An Exact Algorithm for the Min-Interference Frequency Assignment Problem

Vittorio Maniezzo[1], *Roberto Montemanni*

Department of Computer Science
University of Bologna
Via Sacchi 3, 47023 Cesena, Italy

[1] Contact author. Email: maniezzo@csr.unibo.it, fax +39 0547 610054, web: www.csr.unibo.it/~maniezzo

Abstract

In this paper we consider the Frequency Assignment Problem, where the objective is to minimize the cost due to interference arising in a solution. We use a quadratic 0-1 integer programming formulation of the problem as a basis to derive new lower bounds and problem reduction rules.

A tree search algorithm, that uses the lower bounds and dominance criteria is also presented. Computational results are shown on standard benchmark instances from the literature.

Keywords: Integer programming, branch-and-bound, telecommunications.

1. Introduction

In recent years we have witnessed a tremendous growth of mobile communication networks. Several combinatorial optimization problems arise in the context of the design and management of such networks; their actual interest has fostered research on mathematical properties and solution techniques. The problem we consider in this paper is the Frequency Assignment Problem (FAP), a network management problem; we face also a special case of FAP named Radio Link Frequency Assignment Problem (RLFAP). FAP arises when a network of radio links is to be established over a given territory, and a frequency has to be assigned to each link. This happens, for example, when cellular mobile phoning is to be introduced in an area, where the connection between the cellular phones and the transmission network is supported by radio links.

In this last case, a transmitter (i.e., a phone) establishes a radio link with a receiver (an antenna of a *base-station*), on one of the frequencies that the receiver supports. Interference may arise among different communications, but its level has to be acceptable, or communication will be distorted. Acceptability is usually specified by means of a threshold, called *separation*, on the distance between frequencies which can be operated concurrently by the same receiver or which can be used in areas where the transmitter could interact with more than one receiver.

Each receiver can operate on a given spectrum of frequencies, which is usually partitioned into *channels*. The problem arising is to define which among the available channels are to be used by each receiver for servicing the radio links, minimizing the resulting interference.

The FAP has been formulated as an optimization problem with different objective functions. The first objectives considered were the minimization of the *order* (number of different frequencies used in the whole system) or of the *span* (difference between the highest and the smallest frequencies used in the system), under constraints that ensured an acceptable interference level. Currently, parallel to the evolution of the market conditions, new models have been proposed, where all available frequencies can be used and low levels of interference are permitted.

The FAP reduces to a graph coloring problem in the case that all separations are 1 and the objective is to minimize the order, therefore FAP is NP-hard. Simple reductions prove that also the other versions of the problem are NP-hard.

Many heuristic algorithms have been presented, most of them for the problem of minimizing the span or the order (Gamst, Rave, 1982; de Werra, Gay, 1994; Costa, 1993; Smith, Hurley, Thiel, 1998; Hurkens, Tiourine, 1995, Warners, et al., 1997). More recently, heuristic methods have been applied to the problems of minimizing the global interference level of the system or of maximizing the number of satisfied links under maximum global interference constraints (Borndörfer, et al., 1998; Hurkens, Tiourine, 1995; Adjakplé, Jaumard, 1997; Warners, et al., 1997, Maniezzo, Carbonaro, Montemanni, 1999).

Several lower bounds have also been derived but, as for heuristic algorithms, the main results are for span or order minimization. Most of these bounds are obtained through a reformulation of the problem as a *T*-Coloring Problem (Gamst, 1986; Tcha, Chung, Choi, 1997; Smith, Hurley, 1997). For the problem of minimizing the global interference, only two results have been presented in the literature so far, both specific for particular situations.

The first of these bounds was presented by Hurkens and Tiourine (1995) and is based on the solution of a nonlinear problem which is by construction a lower bound to the original FAP optimum solution cost. This bound becomes tight when the instance to solve has high costs associated to particular frequency assignments for specific links (so called *mobility costs*, as detailed in the following).

The second lower bound was presented by Koster, van Hoesel and Kolen (1997) and is based on the linear relaxation of a generalization of the FAP to a problem which can be framed as a Partial Constraint Satisfaction Problem (PCSP). The PCSP of interest can be formulated as a {0,1}- programming problem, and it models a FAP when a suitable cost setting is used. Its linear relaxation yields a lower bound that can be strengthened by means of valid inequalities. Unfortunately, this bound could be tested only on instances with small frequency domains (less than three frequencies available for each link).

Exact algorithms have been presented mainly for the problems for which lower bounds are available. In particular, for the order or span minimization problem, Aardal et al. (1995) presented a branch-and-cut algorithm, while Mannino and Sassano (1999) propose a exact enumeration method with fixing criteria and relaxation/extension techniques. For the problem of maximizing the satisfied demand, Kazantzakis, Demestichas and Anagnostou (1995) proposed a branch and cut method, tested on a 5 x 5 square grid network with uniform cell demand. Other authors used the optimal value of the continuous relaxation of a suitable {0-1}-programming model as a lower bound that directs a binary separation scheme on the problem variables (Giortzis, Turner, 1996); for this same problem Fischetti et al. (1997) presented an approach based on an integer linear programming formulation, which is solved to optimality by a branch and cut algorithm.

