Computers & Operations Research 39 (2012) 3256–3266

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Algorithms for the generalized weighted frequency assignment problem

David F. Muñoz [a,*], Diego F. Muñoz [b]

[a] Department of Industrial and Operations Engineering, Instituto Tecnológico Autónomo de México (ITAM), Río Hondo No. 1, Col. Progreso Tizapán, México City, 01080, Mexico
[b] Department of Biomedical Informatics, Stanford University, Stanford, CA 94305, USA

ARTICLE INFO

Available online 21 April 2012

Keywords:
Frequency assignment
Heuristics
Simulated annealing
Cross entropy
Tabu search
Genetic algorithm

ABSTRACT

We report the performance of 15 construction heuristics to find initial solutions, and 4 search algorithms to solve a frequency assignment problem where the value of an assigned frequency is determined by the site where it is assigned. The algorithms were tested on 3 sets of problems, the first one corresponds to the well-known Philadelphia problems, and the last two correspond to situations frequently encountered when FM frequencies are assigned in Mexico. Our experimental results show that the construction heuristics that consider the weights of the sites perform well. Among the 4 search algorithms tested, the one based on cross entropy performed better than the others in small problems, whereas in large problems the algorithm based on simulated annealing performed the best.

1. Introduction

Since its introduction in the Operations Research community [15], different versions of the frequency assignment problem (FAP) have been widely discussed. A series of conferences held between 1997 and 2004, hosted by the Center for Discrete Mathematics and Theoretical Computer Science, gave birth to a number of new algorithms and versions of this problem that can be found in [13], as well as in the webpage http://mat.gsia.cmu.edu/COLOR04/. FAPs have also been studied within the European Community, sponsored by the European Cooperation in the field of Scientific Cooperation and Technical Research, Action 259 (wireless flexible personalized communications). The results of these studies, as well as a large number of bibliographical references, may be found at the webpage http://fap.zib.de/index.php/. On the other hand, there are several review articles on FAPs, for example, those found in [1,7].

The FAP that motivated our research is a more general version of the (vertex) *set T-coloring problem* (STCP) [11]. The STCP is a generalization of the (vertex) *T-coloring problem* (TCP) that seeks an assignment of a set of colors (instead of only one color) to each vertex. A systematic study of both heuristics to obtain initial solutions and search algorithms for the STCP may be found in [3], where the objective function was to maximize the number of colors (frequencies) assigned to each vertex (site). However, in this article, we are interested in the problem of maximizing a weighted sum of the number of frequencies assigned to each site, which is particularly important when the benefit of assigning a frequency depends upon the site it is assigned to.

The main contribution intended by this article is to report a thorough study of this generalization of the STCP. Therefore, 15 heuristics for initial solutions and 4 search algorithms are tested. Our search algorithms are particular implementations of simulated annealing, tabu search, genetic algorithm and cross entropy, respectively. The results presented can be useful not only for FAPs but also for problems that have similar formulations, such as timetabling and fleet maintenance [18].

After this introduction, we present, in Section 2, a precise definition of the FAP that is of our interest and introduce our notation. In Section 3 we describe the 15 construction heuristics that were considered, and in Section 4 we present a description of the search algorithms tested in this research. In Section 5 we present the results obtained from applying the construction heuristics and search algorithms to two different sets of problems. Finally, in Section 6 we summarize our conclusions.

2. Notation and linear programming formulation

The problem that motivated our research was the simultaneous assignment of frequencies for FM radio broadcasting to more than 600 sites in Mexico. This problem was faced by the Secretary of Communications and Transportation (SCT) because of the need to satisfy the demand from AM radio stations that were switching to FM.

Under Mexican regulations, FM broadcasting can be provided using any of $a=6$ different types of antennas, under the constraint that the distance between two sites v,u that were assigned the frequencies f_v, f_u, using antenna types i,j, respectively, does not exceed

* Corresponding author. Tel.: +52 55 56284118; fax: +52 55 54904611.
E-mail addresses: davidm@itam.mx (D.F. Muñoz), dkedmun@stanford.edu (D.F. Muñoz).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.04.015

a certain value $M_{ij}(k)$ that is specified by law (where $k=|f_v-f_u|$). In addition, Mexican regulators believe that the value of a frequency depends on the site it is assigned to (because of economic considerations), and it would be desirable that the assignment of FM frequencies to the available sites maximizes the value of the assigned frequencies. In this section, we first present a brief review of the main versions of FAPs, and then we formulate the problem faced by the SCT as a FAP. We also introduce the notation that shall be used throughout this article.

2.1. Coloring and frequency assignment problems

The main versions of the FAP can be described as some generalization of coloring problems, which are among the most widespread combinatorial optimization problems due to their large number of applications, e.g., computer register allocation problems [4], school timetabling [6] and, of course, frequency assignment.

The (vertex) coloring problem is defined from an undirected graph $G=(V,E)$ and a set of colors $T=\{1,2,\ldots,N\}$. The objective of the problem is to find the smallest number $K\leq N$, and the function that assigns a color $f_v\leq K$ to each vertex $v\in V$, such that $f_v\neq f_u$ if there exists an edge $\{v,u\}\in E$. In other words, the problem is to find the smallest number of colors (K) that can be assigned to the vertices, in such a way that each pair of adjacent vertices ($\{v,u\}\in E$) is assigned different colors.

For the specific case of FAPs, each color can be interpreted as a frequency (bandwidth) and each vertex $v\in V$ as a site. Therefore, the coloring problem's objective can be interpreted as assigning a frequency to each site, such that the number of frequencies used is minimized while satisfying the constraint that no two adjacent sites may be assigned with the same frequency. For this reason, the graph $G=(V,E)$ is called the interference graph, since the existence of an edge $\{v,u\}\in E$ is interpreted as an unacceptable interference generated when both sites use the same frequency.

In practice, however, FAPs usually possess more complex constraints, since interference is not only generated between sites that use the same frequencies (co-channel interference), but also between sites that use adjacent frequencies. For this reason, the TCP was introduced.

The TCP considers that, for any $v,u\in V$ there is a set $T_{\{v,u\}}\subseteq\{0,1,\ldots\}$ that contains the difference of frequencies that are prohibited for each pair of sites $\{v,u\}$; in other words, the frequency assignment must satisfy $|f_v-f_u|\notin T_{\{v,u\}}$ for any $v,u\in V$. Depending on the application of interest, there are several different objectives the TCP may consider, among the most common objectives we mention: minimizing the number of different assigned frequencies (MOFAP for **M**inimum **O**rder **F**requency **A**ssignment **P**roblem) and minimizing the maximum difference between two assigned frequencies (MSFAP for **M**inimum **S**pan **F**requency **A**ssignment **P**roblem). A particularly important formulation corresponds to the MIFAP (for **M**inimum **I**nterference **F**requency **A**ssignment **P**roblem), where some interference constraints may be violated and the objective is to minimize a measure of total interference.

The STCP is a generalization of the TCP, that seeks the assignment of a set of frequencies $S_v\subseteq T$, instead of assigning only one frequency to every site $v\in V$, while avoiding interferences, i.e., $|f_v-f_u|\notin T_{\{v,u\}}$ for any $v,u\in V$, and $f_v\in S_v$, $f_u\in S_u$. As in the TCP, several objectives can be proposed for the STCP, such as the MOFAP, MSFAP and MIFAP.

A very general formulation for the objective function of a FAP has been proposed by Hurley and Smith [12]. Under their formulation there is a function $h(d, f_1, f_2)$ that evaluates the magnitude of the interference when two sites (separated by a distance d) are assigned with the frequencies f_1 and f_2, respectively. Thus, the constraints must assure that $f_v\in S_v$ and $f_u\in S_u$ are not simultaneously assigned when the interference level $h(d(v,u)f_vf_u)$ exceeds a certain predetermined level (where $d(v,u)$ is the distance between the sites $v,u\in V$). This formulation also takes into account the following variables:

- e_{vio}: number of interference constraints that are violated,
- e_{sum}: sum of the magnitude by which the interference constraints are violated
- f_{large}: highest frequency assigned,
- f_{small}: smallest frequency assigned,
- e_{order}: number of different frequencies assigned, and
- l_{vio}: maximum value by which an interference constraint is violated.

