EURO Journal on Computational Optimization 10 (2022) 100024

Contents lists available at ScienceDirect

EURO Journal on Computational Optimization

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

Multi-Neighborhood simulated annealing for the minimum interference frequency assignment problem

Sara Ceschia, Luca Di Gaspero, Roberto Maria Rosati, Andrea Schaerf*

DPIA, Università degli Studi di Udine, Via delle Scienze 206, I-33100, Udine, Italy

ARTICLE INFO

Keywords:
Frequency assignment
Neighborhood search
Simulated annealing
Parameter tuning

ABSTRACT

We consider the Minimum Interference Frequency Assignment Problem and we propose a novel Simulated Annealing approach that makes use of a portfolio of different neighborhoods, specifically designed for this problem.

We undertake at once the two versions of the problem proposed by Correia (2001) and by Montemanni et al. (2001), respectively, and the corresponding benchmark instances. With the aim of determining the best configuration of the solver for the specific version of the problem we perform a comprehensive and statistically-principled tuning procedure.

Even tough a totally precise comparison is not possible, the experimental analysis show that we outperform all previous results on most instances for the first version of the problem, and we are at the same level of the best ones for the second version.

As a byproduct of this research, we designed a new robust file format for instances and solutions, and a data repository for validating and maintaining the available solutions.

1. Introduction

Allocating radio spectrum resources is a crucial problem in the design and operation of mobile communication networks. In particular, the Frequency Assignment Problem (FAP) consists in assigning, in an efficient way, a limited number of frequencies to communication links. In this work, among all the possible frequency assignment models, we consider the Minimum Interference Frequency Assignment Problem (MI-FAP), which is the most studied model, mostly due to its practical importance. The MI-FAP aims at allocating frequencies so that interferences between adjacent frequencies in geographically close links are minimized.

We undertake the two versions of the problem proposed in the COST 259 project (Correia, 2001) and by Montemanni et al. (2001), respectively, as they both come along with a very challenging dataset, that have been used as benchmark for many studies.

Even though the technology has evolved substantially since these formulations have been proposed, the frequency assignment problem remains essential also on more recent environments, such as 5G (Lin et al., 2015), edge computing (Zhang et al., 2020), D2D networks (Zhao et al., 2018), and military applications (Lal et al., 2018; Wang and Henz, 2017). However, no specific new formulation and benchmark has emerged so far, therefore these datasets remain the most popular and challenging benchmarks.

For the solution of this problem we have designed and implemented a local search approach based on a suitable combination of different neighborhoods, driven by a Simulated Annealing (SA) procedure. SA has been used also by many other authors for FAP with good results, suggesting that it is particularly suitable for this type of problems (see Section 3 on Related work). In addition, we have experienced good results with SA also on problems that are quite similar to FAP, like, for example, Examination Timetabling (Bellio et al., 2021).

In order to obtain the best configuration of the algorithm for the specific problem versions and datasets, we have tuned the algorithm using a comprehensive and statistically-principled tuning procedure.

We also performed an extensive experimentation with the aim of comparing our results with previous literature. Although a totally precise comparison is not possible, the outcomes of the experiments show that our solution method is able to outperform all those reported for the version of Correia (2001) for most of the instances and to reach the same level of the best results for the version of Montemanni et al. (2001), on the respective datasets. In addition, we have reached many new best-known solutions for the COST 259 dataset.

Finally, in order to foster future comparisons, we published our solutions on the web. To this aim, we created a novel data format for both input and output files based on JSON. All data is available for inspection and comparison at https://opthub.uniud.it. Our repository is conceived in such a way that also other researchers can validate and

* Corresponding author.
E-mail addresses: sara.ceschia@uniud.it (S. Ceschia), luca.digaspero@uniud.it (L. Di Gaspero), robertomaria.rosati@uniud.it (R.M. Rosati), andrea.schaerf@uniud.it (A. Schaerf).

https://doi.org/10.1016/j.ejco.2021.100024
Received 20 July 2021; Received in revised form 15 November 2021; Accepted 21 December 2021

upload their solutions, so that they become immediately available for everybody along with the timestamp of the upload.

Summarizing, the main contributions of the paper are:

- an SA procedure based on a novel combination of neighborhoods for two classical versions of MI-FAP;
- a statistically-principled tuning procedure;
- a comparison with state-of-the-art results showing that the method is very effective for both versions and different time limits;
- publication of instances and solutions on the web in a new file format, along with the validator to encourage future comparisons.

2. Problem definition

For the sake of completeness, we describe here informally the two versions of the MI-FAP problem, and we forward the reader to Correia (2001) and Montemanni et al. (2001) for more details on the specific formulations and for the precise mathematical models.

We proceed by firstly describing the problem version proposed within the COST 259 project, and then we move to the simpler variant proposed by Montemanni et al. (2001). In the following, for conciseness, we refer to the two versions as MI-FAP-I and MI-FAP-II, respectively.

The key elements of MI-FAP-I are the following ones:

Cells: a cell is an equipment that provides communication service to a given geographical area and has a fixed number of transmitters.

Transmitters: a transmitter is a single device that transmits the signal, and requires the assignment of a frequency. For each cell, one of the transmitters is designated to carry the control signal whereas the others carry traffic signals. The control signal needs special treatment in terms of *separation constraints*, as explained below (see *handover* separation rule).

Frequencies: a fixed number of frequencies (or channels) are available, each one identified by an integer value. Frequencies that are adjacent in terms of actual transmission bandwidth are assigned to consecutive values. Some frequencies can be forbidden either for a given cell or globally for the entire area.

Sites: a site is a physical installation where several cells (typically three) are located.

In order to avoid disturbance in the communication, the assignment of frequencies to transmitters has to satisfy a set of separation rules:

Co-cell: transmitters belonging to the same cell must have a given frequency separation. The typical separation value is 3 and it is defined at global level, but some cells can have specific (and different) separation requirements.

Co-site: transmitters belonging to cells at the same site must have a given frequency separation, typically 2.

Handover: cells that are geographically adjacent might suffer from the so-called handover effect, therefore requiring a specific separation among the transmitters of those cells on the basis of their specific type (i.e., control / traffic). Typically, there must be a separation of 2 between the control channels, a separation of 1 or 2 between a control channel and a traffic channel, and a separation of 1 between traffic channels.

Additional separations: ad hoc separations between pairs of cells due to specific conditions might be required, and they have to be enforced on all pairwise combinations of transmitters of the two cells involved.

In addition to these mandatory channel separations, which must be always fulfilled (i.e., they are so-called *hard constraints*), also a milder interference of frequencies between cells has to be taken into account in terms of an objective function to be minimized. In detail, for each pair of cells the *interference cost* that occurs in case of assignment to the same frequency (*same-channel* interference) or to adjacent (i.e., whose distance is 1) frequencies (*adjacent-channel* interference) is specified as a pair of real-valued numbers.

When the cells are geographically apart, there is no interference, so there will be no penalty for assigning the same frequency or an adjacent one. It might also be possible that only the same-channel interference is relevant and the adjacent-channel interference is zero (the opposite is obviously not possible). In this version of the problem, no interference penalty is ever assigned in case of frequencies at distance 2 or more.

The specification for MI-FAP-II is much simpler, as there are no explicit notions of cell, site, and handover. Constraints are expressed directly at the transmitter level, by specifying the required separation between pairs of transmitters and the cost of its violation. The penalty for violating the separation is fixed, independently of the degree of violation. This is different from MI-FAP-I in which same- and adjacent-channel interferences are weighted differently. Finally, there is no explicit distinction between hard and soft constraints, though, in some instances, there are separations with an extremely high cost (10^8), that we interpreted as a hard one.

3. Related work

The literature on Frequency Assignment is very vast. For this reason, we focus our overview specifically on the Minimum Interference formulation. We refer to the comprehensive and enlightening survey by Aardal et al. (2007) for the general problem and to the FAP website (Eisenblätter and Koster, 2000) for other publications, benchmark instances, and results (unfortunately not very up-to-date).

The MI-FAP originates from the study by Allen et al. (1987) and was thereafter investigated in many other works (Aardal et al., 1996; Björklund et al., 2005; Borndörfer et al., 1998; Crisan and Mühlenbein, 1998; Duque-Antón et al., 1993; Kapsalis et al., 1995; Kolen, 2007; Koster et al., 1999; 2002; Tiourine et al., 2000), mainly in connection to the CALMA project (Aardal et al., 2002).