For the interference minimization problem no exact method has been so far reported and tested. In the literature in fact, there are only indications on the use in branch and bound algorithms of the two lower bounds previously described for this type of FAP (Hurkens, Tiourine, 1995; Koster, van Hoesel, Kolen, 1997), but with no computational results presented.

This article describes an exact technique for the interference minimization problem. It consists of a branch and bound algorithm, based on new lower bounds derived from a quadratic formulation of the FAP and on dominance rules.

The paper is structured as follows. Section 2 introduces the problem more formally and presents the mathematical formulation we have used. Sections 3 and 4 introduce the lower bounds and the dominance/reduction rules that have been included in the branch and bound algorithm. Section 5 describes the algorithm itself and Section 6 shows the computational results that we obtained from its implementation.

2. The Frequency Assignment Problem (FAP)

The Frequency Assignment Problem considered in this paper can be represented as follows. Consider an index set of links $\mathfrak{L} = \{1, \dots, n\}$, a set $\mathfrak{F}_i = \{f_{i1},\dots, f_{iF_i}\}$, $i=1, \dots, n$ of available frequencies for each link and a Channel Separation Matrix (CSM) = [d(i,j)] $i,j = 1, \dots, n$, where d(i,j) defines the minimal distance between frequencies (channels) assigned to links of index i and j: if a CSM constraint (ij) is violated, a cost c(i,j) has to be paid. In the following, to indicate $|\mathfrak{F}_i|$ we will use the notations F_i and F(i) interchangeably.

It is possible to associate with each FAP instance a weighted interference graph G = (N, E, D), where N is the set of vertices, E is the set of edges and D is a weight vector. A vertex in N corresponds to a frequency request (link). There is an edge (ij) between vertices i and j if and only if there is a non zero entry in the CSM for the corresponding request. The edges are weighted by means of the corresponding values from the CSM.

A FAP instance can thus be described by the 7-tuple *FAPMI* = (N, E, D, C, $\mathfrak{F}$, P, M), where:

- N is the set of nodes of the graph G, corresponding to the links of the FAP instance.
- $E \subseteq (N \times N)$ is the set of (undirected) edges of the graph G. Each edge is associated with a pair of links which can potentially generate interference;
- D contains the weights associated with the edges of G, corresponding to CSM entries: d(i,j) is thus the minimal distance between frequencies assigned to links corresponding to nodes i and j that generates no interference;
- C is a set of weights associated to the arcs of G, corresponding to interference costs: c(i,j) represents the interference cost arising when the frequencies assigned to the links corresponding to nodes i and j are closer than d(i,j);
- $\mathfrak{F}$ is the collection of sets $\mathfrak{F}_i = \{f_{i1},\dots, f_{iF_i}\}$, $i = 1, \dots, n$, which represent the available frequencies for each link $i \in N$. Notice that $|\mathfrak{F}_i| = F_i$.
- $P \subset \{0\} \cup \left(\bigcup_{i \in N} \mathfrak{F}_i \right)$ is an ordered set of integers, where P(i) denotes a preassigned frequency for the link associated with node i. P(i)=0 means that no frequency was preassigned to node i.

- M is a set of weights associated with the nodes of G: m(*i*) is the *mobility* cost for assigning to node *i* a frequency different from P(*i*). If P(*i*)=0 then m(*i*)=0.

In Figure 1 we present a graphical example of the problem.

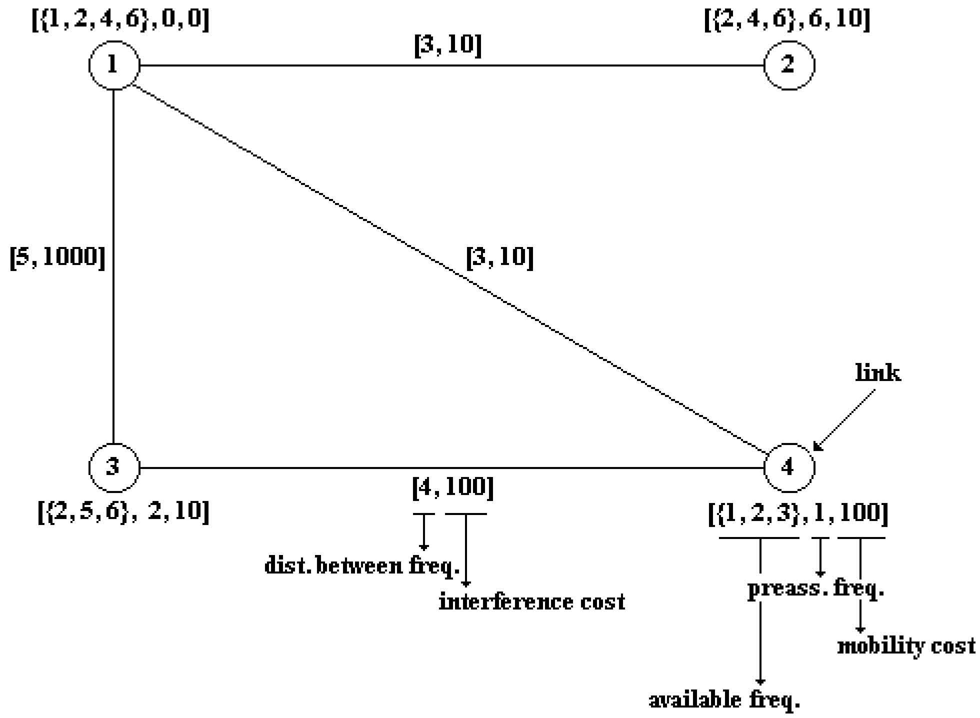

Figure 1. Example of a FAP instance.