The objective function is

$$Z=\mu_1 e_{vio}+\mu_2 e_{sum}+\mu_3(f_{large}-f_{small})+\mu_4 e_{order}+\mu_5 f_{large}+\mu_6 l_{vio}, \tag{1}$$

where selecting particular values for the parameters μ_i adjusts the problem to a particular interest. For example, $\mu_6=1$, and $\mu_1=\mu_2=\cdots=\mu_5=0$ corresponds to the problem of minimizing the maximum interference for every pair of assigned frequencies. Similarly, setting a relatively high value for μ_1 corresponds to the problem of finding a solution that does not violate any of the constraints.

2.2. The generalized weighted frequency assignment problem

In this section we present a linear programming formulation for the FAP that motivated our research. This formulation considers only binary variables, for which it is called a binary linear programming (BLP) formulation. We are interested in a generalization of the STCP that considers weights for the assigned frequencies. To be more precise, there is a benefit w_{vf} to be obtained when the frequency $f\in T$ is assigned to the site $v\in V$, and the objective is to maximize:

$$Z=\sum_{v\in V}\sum_{f\in S_v} w_{vf}, \tag{2}$$

where $S_v\subseteq T$ is the set of frequencies assigned to the site v. Note that the objective function (2) is not a particular case of (1), nor is it similar to the one studied in [3]. At this point, it is worth mentioning that we are particularly interested in the case where $w_{vf}=b_v$ for every $f\in T$, which is precisely the case when the expected benefit is determined solely by the site regardless of what frequency is assigned to it.

With the objective of presenting a BLP formulation of our FAP, for any site $v\in V$ we let $F_v\subseteq T$ be the set of available frequencies for that site, so that the set S_v of assigned frequencies for the site v must satisfy $S_v\subseteq F_v$. This constraint is necessary in our application, because there are several frequencies that have already been assigned to certain sites, which prohibit the use of specific frequencies in a particular site. The decision variables for our BLP are:

$$X_{vf}=\begin{cases}1, & \text{if the frequency } f \text{ is assigned to the site } v,\\ 0, & \text{otherwise,}\end{cases} \tag{3}$$

for $v\in V$, and $f\in F_v$. Under this notation, the objective function (2) becomes:

$$Z=\sum_{v\in V}\sum_{f\in F_v} w_{vf}X_{vf}. \tag{4}$$

Let c denote the maximum difference for a pair of prohibited frequencies, so that prohibiting a pair of frequencies depends on

$(c+1)$ distance-separation matrices $(M_{ij}(k))$, $k=0,1,\ldots,c$, where:

$M_{ij}(k)=$ Minimum distance that must exist between two sites when the difference between two assignedfrequencies (one for each site) is k, where one site uses an antenna typeiand the other uses an antenna type j.

We assume that each site is assigned only one type of antenna, so each frequency assigned to a site v will transmit its signal using an antenna of the type $I_v\in\{1,2,\ldots a\}$, and therefore, if we let d_{vu} denote the distance between sites $v,u\in V$, the interference constraints are:

$$X_{vf}+X_{ug}\leq 1,\ \text{if}\ d_{vu}<M_{I_vI_u}(k),\ \text{and}\ |f-g|=k, \tag{5}$$

for some $k=0,1,\ldots,c$.

The BLP formulation for our STCP is to maximize (4), subject to the constraints (5). From here on, we will identify the problem of interest as the generalized weighted frequency assignment problem (GWFAP), and although some of the methods that were tested may be adapted to a more general case, we will focus on the GWFAP in which $w_{vf}=b_v$ for any $f\in T$ and $v\in V$.

3. Construction heuristics

As is well known, the coloring problem (as well as the generalizations described in Section 2) is NP-complete [8]. Therefore, for a large number of vertices, existing algorithms (e.g., branch and bound for general BLP formulations) to obtain an exact solution might not be capable of providing a solution to the problem. Assigning FM frequencies to different sites in Mexico is a large problem, since it is a FAP with over 600 sites (vertices) and around 40 available frequencies for each site (see Section 5.3). For this reason, obtaining initial solutions using fast procedures, frequently known as heuristics, may be quite useful. On the other hand, a good initial solution may also be used as the starting point of a search algorithm, as those discussed in Section 4. In this section, we will present a set of heuristics to obtain initial solutions for the GWFAP. These heuristics were tested in [3] for the case where no weights are considered and in this research we adapt and test each of them for the GWFAP. The heuristics considered in this article may be classified in three different categories, which we will identify with the following names: ST-GREEDY, G-DSATUR and G-RLF.

The heuristics belonging to the ST-GREEDY category use a corresponding permutation $\pi=(\pi_1,\ldots,\pi_p)$ of the sites (where $p=|V|$ is the number of sites), and frequencies are assigned by using a greedy procedure that is described in the algorithm of Fig. 1. This algorithm starts by initializing every set F_v with the set of feasible frequencies obtained from expression (5). Furthermore, each time a frequency is assigned to a site; these sets must be updated excluding all frequencies that interfere with the assigned frequency.

In order to specify the heuristics belonging to the ST-GREEDY category we need to define some concepts, such as: the degree of a site, the adjusted degree of a site and "smallest degree (or adjusted degree) last" rule. To calculate the degree of a site, we say that there is an edge between two sites $v,u\in V$ ($v=u$ is allowed) if there are two frequencies $f_v\in F_v$, $f_u\in F_u$ such that the simultaneous assignment of f_v to site v and f_u to site u is prohibited, i.e., $d_{vu}<M_{I_vI_u}(k)$, and $|f_v-f_u|=k$, using the notation of Eq. (5). Thus, the degree of a site is the total number of edges incident to that site. Under this notation, we say that the difference between the frequencies of an edge $\{f_v,f_u\}$ is $|f_v-f_u|$, so that the adjusted degree of a site is the sum of all the frequency differences corresponding to the edges incident to the site. Finally, the "smallest degree (or adjusted degree) last" rule, that is required to generate the permutations used in the SDL and SADL heuristics, is implemented by sequencing, starting from the end, the site that has the smallest degree (or adjusted degree) without taking into account the edges incident to the sites that have already been sequenced.

We tested six different heuristics belonging to the ST-GREEDY category, each with a different type of permutation of the sites. All of the heuristics from this category are described as follows:

- **RO**: The permutation is generated randomly, and each site is assigned the same probability.
- **LWF:** The permutation is obtained by sorting the sites using the "largest weight first" rule, and ties are broken by the largest adjusted degree.
- **LDF:** The permutation is obtained by sorting the sites using the "largest degree first" rule.
- **LADF:** The permutation is obtained by sorting the sites using the "largest adjusted degree first" rule.
- **SDL:** The permutation is obtained by sorting the sites using the "smallest degree last" rule.
- **SADL:** The permutation is obtained by sorting the sites using the "smallest adjusted degree last" rule.