The MI-FAP-I formulation emerged within the COST 259 project "Wireless Flexible Personalised Communications" (Correia, 2001), which involved more than 200 European research institutions and companies in the area of mobile radio during the years from 1996 to 2000. One of the contributions of the project was a dataset of 32 realistic instances, which has become well-known and a very challenging benchmark for GSM network planning (see Section 5.1 for details about this dataset).

Over the years, MI-FAP-I has been mainly tackled by metaheuristic methods, because large-size scenarios of the COST 259 dataset are still beyond the reach of exact approaches. Among the metaheuristics methods, a Simulated Annealing (SA) approach for MI-FAP-I was firstly proposed by Beckmann and Killat (1999). This approach relies on a cell-based local search neighborhood with some restrictions: a cell is randomly selected, then the frequency of the transmitter (within the cell) with the largest interference cost is substituted with a new permitted frequency causing the smallest interference cost.

Hellebrandt and Heller (2000) apply a variant of SA, the *threshold accepting* algorithm, where a new solution is accepted if the deterioration of the value of the objective function is less than a given threshold, which is reduced during the search process. They implement a basic neighborhood (i.e., the one that changes the frequency to a single transmitter) but forbidding those moves that produce violations of the hard constraints. In addition, they also employ at each iteration a *one-cell re-optimization* process, by means of a dynamic program that performs a simultaneous exchange of all the frequencies assigned to a cell when this improves the current solution.

The dynamic programming method has been generalized to cliques of vertices by Mannino et al. (2007), who also employ a Simulated Annealing algorithm as their main search procedure. They also show that, for some restricted cases under some specific hypothesis on the subsets of transmitters, the MI-FAP can be reduced to a maximum

weighted stable set problem, which is solvable in polynomial time. This theoretical result has been exploited to search effectively in a large-scale neighborhood, defined as the set of all transmitters whose frequency can be simultaneously replaced without incurring in any violation.

Montemanni et al. (2003) propose a Tabu Search procedure with a dynamic length tabu list in which the neighborhood relation changes the frequency of a single transmitter involved in at least one constraint violation. They also implement cell re-optimization by means of a recursive depth-first search procedure.

More recent works on MI-FAP-I deal with multi-objective optimization variants of the problem (Aardal et al., 2007). In particular, besides the minimization of the total interference, Laidoui et al. (2018) studied the trade-off between interference and the blocking probability, as a function of the number of frequencies assigned to each cell. Instead, Kiouche et al. (2020) dealt with the problem of simultaneously minimizing also the maximum interference and the number of frequencies used. In both cases, the authors implemented genetic algorithms hybridized by combining elements related to game theory for Laidoui et al. (2018) and to Artificial Immune Systems for Kiouche et al. (2020).

The MI-FAP-II, also known as *Fixed-Spectrum Frequency-Assignment Problem*, was introduced by Montemanni et al. (2001) as a generalization of the Graph Coloring Problem. The authors propose different lower bounding techniques that are tested on a new dataset, whose main characteristics are discussed in Section 5.1. Lower bounds for MI-FAP-II have been further improved in (Montemanni et al., 2004). For the solution of MI-FAP-II, a number of effective Tabu Search algorithms have been proposed by Montemanni et al. (2003), Montemanni and Smith (2010), and Lai and Hao (2015). In particular, Lai and Hao (2015) devise a population based strategy with relinking operators tailored to MI-FAP-II, which were able to create solution paths connecting the two high-quality solutions and generate new promising solutions. The Tabu Search algorithm of Montemanni and Smith (2010) has also been tested on a small subset of the MI-FAP-I dataset.

Segura et al. (2016) developed an evolutionary algorithm with a diversification strategy to avoid premature convergence of the population. This was obtained by converting MI-FAP-II to a multi-objective problem that considers the original objective and, as an auxiliary objective, the contribution of each individual to the diversity. The solution method was evaluated both on MI-FAP-II and on a complex formulation, which arose from two real-world instances coming from the cities of Denver and Seattle (Luna et al., 2007; 2011). In this new formulation, there is no notion of sites and separations involve only transmitters in the same cell (co-cell). Analogously to MI-FAP-I the interference matrix is defined at the cell level, but interferences are not given explicitly and they are computed through a probabilistic model.

Similarly to Lai and Hao, Siddiqi and Sait (2018) proposed a population-based heuristic that employs Tabu Search to drive the exploration of the neighborhood of each solution in the population. The search process is guided by the principles of non-dominated sorting, considering both the interference and the entropy criteria (as a measure of the diversity of individuals of a population).

Finally, Lahsinat et al. (2018) developed a Variable Neighborhood Search (VNS) that explores increasingly large neighborhoods, from the smallest one that changes the frequency of a single transmitter, to the largest that changes simultaneously the frequency of 5 transmitters. In addition, the authors introduced different perturbation schemes for helping the VNS process to escape from local optimum.

Although many works on this topic used neighborhood search, and specifically Simulated Annealing, to tackle this problem our contribution distinguishes from existing literature mainly in the following two aspects. First of all, differently from the surveyed approaches, we investigate the use of the combination of complex neighborhood structures for solving the problem. Secondly, no previous approach dealt with the MI-FAP-I and MI-FAP-II formulations in a comprehensive way.

4. Search method

Our search method is based on local search, therefore we now introduce, step by step, the four key ingredients of the application of the local search paradigm, namely: *(i)* the search space definition, *(ii)* the initial solution strategy, *(iii)* the neighborhood operators, and, finally, *(iv)* the metaheuristic that guides the search.

Before proceeding we introduce some notation and terminology that will be useful to illustrate these concepts. We consider the graph in which each single transmitter is taken separately, without their aggregation in cells and sites. This graph is called the *split graph* by Chiarandini and Stützle (2007). Following the graph-coloring terminology, from this point on we will call the transmitters as *nodes*.

We are then given a set of nodes $\mathcal{N} = \{1, \dots, N\}$ and a set of frequencies $\mathcal{F} = \{1, \dots, F\}$. We call $\mathbf{S}$ the integer-valued $N \times N$ matrix such that $\mathbf{S}_{n_1,n_2}$ is the required separation between nodes n_1 and n_2. We are also given for each node n a set of frequencies U_n, representing the forbidden frequencies for node n, with $U_n \subset \mathcal{F}$ for all n.

Given these preliminaries we are now able to illustrate the key features of the local search.

4.1. Search space and initial solution

The search space is represented by an integer-valued array φ, so that $\varphi(n)$ is the frequency assigned to node $n \in \mathcal{N}$.

The array φ is complemented by redundant data structures that help us in accelerating the computation of the difference of costs between neighboring solutions (we call them *delta* costs). The main data structure, which has also been used by Chiarandini and Stützle (2007), is an integer-valued matrix $\mathbf{\Gamma}$ that stores, for each pair $\langle n, f \rangle$, the number of conflict violations that would be created by reassigning node n to frequency f in the current state. In addition, we maintain an array of sets Λ that stores, for each node n, the set of nodes that are in conflict (i.e., violated separation) with n in the current state.

For MI-FAP-I, that has real-valued interferences, in order to exploit the faster arithmetic of the integers, we multiply all interference values by a fixed number, suitably high so as not to lose precision (10^8). This is not necessary for MI-FAP-II, for which the values are natively integers.

The initial solution is generated by assigning one node at the time a uniformly-selected random frequency, among those that are not forbidden for that node. That is, separation violations are admitted, but forbidden frequency violations are not. Indeed, forbidden frequency violations are kept outside the search space, as all neighborhoods discussed below do not include moves that reassign a node n to a frequency in U_n.

4.2. Neighborhood relations

The typical neighborhood relation used in FAPs is the replacement of the frequency assigned to one node. We call this neighborhood Change, which is defined as follows:

- Change(C): the move $\mathsf{C}\langle n, f \rangle$ assigns frequency f to node n.
  Preconditions: $\phi(n) \neq f, \quad f \notin U_n$.

