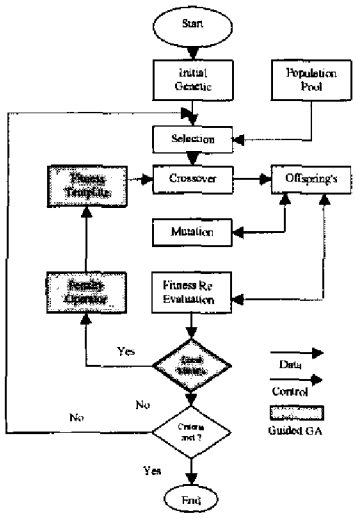

Figure 1. Guided Genetic Algorithm flow diagram

CIR Guided Genetic Algorithm (CGGA)

The GGA basic idea of feature template and penalty operator together with the intuitive chromosome representation leads to the CIR Guided Genetic Algorithm (CGGA). As it was explained before, each gene represents a frequency requirement and may have any numeric value out of the F domain. We will now describe the selection, crossover, and mutation operators. Any supplementary operator such as migration could easily be added and developed from the following.

Selection: selecting the candidate chromosome parents could be totally random or derived from the cost function $g(\gamma_p)$ or the augmented cost function $h(\gamma_p)$. We select randomly the parents with respect to their augmented cost function, thus the two chromosomes that have the lowest value of $h(\gamma_p)$ are likely chosen.

Mutation: the user-defined *mutation rate* determines the amount of genetic data that is changed. The Random Fitness Mutation (RFM) selects a gene using the roulette wheel selection method. In this method the probability for each gene to be selected is related to its fitness value. This can be implemented as follows: given a chromosome γ_p we compute the sum of all fitness templates $sum = \sum_{i=1}^{\alpha} \delta_{pi}$; the probability that a gene will be selected is given by $P(\gamma_p) = \delta_{pi} / sum$. The next step is to seek an appropriate value for replacement [8]. A pseudo code of the operator is illustrated bellow:

CGGA_RFM(Child γ_q)

1. K = *Mutation rate* * *length of chromosome*
2. for $(i = 1; i \le K; i++)$ {
3. roulette γ_{pi} by the fitness template δ_{pi} ,
4. find the best value of frequency to the selected gene,
5. solve any co-site constraint. }

Crossover: given two parents γ_p and γ_q the objective of the operator is to generate a new offspring/s that inherent some of the parents genetic data (frequency plan); the new offspring receives the "best" gene from its parents, best with respect to the fitness template of the parents, δ_{pq}. Thus the gene with the "highest" value will have a less probability to propagate to the offspring. We generate two offspring's, one gets the best genetic data while the other gets the left over genetic data.

Random Fitness Crossover (RFC): the first step is to determine the number of genes that we want to cross over by the *crossover rate*, and then we use the roulette wheel selection method to roulette a gene by its fitness value. The higher is the value the higher is the probability that a gene will be replaced by the operator. Once the crossover is complete we solve any co-site constraint that emerges during the process. A pseudo code of the operator is illustrated as follows:

1. K = *crossover rate* * *length of chromosome*
2. for $(i = 1; i \le K; i++)$ {
3. roulette γ_{pi} by the fitness template δ_{pi} ,
4. swap $\gamma_{pi} \rightarrow \gamma_{qi}$;
5. solve any co-site constraint. }

The main advantage of RFC is its ability to cross over data that gives the fitness to maximum, i.e. the algorithm swaps the "worst" areas of the solution in the sense of the fitness value.

CIR Guided Genetic Fixed Algorithm (CGGFA)

This algorithm uses the GGA basic idea of feature template and penalty operator together with the short space search of the GFA to solve the FA problem with CIR constraints. The CGGFA preserves the number of 1's in the chromosome representation throughout all of its genetic operations and, in order to guide the algorithm to a better solution in a shorter time, the genetic operations use the data from the fitness template. We present the crossover and mutation operators used by CGGFA; the selection operator is implemented in the same manner as in CGGA.