The heuristics belonging to the G-DSATUR category are based on a generalization of the DSATUR heuristic proposed by Costa [5] for the TCP. Under any of these heuristics, frequencies are also assigned using the ST-GREEDY algorithm of Fig. 1. However, the permutation of the sites is determined by the site saturation degree (SSD), which is the number of frequencies that were

Initialize the feasible sets $F_v, v\in V$
For $j=1$ ***to*** p
  Set $v\leftarrow\pi_j$
  While $|F_v|>0$
    Assign the lowest frequency $f_v\in F_v$ **to site** v
    For every site $u\in V$, **exclude from** F_u **all frequencies** $f_u\in F_u$ **that interfere (do not satisfy the corresponding distance separation) with the frequency** f_v **in site** v
  End While
End For

Fig. 1. Pseudo-code for the ST-GREEDY algorithm.

prohibited (in the corresponding site) by the frequencies previously assigned. We tested the following six heuristics in this category:

- **LSWF**: The permutation is obtained according to the "largest SSD first" rule, and ties are broken by the largest weight.
- **SSWF:** The permutation is obtained according to the "smallest SSD first" rule, and ties are broken by the largest weight.
- **LSDF:** The permutation is obtained according to the "largest SSD first" rule, and ties are broken by the largest degree.
- **SSDF:** The permutation is obtained according to the "smallest SSD first" rule, and ties are broken by the largest degree.
- **LSADF:** The permutation is obtained according to the "largest SSD first" rule, and ties are broken by the largest adjusted degree.
- **SSADF:** The permutation is obtained according to the "smallest SSD first" rule, and ties are broken by the largest adjusted degree.

The heuristics belonging to the G-RLF category no longer use the ST-GREEDY algorithm. Instead, they select a frequency, starting from the smallest one available, and assign it to all sites where the assignment is feasible, prohibiting all other assignments that would interfere with it in each site. The following three heuristics belonging to this category were tested.

- **RLWF:** The same frequency is assigned to the sites according to the "largest weight first" rule.
- **RLDF:** The same frequency is assigned to the sites according to the "largest degree first" rule.
- **RLADF:** The same frequency is assigned to the sites according to the "largest adjusted degree first" rule.

4. Search algorithms

As mentioned previously, when the number of sites and frequencies considered in a GWFAP are large, exact methods (e.g., branch and bound) might not be capable of providing a solution. Similarly, some of the heuristics described in the previous section might not be able to provide a fast solution that is reasonably close to the optimal. Therefore, due to the high market value of the frequencies, the use of more complex methods such as search algorithms (also known as meta-heuristics), that may provide a better solution at the expense of more computing time, is justifiable. The search algorithms that have been more frequently explored for the STCP, and the TCP in general, belong to implementations of three general search methods: simulated annealing (SA), tabu search (TS) and genetic algorithms (GA). In this section we consider algorithms based on each of these three principles to solve the GWFAP, and we introduce a new algorithm based on the concept of cross entropy (CE). Among the few references on CE implementations for FAPs, we can mention [16], where the authors describe a successful implementation of a hybrid CE/Hopfield Neural Network algorithm to solve several instances of the STCP, however, this implementation does not apply to our GWFAP.

4.1. Solution neighborhood

SA or TS algorithms require the definition of so-called neighboring solutions, i.e., solutions that are identically with the current solution except for a few assignments. Starting from a current solution, SA and TS iteratively search for neighboring solutions. Among these neighbors, one is chosen as next solution to continue with. The performance of an SA or TS algorithm is intimately linked with the definition of the neighborhood. Although many implementations of these search algorithms consider unfeasible solutions (in other words, they violate some of the interference constraints, see [12]), all the algorithms considered in this research take into account only feasible solutions of the GWFAP. Therefore, the concept of solution neighborhood used in our implementations of the SA and TS algorithms consists of feasible neighbors only.

To define a solution neighborhood we will use the notation introduced in Section 2, by letting $V=\{v_1,\ldots,v_p\}$ be the set of sites, any solution x of the GWFAP may be viewed as a collection $\{S_1^x,\ldots,S_p^x\}$, where $S_i^x \subseteq F_{v_i}$ is the set of frequencies assigned to the site v_i, $i=1,2,\ldots,p$. We say that another solution y is a neighbor solution of x if it can be obtained by applying the following procedure:

i. Select a site $v_i \in V$, and a non-assigned frequency $f_i \in F_{v_i}-S_i^x$.
ii. For every site $v_j \in V$, set $S_j^y = S_j^x - U_j^x(f_i)$, where $U_j^x(f_i) \subseteq S_j^x$ is the set of frequencies in site v_j that interfere with frequency f_i in the site v_i.
iii. Assign frequency f_i to site v_i.

The neighborhood for x is defined as the set of all solutions that are neighbors of x. Note that, given a solution x, the objective function defined in (2) takes the form of

$$Z(x) = \sum_{i=1}^{p} \sum_{f \in S_i^x} w_{if}. \tag{6}$$

The definition of neighborhood introduced in this section, might have the inconvenience that, when prohibiting assigned frequencies that interfere with the entering frequency, other frequencies that do not interfere with those already assigned are not introduced. For this reason, we decided to design another procedure that tries to improve a given feasible solution by assigning frequencies in a greedily fashion. This procedure, which we will call mutation (and which is actually a local search procedure), is defined as follows:

The mutation procedure begins with a solution x, where $S_i^x \subseteq F_{v_i}$, and a permutation $\pi=(\pi_1,\ldots,\pi_p)$ of the sites, and then applies the following steps:

Step 1. All sites v_i (in the order given by π) are explored; for each site, all neighbor solutions are moved through (according to the smallest frequencies in $F_{v_i}-S_i^x$), to determine the best neighbor solution y, and its corresponding objective function $Z(y)$.
Step 2. If $Z(y) \leq Z(x)$ the procedure ends, and x is returned as the mutation of x, otherwise we set $x \leftarrow y$ and go back to step 1.

We say a solution x to the GWFAP is locally optimal if no neighbor y satisfies $Z(y) > Z(x)$, and therefore, the previous procedure returns a mutation of x that is locally optimal.

4.2. Simulated annealing algorithm

The search algorithm SA is a stochastic search procedure that was inspired from certain thermodynamic phenomena like the freezing and crystallization of liquids, or the heating and controlled cooling of certain metals to obtain more stable (less energy) structures. For this reason, the SA procedure uses a variable named temperature (variable *Tem* in Fig. 2), that decreases as the algorithm runs, while attempting to minimize an objective function (which is the analog of energy level in thermodynamic phenomena).

In Fig. 2 we present a pseudo-code of the SA procedure (for the maximization case) that was used in the implementation that we

tested on the GWFAP. As observed in Fig. 2, starting off from an initial temperature T_0 and from an initial solution x_0, a series of loops are generated in which N_c neighbor solutions are explored (each one arising from the current solution x_0). Neighbor solutions are randomly generated by first selecting a site and then assigning a new frequency to the site. Each generated neighbor solution may be accepted, in which case it updates the current solution x_0, otherwise it is rejected.

According to Fig. 2, a neighbor solution x_1 is accepted if $P[Z(x_0),Z(x_1)] > r$, where r is distributed uniformly on (0,1), and

$$P[Z(x_0),Z(x_1)] = \begin{cases} e^{-(Z(x_0)-Z(x_1))/BTem}, & \text{if } Z(x_1) \le Z(x_0), \\ 1, & \text{otherwise,} \end{cases} \quad (7)$$

where *Tem* denotes the temperature and *B* (called the Boltzmann constant) is a scale parameter that must be adjusted empirically. As a result, the new solution is accepted if it has a larger objective value ($Z(x_1) > Z(x_0)$), otherwise the acceptance probability increases with higher temperature, and decreases with lower temperature. This acceptance condition allows us to accept solutions with smaller objective values, in order to avoid being absorbed or stuck in a local optimum. Accepting lower valued solutions, however, has a smaller chance of occurring as the temperature decreases.

Initialize the feasible sets $F_v, v \in V$
Set $Tem = T_0$ and generate the initial solution x_0
Calculate $Z(x_0)$ and initialize N_c
While $Tem > T_{\min}$
  For $i = 1$ to N_c
    Generate a neighbor solution x_1 and calculate $Z(x_1)$
    Generate r uniform on $(0,1)$
    If $r < P[Z(x_0), Z(x_1)]$
      $x_0 \leftarrow x_1, Z(x_0) \leftarrow Z(x_1)$
    End If
  End For
  Set $Tem \leftarrow \alpha Tem$
End While

Fig. 2. Pseudo-code for the SA algorithm.