The preconditions state that a node must be assigned to a *new* frequency ($\phi(n) \neq f$) and that it cannot be a forbidden one ($f \notin U_n$). Moves that do not satisfy the preconditions are removed from the neighborhood, and thus never drawn.

Similarly to the approach followed for the Examination Timetabling problem, which has a similar structure (see Bellio et al., 2021), we complement this basic neighborhood with larger ones that allow us to make more complex movements in one single step. The first one is the so-called Kick move that reallocate two nodes simultaneously, assigning the first one to the frequency of the second one, and the second one to a new frequency.

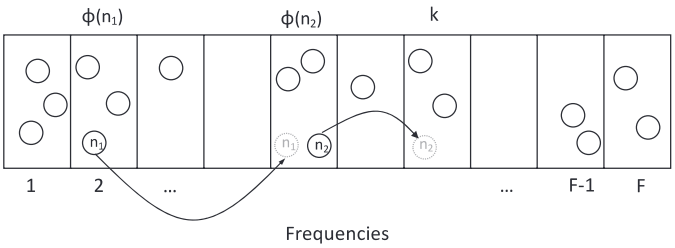

Fig. 1. A Kick move.

- Kick(K): the move $\mathsf{K}\langle n_1, n_2, f\rangle$ assigns $\varphi(n_2)$ to n_1 and f to n_2.
  Preconditions: $\mathbf{S}_{n_1,n_2} > 0$, $\varphi(n_1) \neq \varphi(n_2)$, $\varphi(n_2) \neq f$, $\varphi(n_2) \notin U_{n_1}$, $f \notin U_{n_2}$.

The intuition of the Kick neighborhood is to move a node to a potentially favorable frequency even in presence of a conflicting node, given that at the same time the second one is "kicked out".

Fig. 1 shows graphically an example of a Kick move, in which the new assignments are shown in light grey.

The precondition that n_1 and n_2 must have a separation ($\mathbf{S}_{n_1,n_2} > 0$) is meant to restrict Kick moves only to those that are most effective to overcome a cost barrier, with respect to others that could be obtained by a sequence of two cost-independent Change moves. The other preconditions ensure that the move is effective and it does not assign forbidden frequencies.

Notice that f, the new frequency for n_2, could also be equal to $\varphi(n_1)$, resulting in swapping the two assignments. As a consequence, the Kick neighborhood is a superset of the Swap neighborhood, also used in the literature (see Galinier and Hertz, 2006).

The Kick neighborhood is typical for graph coloring problems (see, e.g., González-Velarde and Laguna, 2002). However, FAPs have the peculiarity that conflicts can occur also between nodes assigned to different frequencies. In order to deal with this situation, we propose an extension of the Kick neighborhood, that we call GKick (G for generalized), that moves the first node to a frequency *close* to the second node, and moves the second node away.

- GKick(G): the move $\mathsf{G}\langle n_1, f_1, n_2, f_2\rangle$ assigns f_1 to n_1 and f_2 to n_2.
  Preconditions: $\mathbf{S}_{n_1,n_2} > 0$, $\varphi(n_1) \neq \varphi(n_2)$, $\varphi(n_1) \neq f_1$, $\varphi(n_2) \neq f_2$, $f_1 \notin U_{n_1}$, $f_2 \notin U_{n_2}$, $|f_1 - \varphi(n_2)| < S_{n_1,n_2}$.

The behavior of the GKick neighborhood is analogous to the one of the Kick neighborhood, except that n_1 can be assigned to a frequency f_1 that differs from $\varphi(n_2)$, but is close enough to it that still creates a separation conflict with n_2 (see precondition $|f_1 - \varphi(n_2)| < S_{n_1,n_2}$).

Like for the Kick neighborhood, we can identify here a subset of the neighborhood that we call GSwap in which f_2 is close to $\varphi(n_1)$, in particular closer than the separation between n_1 and n_2.

As will be shown in Section 5.2, the *subneighborhoods* Swap and GSwap play a prominent role in the search. In detail, we will see that it is more effective to bias strongly the random move generation of Kick (resp. GKick) toward Swap (resp. GSwap) moves, rather than to *pure* kicks. For this reason, we include in our neighborhood portfolio a larger one, called 3-Swap, that involves 3 nodes, but makes only swap movements.

- 3-Swap(T): the move $\mathsf{T}\langle n_1, n_2, n_3\rangle$ assigns $\varphi(n_2)$ to n_1, and $\varphi(n_3)$ to n_2, and $\varphi(n_1)$ to n_3.
  Preconditions: $\mathbf{S}_{n_1,n_2} > 0$, $\mathbf{S}_{n_2,n_3} > 0$, $\varphi(n_1) \neq \varphi(n_2)$, $\varphi(n_2) \neq \varphi(n_3)$, $\varphi(n_3) \neq \varphi(n_1)$, $\varphi(n_1) \notin U_{n_2}$, $\varphi(n_2) \notin U_{n_3}$, $\varphi(n_3) \notin U_{n_1}$.

The intuition for introducing the 3-Swap neighborhood is that, on the one hand, we aim at exploring larger neighborhoods, and on the other hand a general 3-node kick (either generalized or not) would be too large, and thus practically ineffective. Therefore, considering the usefulness of the bias toward swap movements for the Kick and GKick neighborhoods mentioned above, we decided that it is wiser to focus only on swaps, and thus the 3-Swap neighborhood seems to be a good trade-off.

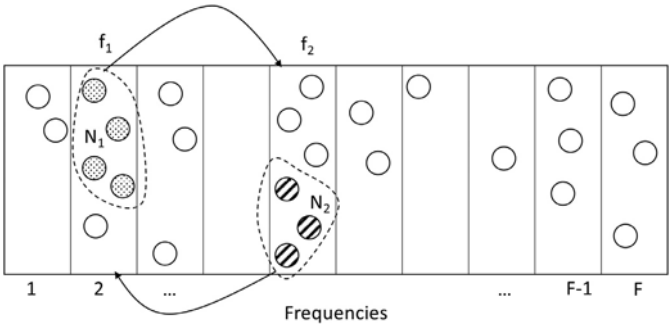

Fig. 2. A PFSwap move with $k = 7$.

Our next neighborhood, called FSwap (F for Frequency), swaps all nodes assigned to one frequency with all assigned to another one.

- FSwap(F): the move $\mathsf{F}\langle f_1, f_2\rangle$ assigns all nodes n such that $\varphi(n) = f_1$ to f_2, and all nodes n such that $\varphi(n) = f_2$ to f_1
  Preconditions: $f_1 \neq f_2$

The FSwap neighborhood is used in graph coloring and its intuition is that the simultaneous movement of all nodes assigned to the same color (frequency in our case) does not increase the number of violations This move however might be less effective in FAPs as, differently from graph coloring, the conflicts come also from nodes in adjacent frequencies. For this reason we define also a partial version of the same move, that swaps a limited number of nodes so that it results less disruptive. We thus design the neighborhood called PFSwap (P for Partial).

- PFSwap(P): the move $\mathsf{P}\langle f_1, f_2, N_1, N_2\rangle$ assigns all nodes $n_1 \in N_1$ to f_2, and all nodes $n_2 \in N_2$ to f_1
  Preconditions: $f_1 \neq f_2$, for all $n_1 \in N_1$ we have $\varphi(n_1) = f_1$, and for all $n_2 \in N_2$ we have $\varphi(n_2) = f_2$.

The selection of N_1 and N_2 is random, except that the cardinalities are kept equal or differ by one. That is, the selection of the sets is preceded by the selection of a number k, and the number of nodes in N_1 and N_2 is $\lceil k/2 \rceil$ and $\lfloor k/2 \rfloor$, respectively. The value of k is selected between 3 and a fixed maximum value K_m.

Fig. 2 shows graphically an example of a PFSwap move (with $k = 7$).

4.3. Move selection

We now describe how we select a random move from our composite neighborhood: Change ∪ Kick ∪GKick ∪ 3-Swap ∪ FSwap ∪PFSwap. This is done in two stages: first we select the atomic neighborhood and then the specific move inside the neighborhood. The first selection is based on fixed probabilities. That is, we add five real-valued parameters called p_K, p_G, p_T, p_S, and p_P, such that at each step neighborhoods Kick, GKick, 3-Swap, FSwap, PFSwap, and Change are selected with probability p_K, p_G, p_T, p_S, p_P, and $1 - p_K - p_G - p_T - p_S - p_P$, respectively.