It is interesting to notice that *FAPMI* is a generalization of the graph coloring problem. To represent a generic coloring decision problem we have to use as graph G the graph to color, and to weight with $d(i,j)=1$ and $c(i,j)=1$ each arc of *G*. Finally, fixing $m(i)=0$ and $\mathfrak{F}_i=\{1,..., k\}$, $i = 1,\dots, n$, where *k* is the number of usable colors, we get the coloring instance: a zero cost solution entails a positive answer to the coloring decision problem.
By means of model *FAPMI* it is also possible to model the decision span minimization FAP (*FAPMS*), provided that all links have the same set [1, *M*] of contiguous available frequencies, with M a positive integer. To represent the span minimization FAP it is possible to proceed as in the coloring problem case. We fix a span *k* ($k \le M$) and we define the frequency domains for the links to be $\mathfrak{F}_i = \{1,..., k+1\}$ for $i = 1, \dots, n$. The remaining elements of *FAPMI* are fixed as for the coloring. If a *FAPMI* solution exists with zero cost, than that solution corresponds to a feasible solution for *FAPMS* with span *k*, otherwise no solution exists for *FAPMS* with span *k*.

Different mathematical formulations have been proposed for the FAP. The one we used, originally presented by Koster, van Hoesel, Kolen (1997), can be derived by suitably defining the costs in the formulation of a problem more general than the FAP, that can be framed as a Partial Constraint Satisfaction Problem (PCSP). PCSPs extend the classical Constraint Satisfaction Problem framework (Tsang, 1993) to over-constrained instances, that is instances where no solution exists which satisfies all constraints. In such cases, the problem becomes that of finding a solution which minimizes some function of the cost incurred due to violated constraints, as is the case of the FAP of interest to us.

The notation we use is the following.

Let c_{ikjl} be the penalty, due to interference costs, paid when link i is assigned to its k-th frequency and link j to its l-th frequency and let m_{ik} be the penalty, due to mobility costs, paid when link i is assigned to its k-th frequency.

The model makes use of two sets of variables. Variable y_{ikjl} has value 1 when link i is assigned to its k-th and link j to its l-th frequency, 0 otherwise. Variable x_{ik} is 1 when link i is assigned to its k-th frequency, 0 otherwise.

Finally, let $\Gamma(i)$ denote the set of neighbors of the generic link i in the interference graph (i.e., $\Gamma(i) = \{j : (i,j)\in E\}$).

The formulation, following Koster, van Hoesel and Kolen (1997), is the following:

$$\text{(F1) } z_{F1} = \text{Min} \sum_{i=1}^{n} \sum_{j\in\Gamma(i);\, j>i} \sum_{k=1}^{F_i} \sum_{l=1}^{F_j} c_{ikjl}\, y_{ikjl} + \sum_{i=1}^{n} \sum_{k=1}^{F_i} m_{ik}\, x_{ik} \qquad (1)$$

$$\text{s.t.} \quad \sum_{l=1}^{F_j} y_{ikjl} - x_{ik} = 0 \qquad i=1,\dots,n;\ k=1,\dots,F_i;\ j\in\Gamma(i); i<j \qquad (2)$$

$$\sum_{k=1}^{F_i} y_{ikjl} - x_{jl} = 0 \qquad j=1,\dots,n;\ l=1,\dots,F_j;\ i\in\Gamma(j); i<j \qquad (3)$$

$$\sum_{k=1}^{F_i} x_{ik} = 1 \qquad i=1,\dots,n \qquad (4)$$

$$x_{ik} \in \{0,1\} \qquad i=1,\dots,n,\ k=1,\dots,F_i \qquad (5)$$

$$y_{ikjl} \in \{0,1\} \qquad i=1,\dots,n;\ j\in\Gamma(i);\ i<j;\ k=1,\dots,F_i;\ l=1,\dots,F_j \qquad (6)$$

Where constraints (2) and (3) correlate the *x* and *y* values, constraints (4) specify that exactly one frequency has to be assigned to each link and constraints (5) and (6) are integrality constraints.

As mentioned, formulation F1 represents a PCSP problem more general than the FAP. FAP instances are characterized by having $c_{ikjl} = 0$ or $c_{ikjl} = c(i,j)$, for each *i*, *j* and $k=1, \ldots, F_i$, $l=1, \ldots, F_j$, where, $c_{ikjl} = c(i,j)$ only for contiguous values of *l* (those for which $|f_{ik} - f_{jl}| < d(i,j)$). In the same way, m_{ik}, $i=1, \ldots, n$, $k=1,\ldots, F_i$, can have a non zero value, equal to $m(i,k)$, only for values of *k* different from P(*i*).

3. Lower bounds

Using the strong similarities that exist between F1 and the well-known Quadratic Assignment Problem (QAP), we have transposed two lower bounds originally studied for the QAP to the FAP. Specifically, we have adapted the same ideas of the bound of Gilmore (1962) and Lawler (1963) for the QAP, to derive two lower bounds, *LB1a* and *LB1b*, for problem F1.