Note that the implementation of the algorithm of Fig. 2 requires that some other parameters must be specified, such as the initial temperature T_0, the final temperature $T_{\min}$, the number of iterations N_c of the main loop, and the cooling speed α, which is set between 0 and 1. In addition, an initial solution x_0 is required, and in our experiments we considered the solution provided by the RLWF heuristics described in Section 3.

For the results reported in Section 5, the parameter N_c is adjusted as follows. We set an initial value of $N_c = N_0 = \sum_{i=1}^{p} |F_{v_i}|$, and after every loop of the algorithm of Fig. 2 we update N_c according to $N_c \leftarrow \min\{\lceil N_c/\alpha \rceil, 2000N_0\}$ (where α is the cooling speed). This cooling scheme is suggested by Hurley and Smith [12] to increase the size of the main loop as the temperature decreases.

To determine the initial temperature T_0 we use the following "auto-adjusting" procedure before running the algorithm. Starting with a temperature $T_0 = 1$, we run one main loop of the algorithm with $N_c = N_0$ iterations, if the acceptance rate (the number of accepted solutions divided by N_c) is lower than 0.9, then the temperature is doubled and another loop of the algorithm is run. This continues until the acceptance rate is not lower than 0.9. Initializing the temperature under this procedure will allow us to obtain a reasonable rate of acceptance.

4.3. Tabu search algorithm

Algorithms based on the TS principle [9] store certain solutions (named tabu) with the objective of not searching again through previously explored solutions. As described in Fig. 3, at any iteration of the main loop, all neighbor solutions of a solution x_k are explored, and to each of them the mutation procedure

Initialize the feasible sets $F_v, v \in V$
Initialize long and short term memories
Generate initial solution x_0 and set $k = 0$
Initialize $x_B = x_0, Z_B = Z(x_0), Z_T = -\infty$
While not finished
  For $i = 1$ to p and each $f \in F_{v_i} - S_{v_i}^{x_k}$
    Generate the neighbor solution y with $f \in S_{v_i}^{y}$ and apply mutation to y
    If y is tabu and $Z(y) > Z_T$ Then $x_T \leftarrow y, Z_T \leftarrow Z(y)$
    If $Z(y) > Z_B$ Then $x_B \leftarrow y, Z_B \leftarrow Z(y)$
  End For
  If $Z_T > Z_B$ and the aspiration criterion is fulfilled
    $x_{k+1} \leftarrow x_T, Z(x_{k+1}) \leftarrow Z(x_T)$
  Else
    $x_{k+1} \leftarrow x_B, Z(x_{k+1}) \leftarrow Z_B$
  End If
  Update long and short term memories and set $k \leftarrow k+1$
End While

Fig. 3. Pseudo-code for the TS algorithm.

(described in the previous subsection) is applied. From these solutions, the best tabu and non-tabu solutions are stored. If the tabu solution is better, it may be explored in the next loop if it reaches an aspiration criterion (that we explain below). Otherwise, the next solution to be explored is the best non-tabu solution. Note that the current solution x_k is updated at the end of any iteration of the main loop.

To explain the mechanism that we used to determine if a solution is tabu or not, let us T_v^k denote the set of frequencies that were assigned to the site v at the beginning of iteration k of the main loop. If in this iteration, the neighbor solution y (see Fig. 3) assigned a frequency $f \in F_{v_i} - S_{v_i}^{x_k}$, such that $f \in T_{v_i}^j$ for some $j \in \{k-M_{SH}-1,\ldots,k-1\}$, and $i \in \{1,\ldots,p\}$ then it is said that the solution y is tabu due to the short term memory (the parameter M_{SH} is the length of the short term memory). A solution may also be considered tabu (due to the long term memory) if $\sum_{j=1}^{k} I(f \in T_{v_i}^j) > k\beta$, where β is another parameter and $I(f \in T_{v_i}^j)$ is 1 if $f \in T_{v_i}^j$, and 0 otherwise.

The aspiration criterion of a tabu solution is that its objective function's value is greater than those of the solutions from the last M_{SH} loops. It is worth mentioning that the TS algorithm, tested in Section 5, ends when a certain run-time is exceeded.

4.4. A genetic algorithm

Search procedures based on the GA concept [10] have been successfully applied in problems related to the GWFAP [3,14]. They attempt to mimic the natural processes of evolution and this is why (see Fig. 4) they start from an initial population of solutions $\{x_1,\ldots,x_P\}$. In any iteration of the main loop, a new generation is acquired. The GA implemented for the GWFAP uses a procedure called crossover to generate a descendant (x) for each element of the previous population (x_i) by combining it with another element (x_k) selected randomly. After the obtained descendant mutates (according to the procedure described in Section 3), it may replace its parent with the lowest objective function value, if its value is not worse than both of its parents. Therefore, some elements of the next generation may come from the previous generation, or be descendants. In our implementation, the GA ends when a certain run-time is exceeded.

The crossover procedure between two solutions $z=\{S_1^z,\ldots,S_p^z\}$, $y=\{S_1^y,\ldots,S_p^y\}$ that we implemented assumes that the set of feasible frequencies for the descendant in site v_i is $F_{v_i}=S_i^z \cup S_i^y$, and determines the assigned frequencies using the ST-GREEDY algorithm of Fig. 1 with a random permutation of the sites. The ST-GREEDY algorithm is also used to generate the initial

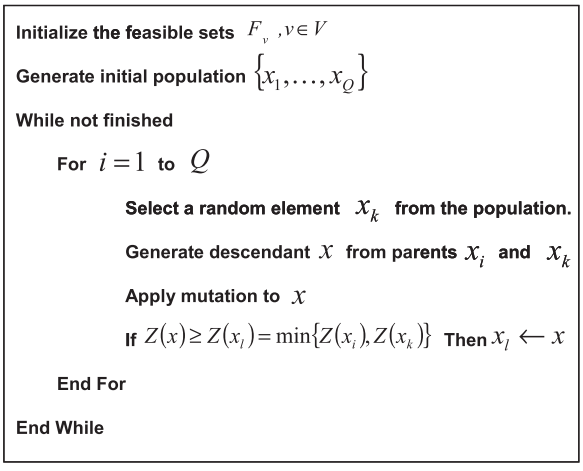

Fig. 4. Pseudo-code for the GA algorithm.

population with the LWF permutation $\pi^1=(\pi_1^1,\pi_2^1,\ldots,\pi_p^1)$ of the sites for the first element of the population. Then $Q=p$ permutations are generated by "spinning the wheel", i.e., $\pi^2=(\pi_p^1,\pi_1^1,\ldots,\pi_{p-1}^1),\ldots,\pi^p=(\pi_2^1,\pi_3^1,\ldots,\pi_p^1,\pi_1^1)$.

4.5. Cross entropy algorithm

The algorithms based on CE [17] are stochastic search algorithms that alter the probabilities of selection on the search space by using the concept of CE to assign larger probabilities to solutions that possess better objective function values. In the following paragraphs, we briefly review the concepts related to CE, and describe our implementation of the CE algorithm.

Let $q_0(x)$ and $q_1(x)$ be probability mass functions on a finite set E, the CE between these probability functions is defined as

$$D(q_0,q_1)=E_{q_0}\left[\log\left(\frac{q_0(x)}{q_1(x)}\right)\right]=\sum_{x\in E} q_0(x)\log(q_0(x))-\sum_{x\in E} q_0(x)\log(q_1(x)). \quad (8)$$

Although in our implementation we only require the case in which E is finite, the concept of CE may be naturally extended to other types of probability distributions. For example, if $q_0(x)$ and $q_1(x)$ are density functions we can use integrals instead of summation operators in (8), on which case we require that the support of q_1 (i.e., $\{x\in E: q_1(x)>0\}$) is contained in the support of q_0 (otherwise $D(q_0,q_1)$ will diverge).