Within the single neighborhood, the specific move is selected uniformly, except for Kick and GKick where a move from the Swap (resp. GSwap) subneighborhood is selected with probability b_s (resp. b_g) and a pure kick is selected with probability $1 - b_s$ (resp. $1 - b_g$). This way, we add two new parameters, b_s and b_g, that are included to the pool of parameters. In turn, the PFSwap neighborhood is parameterized by the value of the parameter K_m mentioned above representing the maximum number of nodes involved in the move.

```
procedure SimulatedAnnealing(SearchSpace S, Neighborhood N,
                             CostFunction F, Parameters T_0, T_f, α, N_s, N_a)
1:    T ← T_0
2:    s ← RandomState(S)
3:    s_best ← s
4:    while T ≥ T_f
5:        n_s ← 0
6:        n_a ← 0
7:        while n_s < N_s ∧ n_a < N_a
8:            m ← RandomMove(s, N)
9:            ΔF ← F(s ⊕ m) − F(s)
10:           if (ΔF ≤ 0)
11:               s ← s ⊕ m
12:               n_a ← n_a + 1
13:               if (F(s) < F(s_best))
14:                   s_best ← s
15:           else
16:               if (RandomReal(0, 1) < e^{−ΔF/T})
17:                   s ← s ⊕ m
18:                   n_a ← n_a + 1
19:           n_s ← n_s + 1
20:       T ← T · α
21:   return s_best
```

Fig. 3. Simulated Annealing procedure.

4.4. Simulated Annealing

Many variants of the SA procedure have been proposed in the literature, see Franzin and Stützle (2019) for a comprehensive and in-depth review of them. We use here a basic version, as originally proposed by Kirkpatrick et al. (1983), shown in Fig. 3.

One of the key ingredients of SA is the random nature of the move selection at each iteration (line 16), as explained in Section 4.3. Improving and sideways moves are always accepted as new state (line 10), whereas worsening ones are accepted with probability $e^{-\Delta F/T}$, where ΔF is the difference of cost between the candidate and the current solution, and T is the *temperature*.

The temperature is decreased after a fixed number of samples N_s is drawn according to the *geometric cooling* scheme (line 20), where α (with $0 < \alpha < 1$) is the *cooling rate*.

In order to speed up the early stages of SA, we adopt the so-called *cut-off* mechanism, which decreases the temperature when a fixed number of moves N_a has been accepted.

Instead of using as parameter N_a, we define the parameter $\rho = N_a/N_s$ (with $0 < \rho \leq 1$) which represents the fraction of the number of iterations N_s, so that after accepting $\rho \cdot N_s$ we apply the cut-off, decreasing the temperature even if the number of sampled moves is less than N_s.

In order to have a fixed running time, instead of the stop criterion of line 4, we stop the SA procedure after the execution of a fixed number of iterations $\mathcal{I}$. In order to have the number $\mathcal{I}$ fixed, we compute the parameter N_s starting from the others using the following formula:

$$n_s = \mathcal{I} \Big/ \left(\frac{\log\left(T_f/T_0\right)}{\log \alpha} \right) \tag{1}$$

where T_0 and T_f are the initial and final temperature, respectively.

4.5. Adaptation to MI-FAP-I and MI-FAP-II

We discuss now how we adapt the local search components introduced above to the specific features of the two versions of the problem. For MI-FAP-I, the presence of many hard constraints due to co-cell, co-site, and handover separations make the problem of finding a feasible solution non-trivial.

Therefore, in order to deal with the feasibility problem properly, but also not to waste time in checking moves that create violations, we make use of a two-stage approach. The first stage starts from a random solution, whereas the second stage starts from the best solution of the first one. Each stage stops according to its own setting of the number of iterations.

In the *Feasibility Stage*, we use only the Change neighborhood and we include also separation violations in the cost function, with a suitably high weight. That is, the cost function that guides the search is the sum of soft constraints (e.g. interference costs) and hard constraints violations. For MI-FAP-I the interference costs occur for same-channel interferences or adjacent channel interferences. On the contrary, hard constraints violations are related to mandatory separations: co-cell frequency separation, co-site frequency separation, frequency separation for cells suffering the handover effect, and ad-hoc separations.

This first stage is rather short in relation to the second one, the *Optimization Stage*, but long enough to obtain a feasible solution. The purpose of the Feasibility Stage is not only to reach feasibility, but also to execute quickly the initial steep descent phase, by using only the Change neighborhood, which is computationally the cheapest.

In the *Optimization Stage*, we use the neighborhood portfolio, and we add the precondition that moves do not introduce violations of hard constraints. As a consequence, for this stage the cost function coincide with the objective function that is to minimize the interference costs.

It turned out that for FSwap and PFSwap it is very rare to find feasible moves, so that most of the time is spent in generating and rejecting moves. For this reason, for MI-FAP-I these two neighborhoods are excluded.

For MI-FAP-II the search method can be simplified in a few ways. First, there are no hard constraints in the strict sense, but a few occurrences of a very high cost. These few high values are considered like all the others, so that the search method proceeds in one single stage. Thus the cost function sums up the cost of transmitter separation violations. Furthermore, there are no forbidden frequencies, so that we don't have to check that a frequency is not available for the node.

The other difference between the two versions is in the computation of the costs: for MI-FAP-II each separation violation must be multiplied by its weight, whereas for MI-FAP-I interference depends on the distance (same or adjacent channel).

A peculiarity of Frequency Assignment problems, that is shared by MI-FAP-I and MI-FAP-II, is that certain nodes are *indistinguishable*, in the sense that they share exactly the same separations and interference levels. For MI-FAP-I this is made explicit by the fact that they are members of the same cell and of the same type (control or traffic), for MI-FAP-II indistinguishable pairs are detected by preprocessing the input data. To improve efficiency, we remove moves that involve two indistinguishable nodes, as they would not change the structure and the cost of the solution.

5. Experimental results

The software was implemented in C++ and compiled using g++ (v. 9.3). The experiments were run on AMD Ryzen Threadripper PRO 3975WX 32-Cores (3.50 GHz) with Ubuntu Linux 20.4. One single core was dedicated to each experiment.

5.1. Benchmarks

Both formulations are equipped with a specific dataset that has been used in previous works. We refer to Eisenblätter and Koster (2000) and Montemanni (2001), respectively, for an accurate description of the origin of these instances.

In order to save time, we decided to identify and remove the "easy" instances. We classify as easy those instances in which the same scores are obtained consistently by all configurations of our technique and by the most effective previous works.

The features of the non-easy instances are summarized in Tables 1 and 2, for MI-FAP-I and MI-FAP-II respectively. For MI-FAP-I we consider the number of sites (S), the number of cells (C), the number of nodes/transmitters (N), the number of frequencies (F), and the average

Table 1
Features of the instances for MI-FAP-I.

Instance	S	C	N	F	AF	SD	SV	ID
bradford_nt-1-eplus	649	1886	1971	75	75.0	0.0041	2.01	0.1305
bradford_nt-10-eplus	649	1886	4145	75	75.0	0.0045	1.90	0.1305
bradford_nt-10-free	649	1886	4145	75	75.0	0.0044	1.90	0.0525
bradford_nt-10-race	649	1886	4145	75	75.0	0.0044	1.90	0.0406
bradford_nt-2-eplus	649	1886	2214	75	75.0	0.0042	2.02	0.1310
bradford_nt-4-eplus	649	1886	2775	75	75.0	0.0043	1.99	0.1304
bradford-0-eplus	649	1886	1886	75	75.0	0.0041	2.00	0.1319
bradford-1-eplus	645	1878	2947	75	75.0	0.0050	2.13	0.1286
bradford-1-free	645	1878	2947	75	75.0	0.0049	2.13	0.0519
bradford-1-race	645	1878	2947	75	75.0	0.0049	2.13	0.0392
bradford-10-eplus	644	1876	4871	75	75.0	0.0052	2.02	0.1314
bradford-10-free	644	1876	4871	75	75.0	0.0052	2.02	0.0532
bradford-10-race	644	1876	4871	75	75.0	0.0052	2.03	0.0395
bradford-2-eplus	644	1876	3406	75	75.0	0.0051	2.12	0.1295
bradford-2-free	644	1876	3406	75	75.0	0.0051	2.12	0.0524
bradford-2-race	644	1876	3406	75	75.0	0.0051	2.12	0.0389
bradford-4-eplus	649	1886	3996	75	75.0	0.0051	2.08	0.1292
bradford-4-free	649	1886	3996	75	75.0	0.0051	2.09	0.0526
bradford-4-race	649	1886	3996	75	75.0	0.0051	2.09	0.0385
K	92	264	267	50	50.0	0.0297	2.00	0.5382
siemens1	179	506	930	75	43.0	0.0140	2.07	0.0764
siemens2	86	254	977	83	76.0	0.0373	2.08	0.4550
siemens3	366	894	1623	55	51.2	0.0175	2.03	0.0743
siemens4	276	760	2785	39	39.0	0.0072	2.14	0.0978
swisscom	87	148	310	68	29.0	0.0832	1.53	0.0433

Table 2
Features of the instances for MI-FAP-II.