In the following we will use a_i to denote the frequency assigned to link *i* in an optimal solution of F1 and $c'_{ikjl} = \begin{cases} c_{ikjl} & \text{if } j > i \\ c_{jlik} & \text{if } j < i \end{cases}$.

Theorem 3.1

$$LB1a = \sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \sum_{j \in \Gamma(i);\, j>i} \min_{l=1,..,F_j} \{ c_{ikjl} \} \right\} \tag{7}$$

is a lower bound to the optimal solution cost of problem F1.

Proof

$$\sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \sum_{j \in \Gamma(i);\, j>i} \min_{l=1,..,F_j} \{ c_{ikjl} \} \right\} \leq$$

$$\leq \sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \sum_{j \in \Gamma(i);\, j>i} c_{ikja_j} \right\} \leq$$

$$\leq \sum_{i=1}^{n} m_{ia_i} + \sum_{i=1}^{n} \sum_{j \in \Gamma(i);\, j>i} c_{ia_i ja_j} = z_{F1}.$$

■

Theorem 3.2

$$LB1b = \left\lceil \sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \frac{1}{2} \sum_{j\in\Gamma(i)} \min_{l=1,..,F_j} \left\{ c'_{jlik} \right\} \right\} \right\rceil \tag{8}$$

is a lower bound to the optimal solution cost of problem F1.

Proof

$$\sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \frac{1}{2} \sum_{j\in\Gamma(i)} \min_{l=1,..,F_j} \left\{ c'_{jlik} \right\} \right\} =$$

$$= \sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \frac{1}{2} \sum_{j\in\Gamma(i);\, j<i} \min_{l=1,..,F_j} \left\{ c_{jlik} \right\} + \frac{1}{2} \sum_{j\in\Gamma(i);\, j>i} \min_{l=1,..,F_j} \left\{ c_{ikjl} \right\} \right\} \le$$

$$\le \sum_{i=1}^{n} \min_{k=1,..,F_i} \left\{ m_{ik} + \frac{1}{2} \sum_{j\in\Gamma(i);\, j<i} c_{ja_jik} + \frac{1}{2} \sum_{j\in\Gamma(i);\, j>i} c_{ikja_j} \right\} \le$$

$$\le \sum_{i=1}^{n} \left(m_{ia_i} + \frac{1}{2} \sum_{j\in\Gamma(i);\, j<i} c_{ja_jia_i} + \frac{1}{2} \sum_{j\in\Gamma(i);\, j>i} c_{ia_ija_j} \right) =$$

$$= \sum_{i=1}^{n} m_{ia_i} + \sum_{i=1}^{n} \sum_{j\in\Gamma(i);\, j>i} c_{ia_ija_j} = z_{F1}.$$

Moreover, as all costs c_{ikjl}, i,j=1,.., n; k=1,.., F_i, l = 1,.., F_j, are integer, also the cost of the optimal solution must be integer, and it is thus possible to round to above ($\lceil\ \rceil$) the value of the bound.

■

Bounds *LB1a* and *LB1b* often yield different results, without having one dominate the other.

Let d denote the average dimension of the links' frequencies domains, which is obviously a determiner of length of the problem input string. The computational complexity of the *LB1* bounds results to be $O(n^2d^2)$. The computational time needed to obtain both *LB1a* and *LB1b* is therefore small, but the quality of the results obtained may be poor. Following the approach that Christofides and Gerrard (1981) used for deriving their bound for the QAP, we can however consider more cost components, obtaining two other lower bounds, *LB2a* and *LB2b*, and tighten *LB1a* and *LB1b*, respectively.

The derivation is similar to that of *LB1 a* and *b*, but we explicitly consider in the bound more cost components; this increases the computational time but permits to *LB2 a* and *b* to find better bounds than the ones found by the corresponding *LB1*s.

To compute the new bounds, we partition the links into pairs and we examine all combinations of available frequencies for each pair. This improves the LB1 bounds, where we simply consider the available frequencies for each link singularly: working on pairs permits in fact to explicitly take into account the interference constraints that can exist between paired links.

Formally, let C be a partition of N into pairs ($|C| = n/2$), where λ_1 and λ_2 are the first and second links of the pair $\lambda \in C$, respectively. In case n is odd, it is necessary to insert a dummy link with only one available frequency and with no constraint. We also define

$$c''_{ikjl} = \begin{cases} c_{ikjl} & \text{if } j \in \Gamma(i) \text{ and } j > i \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad c'''_{ikjl} = \begin{cases} c_{ikjl} & \text{if } j \in \Gamma(i) \text{ and } j > i \\ c_{jlik} & \text{if } j \in \Gamma(i) \text{ and } j < i \\ 0 & \text{otherwise} \end{cases}.$$

It is now possible to state the following results.

Theorem 3.3

$$LB2a = \sum_{\lambda \in C} \min_{\substack{k=1,..,F_{\lambda_1} \\ l=1,..,F_{\lambda_2}}} \left\{ m_{\lambda_1 k} + m_{\lambda_2 l} + c'''_{\lambda_1 k \lambda_2 l} + \sum_{\substack{h \in N; \\ h \notin \lambda}} \min_{m=1,..,F_h} \left\{ c''_{\lambda_1 khm} + c''_{\lambda_2 lhm} \right\} \right\} \tag{9}$$

is a lower bound to the cost of the optimal solution of FAP.