It can be shown that $D(q_0,q_1)\geq 0$, and $D(q_0,q_1)=0$ if and only if $q_0 \equiv q_1$, thus, $D(q_0,q_1)$ measures how close the probability function q_1 is to the probability function q_0; although it is not a distance in the strict mathematical sense, since $D(q_0,q_1)=D(q_1,q_0)$ is not always true.

Given a family of probability mass functions $\{q(x;\theta):\theta\in\Theta\}$ on E, importance sampling and the concept of CE may be used to estimate the probability

$$l(\gamma_0)=P_{\theta_0}[Z(X)\geq\gamma_0]=\sum_{x\in E} I[Z(x)\geq\gamma_0]q(x;\theta_0)=E_{\theta_0}[I[Z(X)\geq\gamma_0]], \quad (9)$$

where Z is a function on E, and $I[A]$ is 1 when event A occurs and zero otherwise. To see this, note that

$$l(\gamma_0)=\sum_{x\in E} I[Z(x)\geq\gamma_0]\frac{q(x;\theta_0)}{q_0(x)}q_0(x)=E_{q_0}[I[Z(X)\geq\gamma_0]L(X)], \quad (10)$$

where $q_0(x)$ is a probability mass function that has the same support as $q(x;\theta_0)$, and $L(x)=q(x;\theta_0)/q_0(x)$ is called the likelihood ratio. Expression (10) indicates that $l(\gamma_0)$ may be estimated from a sample $X_1,\ldots,X_n$ of $q_0(x)$, instead of sampling from $q(x;\theta_0)$ as suggested by expression (9). The probability function $q^*(x)$ that provides the smallest variance is

$$q^*(x)=\frac{I[Z(x)\geq\gamma_0]q(x;\theta_0)}{l(\gamma_0)}, \quad (11)$$

which gives a variance of zero, since $P_{q^*}[I[Z(X)^3\gamma_0]]=1$. However, $q^*(x)$ cannot be known a priori, since it would require us to also know the value of $l(\gamma_0)$, which is precisely what we want to estimate. An alternative is to use the element of $\{q(x;\theta):\theta\in\Theta\}$ that is closest to $q^*(x)$ in the CE sense. In order to find this element we note that minimizing $D(q_0,q_1)$ in (8) for a fixed q_0 is equivalent to minimizing the second term on the right-hand side of (8). Hence, we must solve

$$\max_{\theta_1\in\Theta}\sum_{x\in E}\frac{I[Z(x)\geq\gamma_0]q(x;\theta_0)}{l(\gamma_0)}\log(q(x;\theta_1)),$$

Since $l(\gamma_0)$ is a constant, the problem we just formulated is equivalent to

$$\max_{\theta_1\in\Theta} E_{\theta_0}[I[Z(X)\geq\gamma_0]\ \log(q(x;\theta_1))], \quad (12)$$

where a convenient interpretation of (12) is that $P_{\theta_1}[I[Z(X) \geq \gamma_0]]$ must approach 1.

The CE concept may be applied to the problem of finding γ^* and $x^* \in E$ such that $\gamma^* = Z(x^*) = \max_{x \in E} Z(x)$, by considering a family of probability mass functions $\{q(x;\theta): \theta \in \Theta\}$ on E, and defining the associated problem of estimating

$$l(\gamma) = P_{\theta_0}[Z(X) \geq \gamma] = \sum_{x \in E} I[Z(X) \geq \gamma] q(x; \theta_0),$$

for a value γ_0 close to γ^*. When searching for $\theta_1 \in \Theta$ such that $P_{\theta_1}[Z(X) \geq \gamma_0]$ is close to 1, as seen previously, a good candidate is the solution of (12), which may be approximated from a sample $X_1,\ldots,X_n$ of $q(x;\theta_0)$, as a solution to the problem

$$\max_{\theta_1 \in \Theta} \frac{1}{n} \sum_{i=1}^{n} I[Z(X_i) \geq \gamma_0] \log(q(X_i; \theta_1)). \tag{13}$$

Also, the sample $X_1,\ldots,X_n$ (for a sufficiently large n) allows us to approach γ^*, for which we calculate $Z(X_1),\ldots,Z(X_n)$, and sort the values: $Z_{(1)} \leq \cdots \leq Z_{(n)}$. If p_0 is close to zero, the value

$$\gamma_1 = Z_{(\lceil n(1-p_0) \rceil)} \tag{14}$$

should be close to γ^*. The expressions (13) and (14) may be viewed as updates of the parameters θ_0 and γ_0, in order to have $P_{\theta_1}[Z(X) \geq \gamma_1]$ close to 1, and γ_1 close to γ^*. This is the intuition behind the CE algorithm described in Fig. 5.

In order to describe how a particular algorithm based on CE can be developed from (13) to (14), note that the solution of (13) has an interpretation related to the maximum likelihood principle. If we let $X_{(1)},\ldots,X_{(k)}$ be the values of the sample that satisfy $Z(X_{(j)}) \geq \gamma_0$, then θ_1 is the value that maximizes the likelihood $\prod_{j=1}^{k} q(X_{(j)}; \theta)$. In other words, it is the maximum likelihood estimator from the sample $X_{(1)},\ldots,X_{(k)}$. For the case where parameter θ is itself a probability distribution on a search space (as is the case of the algorithm that we tested), the solution of (13) is the empirical distribution obtained from $X_{(1)},\ldots,X_{(k)}$. Thus, in order to describe our implementation of the CE algorithm, we only need to describe our initial sampling scheme, whose probabilities are updated (at each iteration) by considering the empirical distribution corresponding to the best solutions $X_{(1)},\ldots,X_{(k)}$.

Our implementation of the CE algorithm is inspired in the idea that an optimal solution for the STCP may be obtained through a permutation of the sites by assigning frequencies greedily [19]. Thus, we generate the first site $v(0)$ by sampling from an initial distribution $Q_0(u)$ on $E=\{v_1,\ldots,v_p\}$ and assign the smallest available frequency in $F_{v(0)}$. Then, the next sites are selected sequentially by sampling from a probability mass function $Q_k(u)$ on E, $k=1,2,\ldots$, and in each step, the smallest available frequency is assigned to the site. The family of distributions $\{Q_k(u): k=0,1,\ldots\}$ is updated after the corresponding sample of solutions is obtained.

The corresponding initial family of distributions for the first iteration was obtained by letting $Q_k(u)$ be the same for any $k=0,1,\ldots$, and assigning probabilities that are proportional to the weight (b_u) and the number of feasible frequencies ($|F_u|$) of the corresponding site. On the other hand, following a suggestion from [17], a smoother update of parameter θ_i from the algorithm of Fig. 5 was considered by setting $\theta_i = \beta\theta_{i-1} + (1-\beta)\mu_i$, where μ_i is the solution of (13) with $\gamma_0 = \hat{\gamma}_i$, and β is a parameter that is suggested to take values between 0 and 0.3.

Also, according to Proposition 4.2 of [17], it would be desirable that the GWFAP has a unique solution, and this is why, in our experiments we considered integer weights and the slightly modified objective function

$$Z_1(x) = \sum_{i=1}^{p} \sum_{f \in S_i^x} \left(w_{if} + \frac{i}{p^2(1+M)(k_{if}+1)} \right),$$

where x is as in (6), k_{if} is the step at which frequency f was assigned to site i (the initial step is 0), and $M = \max\{|F_{v_i}| : 1 \leq i \leq p\}$ is the number of feasible frequencies in site v_i. Note that the term $\sum_{i=1}^{p} \sum_{f \in S_i^x} i/p^2(1+M)(1+k_{if})$ is less than 1, so that the optimal solution for $Z_1(x)$ is also optimal for $Z(x)$ as in (6). We remark that $Z_1(x)$ was used only to obtain a better performance for the CE algorithm, however, the results reported in the next section correspond to the original objective function $Z(x)$.