Instance	N	Fs	ID	IS	P	H
AC-95-17	95	15	0.51	1.15	1.00	0
GSM-93	93	9/13	0.25	1.28	1.00	0
GSM-246	246	21/31	0.25	1.32	1.00	0
Test95	95	36	0.27	2.37	1.00	0
Test282	282	61/71/81	0.26	2.38	1.00	0
P06-3	153	31	0.79	1.59	1.00	0
P06-5	88	11	0.79	1.58	1.00	0
P06b-3	153	31	0.79	1.39	1.00	0
GSM2-184	184	39	0.40	1.20	1670.23	609
GSM2-227	227	29/39/49	0.39	1.18	1746.91	918
GSM2-272	272	34/39/49	0.39	1.16	1721.07	1155
1-1-50-75-30-2-50	75	5/10/11/12	0.30	1.26	10.81	0
1-2-50-75-30-4-50	75	9/11	0.30	1.62	11.09	0
1-3-50-75-30-0-50	75	7	0.30	1.00	10.97	0
1-4-50-75-30-2-1	75	6/10	0.30	1.25	1.00	0
1-5-50-75-30-2-100	75	10/12	0.30	1.26	21.35	0
1-6-50-75-30-0-1000	75	10/13	0.30	1.00	2068.48	0

number of available frequencies per transmitter (AF). We consider the split graph corresponding to the separations and we report its density (SD) and the average separation value (SV). Finally, we report the density of the interference split graph (ID).

For MI-FAP-II we consider the number of nodes/transmitters (N), the list of numbers of frequencies (Fs), the density of the interference graph (ID), the average separation (IS), and the average cost (P). For this version, the same instance is used with different number of frequencies, therefore we report here the list of them (Fs) rather than a single value (F). From the average costs we exclude the artificial high value used to state the hard separations. The number of hard separations is reported in the last column (H).

Notice that MI-FAP-I instances are generally much larger than those of MI-FAP-II in terms of number of nodes. For MI-FAP-I, we also notice that only four instances, namely siemens1, siemens2, siemens3 and swisscom, have some forbidden frequencies, shown by the fact that the value of the column AF is smaller than the value in the column F. In particular, siemens1 and siemens2 have only globally forbidden frequencies, whereas siemens3 and swisscom have also locally forbidden ones. For swisscom the AF value is particularly low, and in fact only for this instance it is particularly difficult to find a feasible solution.

Regarding MI-FAP-II, we notice that, based on the average separation cost (P), the dataset can be split on three distinct groups of instances. In detail, there is a group that has all costs equal to one, a second group with medium values (on the order of tens), and a final one with larger costs (on the order of thousands). This partition reflects, with a few exceptions, the different generation procedures: the first set is obtained from existing minimum span problems by limiting the number of available frequencies; the second set is composed of random scenarios generated using a basic graph generator, and the last set is obtained by adapting some fixed spectrum GSM problems to MI-FAP-II. As the cost values have a significant impact on the SA behavior, as described in the next section, we consequently perform separate tuning for these three groups.

Other datasets for MI-FAP have been proposed in the literature, like the Philadelphia and CALMA ones (see Aardal et al., 2002; Anderson, 1973). These cases however are rather simple to solve (most of the instances have been solved to optimality), so that we decided to skip them. On the contrary, the two instances Denver and Seattle proposed by Luna et al. (2007) are supposed to be difficult, but unfortunately they refer to a different version of the problem.

Table 3
Parameter tuning for MI-FAP-I.

Name	Description	Tuning Phase	Initial Range	Value
Feasibility Stage				
T_0	Start temperature	1	[500000, 2500000]	1314815
T_f	Final temperature	1	[1000, 10000]	9280
α	Cooling rate	1	[0.98, 0.999]	0.995
ρ	Accepted moves ratio	1	[0.03, 0.15]	0.076
Optimization Stage				
T_0	Start temperature	2	[300000, 1000000]	697531
T_f	Final temperature	2	[1000, 10000]	8632
α	Cooling rate	2	[0.98, 0.999]	0.985
ρ	Accepted moves ratio	2	[0.05, 0.2]	0.112
b_s	Bias toward swap moves	3	[0.0, 1.0]	0.906
b_g	Bias toward generalized swap moves	3	[0.0, 1.0]	0.906
p_{K}	Probability of Kick moves	4	[0.0, 0.4]	0.216
p_{G}	Probability of GKick moves	4	[0.0, 0.4]	0.042
p_{T}	Probability of 3-Swap moves	4	[0.0, 0.2]	0.009

MI-FAP-I instances have been translated from their original file format to a novel JSON-based one. The original format is rather complex to parse, so that we believe that this new one could foster the dissemination of these instances, which in fact has been quite limited in the recent times. On the contrary, for MI-FAP-II, the original format is extremely simple, so that we kept it as is.

5.2. Parameter tuning

The tuning procedure was performed using the tool JSON2RUN (Urli, 2013), which uses configurations generated according to Hammersley and Handscomb (1964), known as the *Hammersley point set*. JSON2RUN uses the F-Race procedure (Birattari et al., 2010) for selecting the best configuration, which is based on the Friedman and Wilcoxon statistical tests for removing inferior configurations as soon as possible.

The total number of parameters is quite large, hence the parameter tuning proceeds in phases, assuming that the interaction of the parameters involved in the different phases is minimal and can be neglected. In each phase, the parameters belonging to a subsequent phase are set to values given from preliminary experiments.

The winning configuration for MI-FAP-I is shown in Table 3, obtained with $\mathcal{I} = 3 \cdot 10^8$ corresponding to a running time of approximately 1500 seconds per run.

Notice the high values of the temperatures, which are due to the fact that the interference values are integers, obtained by multiplying the actual values by 10^8 (for full precision). Notice also the high values of the two bias parameters b_s and b_g, which show that the most useful kick moves are indeed swap ones.

The tuning procedure for MI-FAP-II works along the same tracks, except that there is no Feasibility Stage and the tuning is done separately for the three groups of instances, due to the different cost values, which influence the corresponding temperature ranges. Furthermore, there is an extra parameter K_m which is the maximum length of a PFSwap move, which is not in Table 3 as PFSwap is not used for MI-FAP-I. The best value found for K_m is 4.

5.3. Comparison results for MI-FAP-I

In our experience, for this problem the results improve consistently with the running time, without any sort of "plateau effect". As a consequence, the comparison should take into account the running times and also the CPU speed. Unfortunately though, a comparison of different CPUs in different years is rather impractical. In addition, Mannino et al. (2007), which hold most of the best results so far, granted an extremely long running time to their experiments (i.e., up to 128 hours per run, depending on the instance).

Therefore, as a fair comparison is not possible and the running times of Mannino et al. (2007) are impractical also for future comparisons, we decided for this version of the problem to grant our experiments a fixed number of iterations, specifically equal to $\mathcal{I} = 3 \cdot 10^9$. The results for 10 runs in comparison with the best in the literature are shown in Table 4. Beckmann and Killat (1999) and Hellebrandt and Heller (2000) do not report the running times. Montemanni et al. (2003) write that the experiments run for "several days". Montemanni and Smith (2010) set a timeout of 2h for all instances.