Proof

The proof is an adaptation of that of Theorem 3.1 and it is reported in the Appendix

■

Theorem 3.4

$$LB2b = \left\lceil \sum_{\lambda \in C} \min_{\substack{k=1,..,F_{\lambda_1} \\ l=1,..,F_{\lambda_2}}} \left\{ m_{\lambda_1 k} + m_{\lambda_2 l} + c'''_{\lambda_1 k \lambda_2 l} + \frac{1}{2} \sum_{\substack{h \in N; \\ h \notin \lambda}} \min_{m=1,..,F_h} \left\{ c'''_{\lambda_1 khm} + c'''_{\lambda_2 lhm} \right\} \right\} \right\rceil \quad (10)$$

is a lower bound to the cost of the optimal solution of the problem.

Proof

The proof is an adaptation of that of Theorem 3.2 and it is reported in the Appendix

■

Also in this case is not possible to prove a dominance between *LB2a* and *LB2b*. The computational complexity of these two lower bounds is clearly higher than that of *LB1a* and *LB1b*, specifically it is $O(n^2d^3)$.

A crucial element that affects the quality of bounds *LB2a* and *LB2b* is the partition of the set of links N into pairs. An effective choice of the pairs is essential to obtain better results than those obtained with the lower bounds *LB1a* and *LB1b*. The partitioning of the graph G so to maximize the bounds is a complex problem, therefore we suggest to use the heuristic procedure presented in Figure 2.

```
Procedure Select_pairs
Input: P                            /* P = Problem instance of type FAPMI */
Output: C                           /* the set of pairs */

Set T ← N                           /* N = link set of problem P */
Set C ← ∅
repeat
    Set i ← argmax_{k∈T}{|Γ(k)|}
    Set T ← T\{i}
    If |T∩Γ(i)|=0
      then j=argmax_{k∈T}{|Γ(k)|}
      else set j ← argmax_{k∈T∩Γ(i)}{|Γ(k)|}
    Set T ← T\{j}, C ← C U {(i,j)}
until (T ≠ ∅)
```

Figure 2. Pseudocode of the procedure used for partitioning the set N into pairs

In order to anticipate the computation of high bound values, a feature which can be of use to decrease the CPU time when the bounds are utilized in a branch and bound framework, as reported in Section 5, we also sort the pairs by decreasing values of the sum of the degrees of the pair elements, as computed on the interference graph G.

It is possible to further generalize bounds *LB2x* (i.e., bounds *LB2a* and *LB2b*) in the same way as bounds *LB1x* were generalized into *LB2x*, that is by considering progressively longer *l*-tuples. We thus obtain bounds LB*l*a and LB*lb*. These bounds are increasingly tighter, as they explicitly consider more cost components, but they are increasingly heavier to compute (because of a computational complexity of $O(n^2d^{l+1})$). Some computational results to this regard are provided in Section 6.
To obtain good results with these bounds, it is essential to use an effective strategy for selecting the *l*-tuples which will be considered in the bound. Even though we have no normative result, the most effective strategy we found is that of fig.2, that is to select as *l*-tuples the current set of *l* links which maximizes the weighted sum of their incoming edges, using distances as weights. Then the tuple is eliminated from the graph and the process is repeated as long as there are unselected links.

4. Reduction and dominance rules

Is this section we describe results which can be used to reduce the size of FAP instances or, given a partial assignment, to determine that a better partial solution can be defined for those same links, leading to a lower cost complete solution. In the following, we denote with f_i the frequency assigned to link *i*.

Rule *R1*

This rule is similar to the one presented by Smith and Hurley (1997) and can be applied to problem instances where links do not have mobility costs.

Theorem 4.1
Let *a* be a generic node of the graph *G* and let $b_1,..., b_k$ be the nodes connected to *a* in G; if condition (11) is true

$$F_a > \sum_{i=1}^{k}\left(2d(a,b_i)-1\right) \tag{11}$$

and $m(i) = 0$, $i=1,.., n$, then the node *a* and all arcs (a, b_i), $i=1,.., k$ can be eliminated from the problem.

Proof
For each $i=1,.., k$ the constraint between *a* and b_i forbids no more than $2d(a,b_i)-1$ frequencies to the link associated with *a*. Consider the case where each b_i, $i = 1,.., k$, has

an assigned frequency that forbids exactly $2d(a, b_i)$-1 frequencies and assume that the sets of forbidden frequencies are all disjoint. The total number of precluded frequencies for a is thus equal to $\sum_{i=1}^{k}\left(2d(a,b_i)-1\right)$. Because of (11), a has a number of available frequencies greater than this, so there will always be a frequency that can be assigned to a with no cost.

■

In case of instances with mobility costs, rule *R1* can be used as a dominance rule as detailed in Corollary 4.1.

Corollary 4.1

Let $\wp$ be a partial assignment of frequencies to the links of an instance, let a be a generic node of the graph G and let $b_1,..., b_k$ be the nodes connected to a in G. If condition (11) is true, node a has no frequency assigned and all nodes b_i, i=1,.., k are assigned a frequency in $\wp$, then the node a and all arcs (a, b_i), i=1,.., k can be eliminated from the remaining subproblem.