5. Experimental results

In this section, we report experimental results from the application of the proposed algorithms using two sets of problems. The first set belongs to the well known Philadelphia problems, available at http://fap.zib.de. The second set of problems is based on situations that arise when assigning FM frequencies in Mexico, and the problems were obtained using a problem generator that was developed for this specific case.

The instances of the BLP formulation of Section 2 were solved using the commercial solver CPLEX 9.0. Our interface requires, as initial parameters, the number of sites in V, their corresponding distances d_{vu}, the sets of feasible frequencies F_v, and the matrices $(M_{ij}(k))$, $k=0,1,\ldots,c$. Once the initial parameters are validated, our routine calls the CPLEX solver and reports the solution found.

Initialize the feasible sets $F_v, v \in V$

Initialize $\theta_0 \in \Theta$ **and set** $i \leftarrow 0$, $done \leftarrow$ false

While not $done$

  Set $i \leftarrow i+1$ **and generate a sample** $X_1,\ldots,X_n$ **from** $g(X;\theta_{i-1})$

  Order the values $Z(X_j)$: $Z_{(1)} \leq \ldots \leq Z_{(n)}$

  Set $\hat{\gamma}_i \leftarrow Z_{(\lceil n(1-p_0) \rceil)}, B_i \leftarrow Z_{(n)}$

  Set θ_i **equal to the solution of (12) with** $\gamma_0 = \hat{\gamma}_i$

  If $i \geq d$ **and** $B_i \leq B_{i-d}, B_{i-1} \leq B_{i-d}, \ldots, B_{i-d+1} \leq B_{i-d}$ **Then** $done \leftarrow$ true

End While

Fig. 5. Pseudo-code for the CE algorithm.

The supported interface was Microsoft Excel using macros from Visual Basic for Applications (VBA) and procedures exported from DLLs coded in Visual C++. The experiments were run in a HP workstation xw8200, with an Intel Xeon processor at 3 GHz and 3 GB of RAM memory.

5.1. Results for the Philadelphia problems

The Philadelphia problems [2] are modeled in a structure made up of 21 hexagonal sites, with distances of 1 between adjacent sites. There is data for 9 of these problems, each of which share the same hexagonal structure, but differ in the demand of frequencies for each site and the rules of minimum separation between frequencies. In our experiments, we have taken the different demands as the weights (b_i) of the sites, assuming the existence of 15 available frequencies; i.e., $F_{v_i} = T = \{1,\ldots,15\}$, $i=1,\ldots,21$.

In Table 1 we present the results obtained after using CPLEX to find the exact solution (denoted as BLP), as well as the results for the 15 heuristics proposed in Section 3 to obtain initial solutions (for the RO heuristic we report the average from 100 independent replications). To summarize the results obtained, we show in the last row of Table 1, the average percentage from the optimal value for each heuristic, and the results are illustrated in Fig. 6.

As can be observed, the RLWF, LSWF and LWF, which take into account the weight of the sites, had a better performance. However, as can be observed from Table 1, these heuristics had a relatively poor performance in problems 5 and 6, where the weights of all sites were equal.

To implement the algorithm based on cross entropy, a new parameter (M_{XE}) was used to determine the sample size, which was set at $n=M_{XE}N_0$ with $N_0 = \sum_{i=1}^{p} |F_{v_i}|$ (as in the SA algorithm). Based on previous runs, it was possible to observe that after conveniently adjusting the M_{XE} and β parameters, the CE method was capable of obtaining the optimal solution for every one of the 9 problems. Therefore, we decided to run a set of experiments using the same values of M_{XE} and β for the 9 problems. The values were set at $M_{XE}=8$ and $\beta=0.35$ because they provided the best average performance, and for these parameters, the run-times for the 9 problems oscillated between 5 and 10 s.

After obtaining the solution using the CE algorithm, the other three methods (SA, TS and GEN) were tested with a run-time limit of 10 s. In Table 2 we present the objective function values obtained for each of the four methods, and the corresponding average percentage from the optimal value are illustrated in Fig. 7. The parameters used to run the SA algorithm were $B=0.01$, $\alpha=0.9$ and $T_{\min} = 0.001$.

Table 1
Objective function values for 15 heuristics in the Philadelphia problems.

Problem	BLP	RO	LWF	LDF	LADF	SDL	SADL	LSWF	SSWF	LSDF	SSDF	LSADF	SSADF	RLWF	RLDF	RLADF
1	910	585	871	446	795	408	387	869	654	548	707	587	618	860	580	754
2	946	683	903	697	849	702	687	923	724	847	706	743	784	902	773	738
3	897	627	785	427	690	465	420	822	705	497	651	602	576	802	583	697
4	987	738	880	705	825	650	610	822	761	743	740	715	755	886	809	801
5	680	547	500	520	600	540	540	560	560	520	560	540	540	540	560	540
6	820	659	660	740	720	760	740	700	680	720	700	720	700	720	720	720
7	1820	1171	1742	892	1590	816	774	1738	1308	1096	1414	1174	1236	1720	1160	1508
8	833	553	768	579	627	677	677	739	735	678	428	594	482	814	629	735
9	3640	2342	3484	1784	3180	1632	1548	3476	2616	2192	2828	2348	2472	3440	2320	3016
Average (%)	100	71	89	64	85	64	62	91	78	72	75	72	72	91	74	82

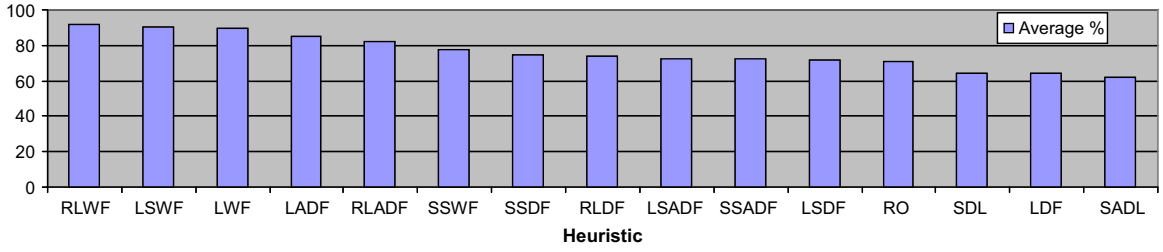

Fig. 6. Average % from the optimal value for 15 heuristics in the Philadelphia problems.

Table 2
Best solution values for the search methods in the Philadelphia problems.

Problem	Weights		Feasible freq. per site	BLP	SA	TS	GEN	CE
	Average	St. dev.						
1	22.9	18.9	15	910	895	895	895	895
2	22.9	18.9	15	946	946	919	935	946
3	22.4	11.7	15	897	875	850	870	880
4	22.4	11.7	15	987	937	942	987	987
5	20.0	0.0	15	680	680	660	620	660
6	20.0	0.0	15	820	800	740	780	820
7	45.8	37.8	15	1820	1790	1790	1790	1790
8	22.9	18.9	15	833	820	795	804	823
9	91.6	75.6	15	3640	3580	3580	3580	3580
Average % from the optimal solution				100%	98.2%	100%	97.1%	98.7%

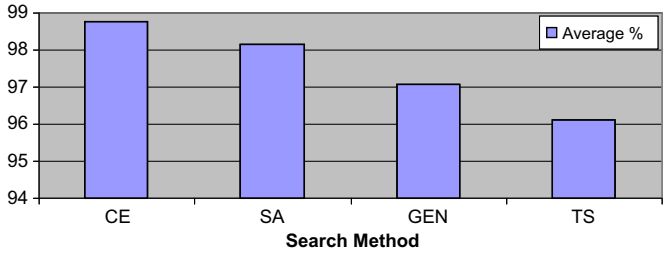

Fig. 7. Average % from the optimal value for the search methods in the Philadelphia problems.