We can see that we outperformed all previous results on most of the instances, considering both the best and the average values. Only in six cases our average results are slightly inferior to the best ones, which however are obtained with much longer running time (on an older CPU, tough). In addition, we improve the best known solutions for 23 out of 25 instances.

5.4. Comparison results for MI-FAP-II

Table 5 shows the results for 30 runs with timeout 2400s, which has been set also by the other authors. For the path relinking approach by Lai and Hao (2015), we report the results obtained by both the random path relinking operator (denoted with rPR) and the randomized and mixed relinking operator (denoted with mrPR).

We can see that for all instances excluding instance GSM2-227 we reach the best known result, whereas the average results are in some cases worse than the previous ones, mainly by Lai and Hao. For the GSM2 instances, we have the best average results for 4 out of 7 instances.

In Table 6, we compare our results with those obtained by the hybrid genetic algorithm (HGA) of Siddiqi and Sait (2018) with a common time-limit of maximum 2 hours. For SA, the table reports the best and average values of 10 runs. It can be noticed that for all instances with a separation cost equal to one or to medium values, the performances of the two methods are almost equivalent with 14 ties, four instances for which the results of SA are better than those of HGA and five for which SA is worse. Conversely, for the last family, which comprises the GSM2* instances and the instance 1-6-50-75-30-0-1000 and that is characterized by large separation costs, HGA exhibits superior results, being better, equal and worse than SA in five, two and two cases, respectively.

Finally, in Table 7 we present comparative results on a subset of the MI-FAP-II dataset composed by the most challenging instances for a time-limit of 48 hours. The first column reports some lower bounds (LB) computed by Montemanni et al. (2004); for the evolutionary algorithm (EA) developed by Segura et al. (2016), the table shows average and best values of 30 runs, for SA those of 5 runs. It can be noticed that SA outperforms EA in nine out of 13 instances, founding four new best

Table 4
Computational results for MI-FAP-I on COST 259 instances.

	Beckmann	Hellebrandt	Montemanni	Montemanni		Mannino		SA		
	1999	2000	2003	2010		2007				
instance	best	best	best	best	avg	best	t[h]	best	avg	t[h]
bradford_nt-1-eplus	1.04	**0.86**				**0.86**	22	0.871	0.951	2.1
bradford_nt-10-eplus	148.12	146.12				144.94	19	**142.746**	143.719	3.7
bradford_nt-10-free	8.63	5.863				5.42	14	**4.945**	5.213	3.3
bradford_nt-10-race	1.73	1.074				1.09	14	**1.035**	1.077	4.2
bradford_nt-2-eplus	3.79	3.168				3.20	24	**3.152**	3.372	2.3
bradford_nt-4-eplus	19	17.728				17.72	21	**17.209**	17.682	2.7
bradford-0-eplus	0.8					0.60	64	**0.597**	0.641	2.0
bradford-1-eplus	33.99					33.80	64	**32.381**	32.735	3.2
bradford-1-free	0.16					**0.12**	30	0.164	0.172	3.2
bradford-1-race	0.03					0.01	26	**0.009**	0.011	4.0
bradford-10-eplus	400					395.50	128	**387.206**	388.573	5.9
bradford-10-free	117.8					113.70	63	**104.612**	105.997	4.7
bradford-10-race	30.22					27.38	46	**23.758**	24.072	5.3
bradford-2-eplus	80.03					79.38	75	**76.674**	76.984	3.9
bradford-2-free	2.95					2.69	37	**2.275**	2.365	3.6
bradford-2-race	0.42					0.32	39	**0.201**	0.222	4.3
bradford-4-eplus	167.7					167.00	90	**161.325**	162.894	4.5
bradford-4-free	22.09					20.00	46	**17.883**	18.348	3.9
bradford-4-race	3.04					2.93	36	**2.174**	2.277	4.7
K		0.45	0.447	0.4647	0.4886			**0.415**	0.434	0.4
siemens1	2.78	2.301		2.7642	2.8492	2.20	5	**1.970**	2.035	0.9
siemens2	15.46	14.751	14.275	14.936	15.0578	14.27	9	**14.005**	14.156	2.7
siemens3	6.75	5.259	5.186	6.6496	6.7358	5.13	15	**4.852**	4.971	3.1
siemens4	89.15	80.967	81.876	110.9725	112.482	77.25	18	**76.298**	77.014	5.6
swisscom	27.36		27.211					**27.027**	29.444	1.3

Table 5
Computational results for MI-FAP-II with a time-limit of 2400 secs.

		Montemanni	Montemanni		Lai				Lahsinat		SA	
		2003	2010		2015				2018			
					rPR		mrPR					
Instance	F	best	best	avg	best	avg	best	avg	best	avg	best	avg
AC-95-17	15	33	33	**33.0**	33	**33.0**	33	**33.0**			33	33.1
GSM-93	9	32	32	**32.2**	32	**32.2**	32	**32.2**	33	34.18	32	33.2
GSM-93	13	7	7	**7.0**	7	**7.0**	7	**7.0**	8	8.60	7	**7.0**
GSM-246	21	79	79	80.2	79	80.6	78	79.0	83	84.78	77	**78.9**
GSM-246	31	25	25	26.1	26	26.1	24	25.1			24	**24.7**
Test95	36	12	8	**8.0**	8	**8.0**	8	**8.0**	8	**8.00**	8	**8.0**
Test282	61	51	51	**53.2**	56	56.8	56	57.1			51	54.3
Test282	71	27	27	29.3	29	30.5	29	30.6			27	**28.9**
Test282	81		10	11.9	9	10.9	10	11.5			9	**10.5**
P06-5	11	133	133	**133.0**	133	**133.0**	133	**133.0**	137	137.26	133	**133.0**
P06-3	31	115	115	**115.0**	115	**115.0**	115	**115.0**	115	115.10	115	**115.0**
P06b-3	31	112	112	**112.0**	112	**112.0**	112	**112.0**	112	117.00	112	**112.0**
GSM2-184	39	5521	5447	5598.8	5258	**5270.8**	5250	5276.9	5898	6180.71	5250	5279.9
GSM2-227	29		61586	66510.0	57790	59555.4	58834	59907.7	67586	68721.00	56122	**57932.6**
GSM2-227	39	10979	10550	10897.7	8656	**9022.4**	8760	9329.7			8702	9087.4
GSM2-227	49	2459	2459	2613.1	1998	**1998.0**	1998	2009.4			1998	2019.0
GSM2-272	34		56128	58691.4	53254	55954.2	54085	56916.3	65150	67888.30	51579	**53118.9**
GSM2-272	39	27416	27416	28488.2	27503	28299.7	28074	28880.4			26479	**27165.1**
GSM2-272	49	7785	7785	7946.7	7185	7265.2	7107	7252.5			7075	**7169.4**
1-1-50-75-30-2-50	5		1242	1253.9	1242	**1242.0**	1242	**1242.0**	1257	1268.44	1242	**1242.0**
1-1-50-75-30-2-50	10		97	103.8	96	**96.0**	96	**96.0**			96	96.2
1-1-50-75-30-2-50	11		59	66.1	55	**55.0**	55	**55.0**			55	56.2
1-1-50-75-30-2-50	12		36	38.7	32	**32.0**	32	**32.0**			32	32.6
1-2-50-75-30-4-50	9		671	680.6	665	**665.0**	665	**665.0**	670	674.18	665	**665.0**
1-2-50-75-30-4-50	11		317	325.0	313	**313.0**	313	**313.0**			313	313.6
1-3-50-75-30-0-50	7		194	196.5	194	**194.0**	194	**194.0**	196	196.76	194	**194.0**
1-4-50-75-30-2-1	6		70	70.9	70	**70.0**	70	**70.0**	71	74.20	70	**70.0**
1-4-50-75-30-2-1	10		19	**19.0**	19	**19.0**	19	**19.0**			19	**19.0**
1-5-50-75-30-2-100	10		176	183.8	168	**168.0**	168	**168.0**			168	173.6
1-5-50-75-30-2-100	12		63	69.3	57	**57.0**	57	**57.0**			57	57.7
1-6-50-75-30-0-1000	10		6840	7064.3	6777	**6777.0**	6777	**6777.0**			6777	**6777.0**
1-6-50-75-30-0-1000	13		1207	1365.2	1190	**1190.0**	1190	**1190.0**			1190	**1190.0**

Table 6
Computational results for MI-FAP-II with a time-limit of 2 hours.