Figure 3 presents the described situation.

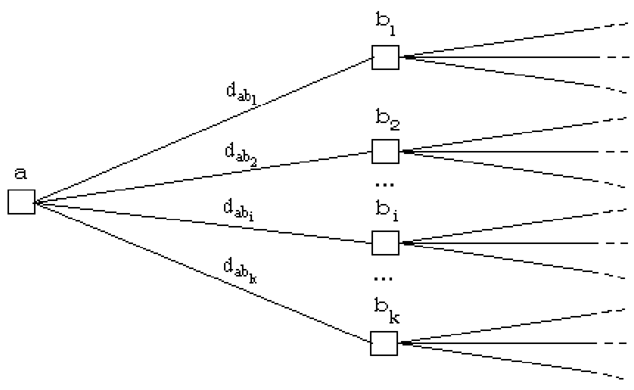

Figure 3. Conditions for the use of R1

Rule *R2*

This rule can be applied to reduce the size of instances where the frequency domains are integer intervals, there are no mobility costs and where the problem is to decide about the existence of an interference-free solution.

Theorem 4.2

In a problem where $\mathfrak{F}_i = \{1,..., F_i\}$, $m(i)=0$, $i=1,.., n$, and where the objective is to find an interference-free solution, if there is a node *a*, connected in *G* with exactly two nodes, called *b* and *c*, for which

$$d(b,c) \geq d(a,b) + d(a,c) \tag{12}$$

and $F_a \geq F_c + d(a,c)$ (assuming without loss of generality that $F_b < F_c$), then the node *a* and the arcs (a,b) and (a,c) can be eliminated from *G*.

Proof

If the objective is to find a solution with no interference, no constraint can be violated and in particular the constraint associated to the arc *(b, c)* must be satisfied. In this case, also the constraints corresponding to arcs *(a, b)* and *(a, c)* are automatically not violated and the frequency $f_a = f_c + d(a,c)$ can be assigned to the link corresponding to node *a*.

■

In case of instances with mobility costs and possible nonzero objective, rule *R2* can be used as a dominance rule.

Corollary 4.2

Let $\mathcal{P}$ be a partial assignment of frequencies to the links of an instance. If there is a node *a*, connected in *G* with exactly two nodes, called *b* and *c*, for which condition (12) holds, $F_a \geq F_c + d(a,c)$ (assuming without loss of generality that $F_b < F_c$), and node *a* has no frequency assigned while nodes *b* and *c* are assigned a frequency in $\mathcal{P}$ which satisfies the *d(b,c)* distance constraint, then the node *a* and the arcs (a,b) and (a,c) can be eliminated from the remaining subproblem.

A graphical representation of the described situation is shown in Figure 4.

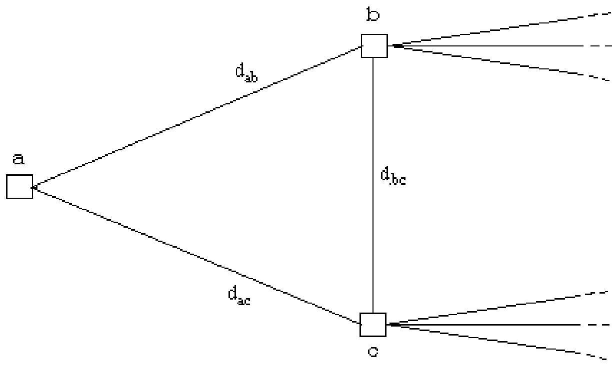

Figure 4. Conditions for the use of R2

Rule *R3*

This rule is similar to the one presented by Aardal et al. (1995) and can be applied to reduce instances where the problem is to decide about the existence of an interference-free solution, and where there are no mobility costs.

Theorem 4.3

In a problem where $m(i)=0$, $i=1,..,n$, and where the objective is to find an interference-free solution, if the graph G contains two nodes, a and b, such that

$$(a,b) \notin E \tag{13}$$

$$\mathfrak{F}(b) \subseteq \mathfrak{F}(a) \tag{14}$$

$$c \in \Gamma(a) \Rightarrow c \in \Gamma(b) \text{ and } d(a,c) \leq d(b,c) \tag{15}$$

then the node a and all the arcs connected to it can be eliminated from the graph G.

Proof

If a frequency can be assigned to the link corresponding to b without generating any interference with any link c, $\forall c \in \Gamma(b)$, then it is possible to assign that frequency also to a without incurring in any interference cost. In fact, since $f_b \geq f_c+d(b,c)$, it is possible to set $f_a = f_b \geq f_c+d(a,c)$ because $d(a,c) \geq d(b,c)$. This is feasible because $f_a \in F(a)$ for (14) and the links associated to a and b can receive the same frequency due to (13).

■

In case of instances with mobility costs and possible nonzero objective, rule *R3* can be used as a dominance rule.

Corollary 4.3

Let $\mathcal{P}$ be a partial assignment of frequencies to the links of an instance, let a and b be two nodes of the graph G for which conditions (13), (14) and (15) are true. If node a has no frequency assigned while node b is assigned a frequency in $\mathcal{P}$ which causes no interference cost, then the node a and all its outgoing the arcs can be eliminated from the remaining subproblem.