As can be observed from Table 2, the CE algorithm obtained the best performance according to the average percentage from the optimal value, followed closely by the SA algorithm. We should mention that that the SA algorithm was stopped before convergence in all 9 problems.

5.2. Results for the FM frequency assignment problem of Mexico

These sets of problems are similar to those encountered when assigning FM frequencies in different sites within Mexico. Contrary to the Philadelphia problems, an important feature of these problems

Table 3
Average run-times using CPLEX and feasible frequencies for different number of sites.

Number of sites (n^2)	4	9	16	25	36	49	64
Feasible frequencies per site	48.6	45.21	40.23	40.73	39.72	38.98	38.41
Average time (s)	0.1	0.3	1.2	5.1	81.9	532.35	5213.6

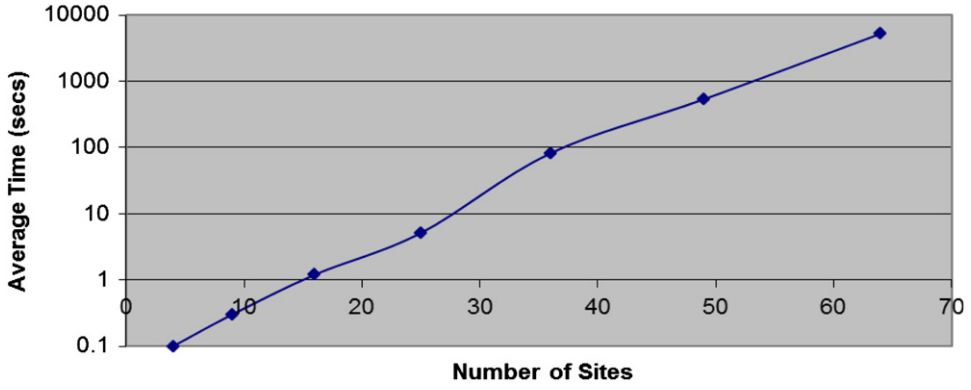

Fig. 8. Average run-time (on a logarithmic scale) using CPLEX as a function of the number of sites.

Table 4
Objective function values for 15 heuristics in the Mexico problems with 64 sites.

Problem	BLP	RO	LFW	LDF	LADF	SDL	SADL	LSWF	SSWF	LSDF	SSDF	LSADF	SSADF	RLWF	RLDF	RLADF
1	1306	781	1110	777	795	702	698	906	999	875	806	875	806	1166	929	922
2	1336	795	1083	772	801	696	673	931	1046	948	785	948	785	1113	929	935
3	1325	795	1065	756	774	693	652	979	983	971	828	971	828	1127	921	950
4	1326	798	1093	753	819	658	675	1036	1055	926	809	926	809	1151	971	1002
5	1308	806	1080	787	824	716	717	883	1018	900	769	900	769	1135	925	923
6	1329	779	1114	812	853	713	677	981	922	886	857	886	857	1145	932	932
7	1291	802	1068	735	742	673	669	961	991	882	785	882	785	1113	951	935
8	1292	795	1055	802	782	700	673	1005	959	912	677	912	677	1090	901	888
9	1334	806	1086	754	800	685	683	890	1060	822	749	822	749	1165	897	878
10	1328	815	1080	760	792	722	703	917	1097	894	844	894	844	1157	946	936
Average (%)	100	61	82	59	61	53	52	72	77	68	60	68	60	86	71	71

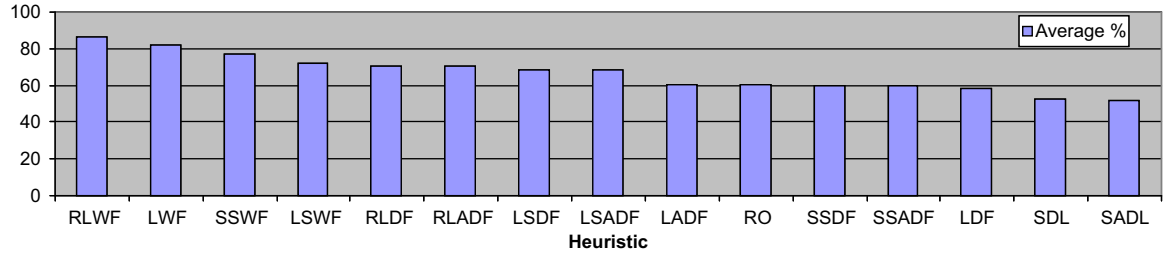

Fig. 9. Average % from the optimal value for 15 heuristics in the Mexico problems with 64 sites.

is that some frequencies have already been assigned and are being used with different types of transmitting antennas. Therefore, the objective of these problems is to maximize the efficient use of the remaining frequencies while taking into account that each assignment has a different benefit according to the site (most likely given by the demand or economic value of a frequency in the site).

A problem generator was developed by taking into account the main characteristics of this type of problem. The generator begins by setting the coordinates of n^2 sites v_{ij}; $i,j=1,\ldots,n$ (where n is a given integer) from the coordinates (LAT_1,LON_1) of site v_{11}, and the amplitudes $ALAT$ and $ALON$ (these parameters are user-specified), so that the coordinates of site v_{ij} are (LAT_i, LON_j), where $LAT_i=LAT_1+(i-1)ALAT$, and $LON_j=LON_1+(j-1)ALON$. Then, a type of antenna (from six different options) and the weight (b_i) of every site are determined randomly, along with the frequencies that have already been assigned to each site (chosen from 100 different possible frequencies). Finally, the distances between each site are calculated and we also obtain which of the 100 available frequencies are feasible in each site, taking into account the types of antennas, the already assigned frequencies and the rules of separation between frequencies in Mexico. In all of our experiments, the coordinates of site v_{11} were $LAT_1=N19.5^\circ LONC=W89.5^\circ$ and the amplitudes were $ALON=0.52^\circ$ and $ALAT=0.52^\circ$.

Since we are interested in solving large problems, we first run a set of experiments to investigate the time that was required to obtain an exact solution using CPLEX. For a given value of n, we considered a problem with n^2 sites (as explained before), and 10 replications of the problem were generated by assigning (at random) a feasible frequency to each installed antenna. In Table 3 we present the average run-time (in seconds) for $n=2,\ldots,8$. For $n^2=81$ sites the size of the problems exceeded the memory of our workstation; however, we can see from Fig. 8 that the run-time is exponential in the number of sites.

In a second set of experiments we compared the performance of the 15 heuristics using the largest problems that we solved using CPLEX ($n^2=64$ sites). In Table 4 we present the exact solution using CPLEX (denoted as BLP) and the results for the 15 heuristics for each of 10 replications with 64 sites. These results show that, the RLWF and LWF heuristics have a very good performance. Note that the RLFW heuristic performed the best in all 10 replications (see also Fig. 9).

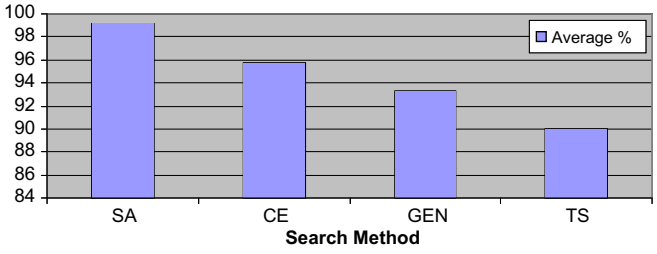

Fig. 10. Average % from the optimal value for the search methods in the Mexico problems with 64 sites.

After some experiments using the SA and the CE algorithms in the Mexico problems with 64 sites, we observed that the SA and the CE algorithms had similar run-times. Using the same parameters as in the Philadelphia problems, the run-times for the SA algorithm oscillated between 660 and 750 s in all of 10 replications, and this why we decided to test the other three methods (CE, TS and GEN) under a run-time limit of 750 s. The parameters for the CE algorithm were set at $\beta=0.2$ and $n=1000$, since they provided the best average performance. In Table 4 we present the objective function values obtained for each of the four methods, and the corresponding average percentage from the optimal value are illustrated in Fig. 10. As we see from these results, the SA algorithm performed significantly better than the others in all 10 replications. The average run-time for the CE and the SA algorithms were 423 and 628 s, respectively.