Mutation: the user-defined mutation rate determines the amount of genetic data that is changed. The genetic data change is done in pairs to maintain the number of 1's with respect to the fitness template. We select a gene using the roulette wheel selection method. The next step is to find a zero in the vector and swap the genetic data. Thus, once we have selected a gene γ_{pi} we look for a gene γ_{pj} such that $\gamma_{pi} \otimes \gamma_{pj} = 1$, where the operator is Exclusive OR.

Crossover: it basically works as for the case of CGGA, with the difference that it uses FR of the chromosomes. We present two algorithms of crossover based on the fitness template data.

Sort Fitness Crossover (SFC): the first step is to generate two index points, c_1 and c_2, along the chromosome length randomly according to the crossover rate determined by the user, where $c_1 < c_2$, and moving along the index i until we find i such that $\gamma_{pi} \otimes \gamma_{qi} = 1$. We push all i's and their fitness template into a stack and we sort the stack by the fitness value. We gradually start to swap genetic data starting with the highest fitness value down to the lowest or until the stack is empty. Elements in the stack that have been changed are deleted from it to prevent double genetic change. A pseudo code of the operator can be written as follows:

Crossover_CGGFA (parent γ_p, parent γ_q)

Generate randomly two crossover points, c_1 and c_2, considering the crossover rate value and the length of the chromosome;

for $(i = c_1; i \le c_2; i++)$ {
  if ($\gamma_{pi} \otimes \gamma_{qi} = 1$){
  stack[0][k] = i ; stack[1][k] = fitness of γ_{pi} + fitness of γ_{qi};
  k++;
  }
}
Sort stack by row 1 (fitness value);

```
If not stack empty {
i, j = 0;
  while( ( γ_p [ stack[0][i] ] = = γ_p [ stack[0][j] ] ) & ( j < k)) j++ ;
  swap γ_p [ stack[0][i] ] , γ_q [ stack[0][j] ] ;
  delete stack[][i],stack[][j] ;
  }
```

Let us consider an example with a 9 bit string, $c_1 = 3$ and $c_2 = 7$; the Crossover_CGGFA is illustrated in Fig. 2. The fitness template values are not inherited by the offspring's but are evaluated later on by the penalty operator.

Random Fitness Crossover algorithm resembles the crossover operator presented in the CGGA algorithm. Also here, the first step is to determine the amount of genetic data that will be crossed over and then to roll a gene with respect to the fitness template. The difference is that here we should find two pairs of genes whose Exclusive OR equals one and then swap them with each other. By doing so we preserve the required number of 1's in the solution.

CGGA/CGGFA algorithms flow

As in the case of GGA all penalties are set to zero and the genetic algorithm finds a local minimum by minimizing the augmented cost function $h(\gamma_p)$ defined by Eq. (4). When a local minimum is reached the penalty operator checks the best solution and penalizes the features that gives a maximum to the utility function defined as:

$$util(\gamma_p, \theta_i) = I_i(\gamma_p) \cdot \frac{c_i}{1 + p_i} \qquad (7)$$

$I_i(\gamma_p) = 1$ if solution γ_p exhibits feature θ_i, and zero otherwise; c_i and p_i are cost and penalty of feature θ_i, respectively.

All the fitness templates, δ_{pq}, of the genes related to those features and their corresponding penalties are then increased by one. The penalty operator reassesses the population and the selection operator chooses randomly two chromosomes out of the main population and the offspring's to start the mating process. Crossover and mutation operators form a new generation, which uses the data from the fitness template. When a new local minimum is reached, the penalty operator checks the best chromosome for any undesired features and starts penalizing them once again. This process is continued until a criterion is matched.

Complexity

The complexity of the algorithms CGGFA is smaller than that of CGGA due to the smaller search space as the CSC are solved by the chromosome representation. The growth function of the algorithms could easily be developed from the pseudo code.