A graphical representation of the described situation is shown in Figure 5.

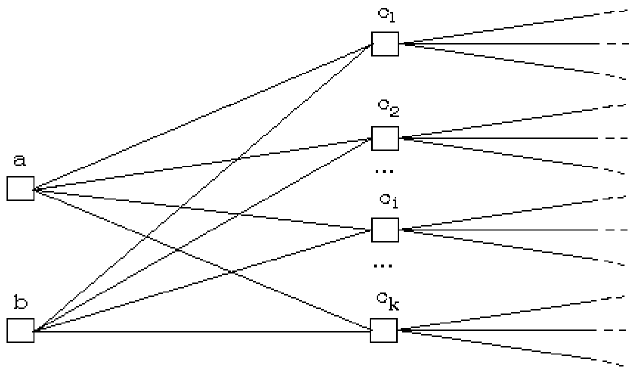

Figure 5. Conditions for the use of R3

5. The Branch and Bound algorithm

An optimal FAP solution can be obtained by means of the following branch and bound algorithm, based on formulation F1 of Section 3. Each node of the search tree corresponds to a partial frequency assignment obtained by adding to the partial assignment of *t* links, associated to the father node, a new link assignment. The state of a node α at level $h(\alpha)$ of the tree can thus be represented by a ordered list of pairs, $\mathcal{P}(\alpha) = \{(l_1,f(l_1)), (l_2, f(l_2)), \dots , (l_{h(\alpha)}, f(l_{h(\alpha)})\}$, where $f(l_i)$ is the frequency assigned to the l_i-th link.

A forward branching, from node α at level $h(\alpha)$, involves the generation of $F(l_{h(\alpha)})$ descendent nodes, one for each f_l, $l \leq F(l_{h(\alpha)})$. A node of the search tree can be eliminated either if the corresponding lower bound becomes greater or equal to the cost of the incumbent solution or if it is possible to prove, by means of some dominance rule, that such node cannot lead to the optimal solution. In the branch and bound procedure, we use the dominance criteria introduced in Section 4 on every subproblem met during the search that respects the previously described applicability conditions.

At each node of the search tree we create a new PCSP instance, with a link for each unassigned link of the original problem. The interference costs are equal to those of the original FAP, while the new mobility costs m_{ik}, for each i and k=1,.. F_i, do not represent only the original mobility costs, but also the interference costs between the already assigned links and the available frequencies for the unassigned links. Practically, for each unassigned link *i* and for every frequency *k* available for it that could generate interference with the already assigned frequencies, we add in m_{ik} both the original mobility costs and the arising interference costs. Since we can have different m_{ik} values for a same link *i* but different *k*, the subproblem solved at each node of the search tree are not real FAP, but more generically a PCSP.

In the following we give the outline of a tree search algorithm, called BBFAP, which makes use of the reduction rules *R1*, *R2* and *R3* and the lower bounds *LB2a* and *LB2b*. We denote by LB2a(α) and LB2b(α) the values of the lower bound *LB2a* and *LB2b* computed at node α of the search tree.

The general structure of the algorithm follows a depth first strategy. A branching from node α involves the selection of a link *l* to be considered at level $h(\alpha+1)$ and the generation of F_l descendent nodes. The link *l* is the unassigned link with the highest value of $v_l = \sum_{j\in\Gamma(l)} d(l,j)$. This strategy tries to make the algorithm assign first the most difficult links of the problem.

In Figure 6 we present a pseudocode of our branch and bound algorithm, where we indicate with X the best solution found and with z_{UB} the relative cost.

Procedure BBFAP

input: P /* *P* = Problem instance of type *FAPMI* */
output: X, z_{UB}

/ Initializations */*

Compute a heuristic solution X of P, **set** z_{UB} ← cost(X)

Set α ← 0, X{α} ← ∅, level{α} ← 0, OPEN ← {α}. */* push */*

Order the links in L by decreasing $v_l = \sum_{j\in\Gamma(l)} d(l,j)$, $l\in$ L. Let O be the index ordering vector.

While OPEN ≠ ∅

/ choose node for branching */*

Pop α from OPEN

Compute bound LB(α)= max{LB2a(α),LB2b(α)};

If LB(α) ≥ z_{UB} **then** go to EndLoop

If the solution corresponding to LB(α) is feasible, **then** update zUB=LB(α) and X.

Apply dominance rules 1, 2 and 3 to node α. **If** node α is dominated **then** go to EndLoop

/ branching */*

Set *l* ← O(level(α)+1)

```
    For each frequency f ∈ F(l)
        Generate a node β, with X(β) ← X(α) ∪ (f)
        Set level(β) ← level(α)+1, OPEN ← OPEN ∪ {β}
    End for
Label: EndLoop
End while
```

Figure 6. Pseudocode of branch and bound algorithm BBFAP

The heuristic used for initialization was the ANTS algorithm presented in Maniezzo, Carbonaro, Montemanni (1999).

6. Computational Results

The algorithm presented in this paper was coded in ANSI C and tested on a Pentium II PC, 400 MHz with 128 MB of RAM running under Linux v2.0.32. Our computational study used three sets of test problems with the objective of evaluating the performance both of the new lower bounds (*LB1a*, *LB1b*, *LB2a*, *LB2b*) and of the new exact algorithm BBFAP.