		HGA		SA	
Instance	F	best	avg	best	avg
AC-95-17	15	33	**33.0**	33	**33.0**
GSM-93	9	32	**32.0**	32	33.2
GSM-93	13	7	**7.0**	7	**7.0**
GSM-246	21	78	79.2	78	**78.6**
GSM-246	31	24	24.8	24	**24.3**
Test95	36	8	**8.0**	8	**8.0**
Test282	61	52	53.8	51	**53.5**
Test282	71	27	**27.4**	26	27.5
Test282	81	8	**8.3**	8	9.5
P06-5	11	133	**133.0**	133	**133.0**
P06-3	31	115	**115.0**	115	**115.0**
P06b-3	31	112	**112.0**	112	**112.0**
GSM2-184	39	5250	5265.0	5258	**5264.4**
GSM2-227	29	55513	**56789.0**	56464	57474.7
GSM2-227	39	8520	**8700.3**	8762	8911.3
GSM2-227	49	1998	**1998.0**	1998	2002.0
GSM2-272	34	51493	**52354.5**	51877	52907.1
GSM2-272	39	25932	**26685.0**	26198	26766.4
GSM2-272	49	7056	7129.4	7017	**7089.0**
1-1-50-75-30-2-50	5	1242	**1242.0**	1242	**1242.0**
1-1-50-75-30-2-50	10	96	**96.0**	96	**96.0**
1-1-50-75-30-2-50	11	55	**55.0**	55	**55.0**
1-1-50-75-30-2-50	12	32	**32.8**	32	**32.0**
1-2-50-75-30-4-50	9	665	**665.0**	665	**665.0**
1-2-50-75-30-4-50	11	313	**313.0**	313	**313.0**
1-3-50-75-30-0-50	7	194	**194.0**	194	**194.0**
1-4-50-75-30-2-1	6	70	**70.0**	70	**70.0**
1-4-50-75-30-2-1	10	19	**19.0**	19	**19.0**
1-5-50-75-30-2-100	10	168	**168.0**	168	169.2
1-5-50-75-30-2-100	12	53	**56.5**	57	57.0
1-6-50-75-30-0-1000	10	6777	**6777.0**	6777	**6777.0**
1-6-50-75-30-0-1000	13	1190	**1190.0**	1190	**1190.0**

known solutions. These last cases are marked with an $*$ in the column corresponding to the best values obtained by SA. We want to remark that the EA was specifically designed to deal with long-term executions, and it has not been tested on shorter running times. Indeed, the authors themselves claim that with short time "high-quality results could not be obtained".

6. Discussion

Our solver works reasonably well for both formulations, though the results are somewhat better for MI-FAP-I than for MI-FAP-II. This shows that it is particularly competitive for large instances and the more complex structure.

We see that the approach of Montemanni and Smith (2010), which has good results for MI-FAP-II, works less effectively for the few instances of MI-FAP-I upon which it has been tested. The other approaches that performed well on MI-FAP-II have not been applied to MI-FAP-I, therefore we cannot make any conclusions on their potential performance on this version.

In addition, the SA method obtains very competitive results for both short and long executions, proving that it is flexible to different timeout. In particular, with the long runs it improved many best known results.

6.1. Larger neighborhoods

It would be worth discussing whether larger neighborhoods could contribute to improve the results. For example, we could consider the X-Swap neighborhood (with X > 3), i.e., the generalization of 3-Swap. However, the tuning experiments showed that the contribution of the 3-Swap neighborhood in the overall best configuration is rather limited ($p_T = 0.009$). They also showed (not reported in the paper) that the configurations with even lower values of p_T (even $p_T = 0.0$) do not have a significant loss of performance. In addition, X-Swap would result in a more complex neighborhood structure with a less efficient evaluation of the delta costs. For these reasons, we decided not to investigate further in the X-Swap direction.

6.2. Instance-based tuning

Some additional insights about the results come from the analysis of the ratio between the time to find the best solution and the total elapsed time. In most instances this ratio is close to 1, showing that the best solution is found toward the end of the search. This is a positive behavior that shows that no time is wasted during the search.

There are however a few instances in which this ratio is constantly much lower than 1. In particular, for one specific instance, namely swisscom, this ratio turned out to be extremely low (around 0.01). This is a very peculiar and constrained instance, in which finding a feasible solution is much more difficult than in all the others. This behavior rises the question whether the parameter values coming for the general tuning are suitable for this "outlier".

Additional experiments on this instance alone proved that an ad-hoc tuning yields to different values (in particular a much higher value for T_0), which would result in much better scores. Specifically, we obtain an average cost of 25.99 and a best one of 23.478, compared to 29.444 and 27.027 of Table 4, respectively.

This is however the result of an "overtuning", which is methodologically unacceptable, as the tuning procedure is expected to prepare the method for generic unforeseen instances. Otherwise, the tuning procedure should be considered as part of the solution of the instance and its time should be included in the solution time.

Nonetheless, this situation might pave the way for a feature-based tuning that relates the parameters of the search method to the features of the instance. This however would require a much larger dataset of instances, and thus will be subject of future work.

Table 7
Computational results for MI-FAP-II with a time-limit of 48 hours.

		LB	EA		SA	
Instance	F	value	best	avg	best	avg
GSM-246	21	50	77	79.2	77	**77.8**
GSM-246	31	16	25	26.2	*23	**23.6**
Test282	61	21	53	54.7	*50	**50.6**
Test282	71	6	27	28.7	*25	**25.4**
Test282	81		8	9.9	*7	**7.8**
GSM2-184	39	4856	5250	5251.6	5250	**5251.2**
GSM2-227	29		55,339	**56349.0**	55,796	56447.4
GSM2-227	39	7445	8283	**8567.0**	8467	8580.4
GSM2-227	49	1998	1998	**1998.0**	1998	**1998.0**
GSM2-272	34		50,940	51757.0	50,959	**51565.0**
GSM2-272	39		25,542	26099.6	25,780	**25923.4**
		16,144				
GSM2-272	49	6310	6957	7096.6	6978	**7012.6**
1-5-50-75-30-2-100	10	94	168	**168.0**	168	**168.0**

7. Conclusions and future work

We have proposed a multi-neighborhood Simulated Annealing approach for the MI-FAP problem. The solver has been designed to deal with two versions of the problem (with some adaptations), that we called MI-FAP-I and MI-FAP-II. Our solver proved to be effective and robust on both formulations, on many diverse instances, and with different time-limits, and compares favorably with previous results.

Most of the recent work focused on the MI-FAP-II version, probably due to the higher structural complexity of the MI-FAP-I formulation, and maybe also to the larger size of its instances. Another reason for the lack of recent "success" of the MI-FAP-I formulation might be its cumbersome file format. To this regard, we have translated all instances to a novel JSON format, with the expectation that this "restyling" might bring it back on tracks for future comparisons.

To this aim, the publication of instances and solutions, along with the online validator, might also attract new research on this interesting problem.

For the future we plan to apply and adapt our approach to different versions of the Frequency Assignment Problem, in particular to new formulations that will emerge from recent technologies, such as for example 5G wireless networks.

We also plan to design new neighborhoods to be added to our portfolio, in order to further improve the performances of our solver. In detail, we plan to introduce larger neighborhoods that better exploit the specificity of the problem. For example, we plan to consider neighborhoods based on the concept of *cell-reoptimization*, reassigning simultaneously all transmitters of a single cell.

As another possible research direction, we would like to consider the possibility to hybridize our solver with exact methods, developing some form of matheuristic. Finally, we would like to explore the possibility to employ some form of learning mechanism in order to adapt the rates of the neighborhoods, i.e. p_* parameters, during the search.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank Carlo Mannino and Roberto Montemanni for helpful discussions about their work.