Implementation

In a software implementation the user needs only to enter the program the location and transmitter power of each Base Station (BS), and a set of parameters such as: F_{max}, λ, stopping conditions, minimization criteria (span, order, both), population size, crossover rate, mutation rate, etc. The system default fitness template of each gene is defined by $\delta_{pq} = CIR_p + \lambda \cdot \sum_i I_{(\theta_i)} \cdot p_i$, and the fitness of each chromosome is the sum of all fitness templates. Some tests were performed using the Philadelphia benchmark examples [1] assuming the same transmitter power to each BS. As it was expected, the CGGA moved slowly towards the desired solution due to the large search space, despite the fact that the genetic functions RFM and RFC use the fitness template to determine which gene would be changed to solve the undesired regions of the solution as it is evolving. When applying the CGGFA a further improvement in processing time could be seen due to the shorter search space, obtained by the removal of the CSC that were used in the CGGA.

	Before Crossover		*After Crossover*
γ_p	Bits/freq 10|11000|110	γ^*_p	Bits/freq 10|10100|110
δ_p	fitness 20|32000|520	δ^*_p	fitness 00|00000|000
γ_q	Bits/freq 01|10110|010	γ^*_q	Bits/freq 01|11010|010
δ_q	fitness 03|10040|130	δ^*_q	fitness 00|00000|000

Figure 2. Crossover in CGGFA

Conclusions

We have studied the conflict-free fixed FA problem for non-uniform cellular radio networks. We have proposed two new genetic algorithms that minimize CIR constraints based on the fixed genetic algorithm and the guided genetic algorithm. These algorithms were called the CIR Guided Genetic Algorithm (CGGA) and the CIR Guided Genetic Fixed Algorithm (CGGFA), respectively. Both of them use the guided local search technique to bring to a minimum the overall fitness of the chromosomes. CGGA uses the intuitive representation of the chromosome while the CGGFA uses a fixed representation that solves the co-site constraints thus decreasing the search space. Both algorithms use the fitness data in their genetic operation, which leads to a faster solution. This was achieved by changing the genetic data or swapping the solution sections that have the highest interference with respect to CIR. Thus the "Red" sections are changed in each genetic evolution to be replaced with a more fitted genetic data. The overall performance of CGGFA is better than that of CGGA.

References

[1] S. Hurley, D. H. Smith and S. U. Thiel, "FASoft: A System for Discrete Channel Frequency Assignment," *Radio Science*, Vol. 32, No. 5, pp 1921-1939, September/October 1997.

[2] W. C. Y. Lee, *Mobile Communications Engineering*, McGraw-Hill, 1982.

[3] L. David and M. Steenstrup, "Genetic Algorithms and Simulated Annealing: An Overview," in *Genetic Algorithms and Simulated Annealing*, Ed. Morgan Kaufman, 1987, pp 1-11.

[4] T. L. Lau and E. P. K. Tsang, "Solving the Radio Link Frequency Assignment Problem with the Guided Genetic Algorithm," in *Proc. NATO Symp. on Radio Length Frequency Assignment, Sharing and Conservation Systems (Aerospace)*, Aalborg, Demark, October 1998.

[5] J. K. Hao and R. Dorne, "Study of Genetic Search for the Frequency Assignment Problem," in *Proc. French Nat. Conf. on Artificial Evolution (EA'95)*, Brest, France, September 1995.

[6] C. Voudouris and E. P. K. Tsang, "Partial Constraint Satisfaction Problems and Guided Local Search," in *Proc. Practical Application of Constraint Technology (PACT'96)*, London, pp. 337-356, April 1996.

[7] C. Y. Ngo and V. O. K. Li, "Fixed Channel Assignment in Cellular Radio Networks Using a Modified Genetic Algorithm," *IEEE Trans. Veh. Technol.*, Vol. 47, No 1, pp. 163-171, February 1998.

[8] C. Voudouris and E. P. K. Tsang, "Solving the Radio Link Frequency Assignment Problem Using Guided Local Search Proceedings," in *Proc. NATO Symp. on Radio Length Frequency Assignment, Sharing and Conservation Systems (Aerospace)*, Aalborg, Demark, October 1998.