Finally, we tested the search algorithms in the Mexico problems with 400 sites. Since we did not observe convergence for the SA and the CE algorithms, we decided to set a run-time limit of 3600 s. For the SA algorithm we considered the same parameters as in the Philadelphia problems, and for the CE algorithm we considered $\beta=0.1$ and $n=1000$. As we see from the results of Table 5 and Fig. 11, the SA algorithm performed significantly better than the others in all 10 replications, and in this case, the TS algorithm performed better than the CE algorithm Table 6.

6. Conclusions

The results obtained from testing 15 heuristics for initial solutions suggest that a good heuristic should consider the weights of the sites when assigning frequencies in the GWFAP. The LWF, RLWF and LSWF heuristics performed considerably better than the others because their frequency assignment is heavily influenced by the site weights. In particular, the RLWF heuristic performed the best in all instances of the large (Mexico) problems.

Our experimental results suggest that the SA implementation performs better than the other three implementations when the instance of the GWFAP is large. The overall performance of the SA algorithm was impressive in the Mexico problems with 64 sites, where it obtained an average percentage from the optimal solution of 99.2%.

Table 5
Best solution values for the search methods in the Mexico problems with 64 sites and run-time of 750 s.

Problem	Weights		Feasible freq. per site	BLP	SA	TS	GEN	CE
	Average	St. dev.						
1	1.83	2.04	37.06	1306	1292	1216	1218	1256
2	1.83	2.04	39.19	1336	1329	1176	1247	1284
3	1.83	2.04	38.11	1325	1315	1174	1241	1262
4	1.83	2.04	38.58	1326	1312	1205	1238	1267
5	1.83	2.04	39.27	1308	1301	1182	1210	1249
6	1.83	2.04	39.20	1329	1316	1195	1244	1269
7	1.83	2.04	37.27	1291	1275	1155	1218	1244
8	1.83	2.04	37.06	1292	1284	1145	1201	1232
9	1.83	2.04	40.52	1334	1325	1214	1236	1263
10	1.83	2.04	37.83	1328	1317	1188	1233	1282
Average % from the optimal value				100%	99.2%	89.9%	93.3%	95.7%

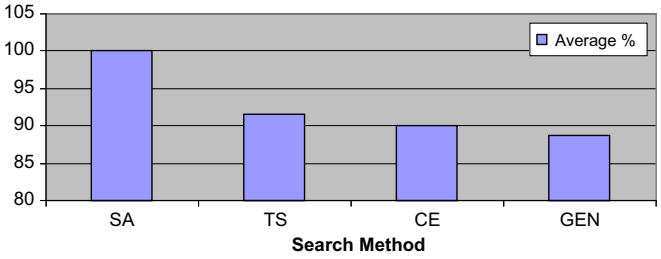

Fig. 11. Average % from the best solution for the search methods in the Mexico problems with 400 sites.

Table 6
Best solution values for the search methods in the Mexico problems with 400 sites and run-time of 3600 s.

Problem	Weights		Feasible freq. per site	SA	TS	GEN	CE
	Average	St. dev.					
1	1.62	1.87	32.36	6894	6323	6152	6196
2	1.62	1.87	31.67	6814	6247	6030	6185
3	1.62	1.87	32.25	6912	6263	6135	6136
4	1.62	1.87	32.27	6847	6282	6132	6200
5	1.62	1.87	31.37	6881	6268	6076	6144
6	1.62	1.87	32.41	6908	6309	6068	6228
7	1.62	1.87	32.32	6826	6282	6062	6238
8	1.62	1.87	31.76	6852	6290	6017	6138
9	1.62	1.87	32.34	6869	6338	6188	6253
10	1.62	1.87	32.64	6876	6252	6093	6130
Average % from the optimal value				100.0%	91.5%	88.8%	90.1%

Although the CE algorithm reported the best performance in the small (Philadelphia) problems, it performed in second place for the Mexico problems with 64 sites, and in third place for the Mexico problems with 400 sites. Note that the generation of solutions in the GA and the CE algorithms requires a sampling procedure so that the time to produce a new solution increases with the problem size. In contrast, the generation of a new solution under the SA and TS algorithms does not require a sampling procedure, and this can explain why the performance of the CE and the GA algorithms deteriorates with the size of the problem.

Finally, we would like to remark that our experimental results apply to the particular implementations of the four algorithms (SA, CE, TS and GA) that we tested. A different implementation of any of these algorithms may provide different conclusions. In particular, a faster generation of new solutions may improve the performance of any implementation.

Acknowledgments

Support provided from the Asociación Mexicana de Cultura A.C. is highly appreciated. The authors would like to express their sincere gratitude to the Associate Editor and two anonymous Referees for their valuable hints and suggestions.

References

[1] Aardal KI, van Hoesel CPM, AMCA Koster, Mannino C, Sassano A. Models and solution techniques for frequency assignment problems. Annals of Operations Research 2007;153(1):79–129.
[2] Anderson LG. A simulation study of some dynamic channel assignment algorithms in a high capacity mobile telecommunications system. IEEE Transactions on Vehicular Technology 1973;22(4):210–217.
[3] Chiarandini M, Stützle T. Stochastic local search algorithms for graph set T-colouring and frequency assignment. Constraints 2007;12(3):371–403.
[4] Chow FC, Hennesy JL. The priority-based coloring approach to register allocation. ACM Transactions on Programming Languages and Systems 1990;12(4):501–536.
[5] Costa D. On the use of some known methods for T-colorings of graphs. Annals of Operations Research 1993;41(4):343–358.
[6] De Werra D. An introduction to timetabling. European Journal of Operations Research 1985;19(2):151–162.
[7] Eisenblätter A, Grötschel M. Koster AMCA. Frequency assignment and ramifications of coloring. Discussiones Mathematicae Graph Theory 2002;22(1):51–88.
[8] Garey MR, Johnson DR. Computers and Intractability: A Guide to the Theory of NP-Completeness. San Francisco: Freeman; 1979.
[9] Glover F, Taillard E, De Werra D. A user's guide to tabu search. Annals of Operations Research 1993;41(1):3–28.
[10] Goldberg D. Genetic Algorithms in Search, Optimization and Machine Learning. New York: Addison Wesley; 1989.
[11] Hale WK. Frequency assignment: theory and applications. Proceedings of the IEEE 1980;68:1497–1514.
[12] Hurley S, Smith D. Meta-heuristics and channel assignment. In: Leese R, Hurley S, editors. Methods and Algorithms for Radio Channel Assignment. USA: Oxford University Press; 2002. p. 22–44.
[13] Johnson DS, Mehrotra A, Trick MS, editors. Discrete Applied Mathematics, 156; 2008. p. 2.
[14] Kolen A. A genetic algorithm for the partial binary constraint satisfaction problem: an application to a frequency assignment problem. Statistica Neerlandica 2007;61(1):4–15.
[15] Metzger BH. Spectrum management technique. Presentation in the 38th National ORSA Meeting, Detroit, USA, 1970.
[16] Ortiz-García EG Pérez-Bellido AM. Hybrid cross-entropy method/Hopfield neural network for combinatorial optimization problems. In Proceedings of the Intelligent Data Engineering and and Automated Learning-IDEAL 2007; p. 1160–1169.
[17] Rubinstein RY, Kroese DP. The Cross-Entropy Method. A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning. New York: Springer; 2004.
[18] Tesman BA. Set T-colorings. Congressus Numerantium 1990;77:229–242.
[19] Valenzuela CL. A study of permutation operators for minimum span frequency assignment using an order based representation. Journal of Heuristics 2001;7(1):5–21.