References

Aardal, K., Hurkens, C., Lenstra, J., Tiourine, S., 1996. Algorithms for frequency assignment problems. CWI Quarterly 9, 1–8.
Aardal, K., Hurkens, C., Lenstra, J.K., Tiourine, S., 2002. Algorithms for radio link frequency assignment: the calma project. Oper Res 50, 968–980.
Aardal, K.I., Van Hoesel, S.P., Koster, A.M., Mannino, C., Sassano, A., 2007. Models and solution techniques for frequency assignment problems. Ann Oper Res 153, 79–129.
Allen, J.D., Helgason, R.V., Kennington, J.L., 1987. The frequency assignment problem: a solution via nonlinear programming. Nav. Res. Logist. 34, 133–139.
Anderson, L.G., 1973. A simulation study of some dynamic channel assignment algorithms in a high capacity mobile telecommunications system. IEEE Trans. Commun. 21, 1294–1301.
Beckmann, D., Killat, U., 1999. Frequency planning with respect to interference minimization in cellular radio networks. Technical Report,TD(99) 032, Vienna, Austria COST 259
Bellio, R., Ceschia, S., Di Gaspero, L., Schaerf, A., 2021. Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling. Computers and Operations Research 132, 105300.
Birattari, M., Yuan, Z., Balaprakash, P., Stützle, T., 2010. F-Race and Iterated F-Race: An Overview. In: Experimental methods for the analysis of optimization algorithms. Springer, Berlin, pp. 311–336.
Björklund, P., Värbrand, P., Yuan, D., 2005. Optimized planning of frequency hopping in cellular networks. Computers & Operations Research 32, 169–186.
Borndörfer, R., Eisenblätter, A., Grötschel, M., Martin, A., 1998. Frequency assignment in cellular phone networks. Ann Oper Res 76, 73–93.
Chiarandini, M., Stützle, T., 2007. Stochastic local search algorithms for graph set T–colouring and frequency assignment. Constraints 12, 371–403.
Correia, L. M. (Ed.), 2001. Wireless Flexible Personalized Communications - COST 259: European Co-operation in Mobile Radio Research, John Wiley & Sons. COST Action 259—Final Report.
Crisan, C., Mühlenbein, H., 1998. The frequency assignment problem: a look at the performance of evolutionary search. Lect. Notes Comput. Sci. 1363, 263–274.
Duque-Antón, M., Kunz, D., Rüber, B., 1993. Channel assignment for cellular radio using simulated annealing. IEEE Trans. Veh. Technol. 42, 14–21.
Eisenblätter, A., Koster, A., 2000. Fap web - a website about frequency assignment problems. fap.zib.de, Last modified Jan 2010
Franzin, A., Stützle, T., 2019. Revisiting simulated annealing: a component-based analysis. Computers and Operations Research 104, 191–206.
Galinier, P., Hertz, A., 2006. A survey of local search methods for graph coloring. Computers & Operations Research 33, 2547–2562.
González-Velarde, J.L., Laguna, M., 2002. Tabu search with simple ejection chains for coloring graphs. Ann Oper Res 117, 165–174.
Hammersley, J.M., Handscomb, D.C., 1964. Monte carlo methods. Chapman and Hall, London.
Hellebrandt, M., Heller, H., 2000. A new heuristic method for frequency assignment. Technical Report, TD(00) 003, Valencia, Spain, COST 259
Kapsalis, A., Chardaire, P., Rayward-Smith, V.J., Smith, G.D., 1995. The radio link frequency assignment problem: a case study using genetic algorithms. Lecture Notes on Computer Science 993, 117–131.
Kiouche, A.E., Bessedik, M., Benbouzid-SiTayeb, F., Keddar, M.R., 2020. An efficient hybrid multi-objective memetic algorithm for the frequency assignment problem. Eng Appl Artif Intell 87, 103265.
Kirkpatrick, S., Gelatt, D., Vecchi, M., 1983. Optimization by simulated annealing. Science 220, 671–680.
Kolen, A., 2007. A genetic algorithm for the partial binary constraint satisfaction problem: an application to a frequency assignment problem. Stat Neerl 61, 4–15.
Koster, A.M.C.A., van Hoesel, C.P.M., Kolen, A.W.J., 1999. Optimal solutions for a frequency assignment problem via tree-decomposition. Lect. Notes Comput. Sci. 1665, 338–349.
Koster, A.M.C.A., van Hoesel, S.P.M., Kolen, A.W.J., 2002. Solving partial constraint satisfaction problems with tree decomposition. Networks 40, 170–180.
Lahsinat, Y., Boughaci, D., Benhamou, B., 2018. Breakout variable neighbourhood search for the minimum interference frequency assignment problem. Journal of Systems and Information Technology 20, 468–488.
Lai, X., Hao, J.K., 2015. Path relinking for the fixed spectrum frequency assignment problem. Expert Syst Appl 42, 4755–4767.
Laidoui, F., Bessedik, M., Si-Tayeb, F.B., Bengherbia, N., Khelil, M.Y., 2018. Nash-pareto genetic algorithm for the frequency assignment problem. Procedia Comput Sci 126, 282–291. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 22nd International Conference, KES-2018, Belgrade, Serbia
Lal, B., Balakrishnan, A., Caldwell, B. M., Buenconsejo, R. S., Carioscia, S. A., 2018. Global trends in space situational awareness (SSA) and space traffic management (STM). Technical Report, Institute for Defense Analyses Washington DC.
Lin, K., Wang, W., Wang, X., Ji, W., Wan, J., 2015. Qoe-driven spectrum assignment for 5g wireless networks using sdr. IEEE Wireless Commun. 22, 48–55.
Luna, F., Blum, C., Alba, E., Nebro, A.J., 2007. Aco vs Eas for Solving a Real-world Frequency Assignment Problem in Gsm Networks. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, NY, USA, pp. 94–101. In GECCO '07
Luna, F., Estébanez, C., León, C., Chaves-Gonzlez, J.M., Nebro, A.J., Aler, R., Segura, C., Vega-Rodríguez, M.A., Alba, E., Valls, J.M., Miranda, G., Gómez-Pulido, J.A., 2011. Optimization algorithms for large-scale real-world instances of the frequency assignment problem. Soft comput 15, 975–990.
Mannino, C., Oriolo, G., Ricci, F., Chandran, S., 2007. The stable set problem and the thinness of a graph. Operations Research Letters 35, 1–9.
Montemanni, R., 2001. Upper and lower bounds for the fixed spectrum frequency assignment problem. Ph.D. thesis University of Glamorgan.
Montemanni, R., Moon, J.N., Smith, D.H., 2003. An improved tabu search algorithm for the fixed-spectrum frequency-assignment problem. IEEE Trans. Veh. Technol. 52, 891–901.
Montemanni, R., Smith, D., Allen, S., 2004. An improved algorithm to determine lower bounds for the fixed spectrum frequency assignment problem. Eur J Oper Res 156, 736–751.
Montemanni, R., Smith, D., Allen, S.M., 2001. Lower bounds for fixed spectrum frequency assignment. Ann Oper Res 107, 237–250.
Montemanni, R., Smith, D.H., 2010. Heuristic manipulation, tabu search and frequency assignment. Computers & operations research 37, 543–551.
Segura, C., Hernández-Aguirre, A., Luna, F., Alba, E., 2016. Improving diversity in evolutionary algorithms: new best solutions for frequency assignment. IEEE Trans. Evol. Comput. 21, 539–553.
Siddiqi, U.F., Sait, S.M., 2018. An optimization heuristic based on non-dominated sorting and tabu search for the fixed spectrum frequency assignment problem. IEEE Access 6, 72635–72648.
Tiourine, S.R., Hurkens, C.A.J., Lenstra, J.K., 2000. Local search algorithms for the radio link frequency assignment problem. Telecommun Syst 13, 293–314.
Urli, T., 2013. json2run: a tool for experiment design & analysis. CoRR abs/1305.1112.
Wang, P., Henz, B., 2017. Frequency assignment for joint aerial layer network high-capacity backbone. Technical Report, Army Research Laboratory Aberdeen Proving Ground United States
Zhang, D., Piao, M., Zhang, T., Chen, C., Zhu, H., 2020. New algorithm of multi-strategy channel allocation for edge computing. AEU-International Journal of Electronics and Communications 126, 153372.
Zhao, L., Wang, H., Zhong, X., 2018. Interference graph based channel assignment algorithm for d2d cellular networks. IEEE Access 6, 3270–3279.