The benchmark data sets are the following.

CELAR: these problems were presented in 1994 within the EUCLID (EUropean Cooperation for the Long term In Defence) CALMA (Combinatorial ALgorithms for Military Applications) project, of the "Centre d'Electronique de l'Armament". These problems have all the characteristics of the *FAPMI* model, plus the so-called *parallel links*, which are pair of links for which the assigned frequencies must be exactly at a predefined distance (mandatory constraints). In the CELAR instances, each link has one parallel link, and for each frequency available for a link there is one and only one frequency of its parallel link that satisfies the required distance constraint.

It is possible to represent the CELAR benchmarks by means of formulation F1 creating a new instance as described below.

For each pair (q,u) of parallel links of the original instance (*CEL*) we create a link i of the new instance (represented following formulation *F1*) where the frequency domain of i is equal to the domain of q. For each two pairs, (q,u) and (v,w), of the original instance, corresponding to new links i and j, respectively, the cost c_{ikjl} has a value equal to the sum of the interference costs incurred when the link q uses its *k-th* frequency, the link u the frequency parallel to k, v uses its *l-th* frequency and w the one parallel to l. In the same way, m_{ik} has value equal to the sum of the mobility costs incurred into when q uses its *k-th* frequency and u the parallel one.

PHILADELPHIA: these benchmarks (Smith et al., 1999) are derived from an istance proposed by Anderson (1973). They are *span* minimization Frequency Assignment Problems which can be directly represented by the *FAPMI* model. It is possible to apply them the rules *R1*, *R2* and *R3* both as problem reduction and as pruning rules.

COLORING: in this set we have collected some of the best known Graph Coloring instances. It is possible to represent these problems by formulation F1, and to apply them the rules *R1*, *R2* and *R3* both as problem reduction and as pruning rules.

In the following tables all results presented were obtained with a maximum CPU time constraint of 1 hour; after that time the processes were aborted and the relative rows in the tables show a *n.a.* value.

Table 1 presents the results obtained by the lower bounds *LB1a*, *LB1b*, *LB2a* and *LB2b* on instances, obtained from CELAR problems, with optimum value greater than 0 (so that it is possible to evaluate the quality of the lower bounds). The columns show the following data.

Problem: problem identifier. By CELARxx-yy we indicate an instance obtained considering only the first yy links of the problem CELARxx.

Dimension: number of links and of constraints of the instance.

Best known UB: Best known upper bound to the cost of the optimal solution.

Lower Bounds LB1: values of lower bounds *LB1a* and *LB1b*, and CPU time to compute either of them.

Lower Bounds LB2: values of lower bounds *LB2a* and *LB2b*, and CPU time to compute either of them.

Table 1. Lower bounds comparison.

Problem	Dimension		Best known UB	Lower Bounds *LB1*			Lower Bounds *LB2*		
	nr links	nr constr		*LB1a*	*LB1b*	time (sec)	*LB2a*	*LB2b*	time (sec)
CELAR06-40	40	70	11	0	0	0	0	6	0
CELAR06-50	50	151	67	0	0	0	11	12	0
CELAR06-60	60	190	155	0	0	0	100	106	0
CELAR06-70	70	251	155	0	0	0	100	106	0
CELAR06	200	1 322	3 437	0	0	0	167	421	5

CELAR07-70	70	243	20 003	0	0	0	0	5 202	1
CELAR07-80	80	279	20 003	0	0	0	0	5 202	1
CELAR07-90	90	296	20 003	0	0	0	0	5 202	1
CELAR07	400	2 865	343 594	0	0	2	40 207	55 658	12
CELAR08	916	5 744	262	0	0	25	37	53	49
CELAR09-40	40	5	200	200	200	0	200	200	0
CELAR09-80	80	183	200	200	200	0	200	200	0
CELAR09-120	120	387	620	620	520	0	620	520	0
CELAR09-160	160	525	962	940	851	0	962	862	0
CELAR09-200	200	783	3 147	662	662	0	2 905	2 911	0
CELAR09	680	4 103	15 571	8 657	8 657	4	9 860	9 989	6
CELAR10-60	60	127	2 000	2 000	2 000	0	2 000	2 000	0
CELAR10-100	100	265	2 200	2 200	2 200	0	2 200	2 200	0
CELAR10-140	140	425	2 603	2 603	2 603	0	2 603	2 603	0
CELAR10-180	188	695	5 026	5 020	5 020	0	5 026	5 023	0
CELAR10-220	220	949	6 463	5 332	5 333	0	6 447	6 449	0
CELAR10-260	260	1 158	8 565	7 434	7 435	0	8 547	8 548	0
CELAR10-300	300	1 327	9 970	8 839	8 840	0	9 952	9 953	0
CELAR10-340	340	1 553	9 975	8 644	8 744	0	9 655	9 809	0
CELAR10-380	380	1 807	9 981	8 647	8 747	0	9 658	9 812	0
CELAR10-420	420	2 017	10 198	8 852	8 952	0	10 029	9 876	1
CELAR10	680	4 103	31 516	24 794	24 794	2	27 609	28 559	3

Notice that the *LB2* bounds always obtain results better or equal to those obtained by the *LB1* ones, but with higher computation times.

The overall quality of the bounds is not always good, in fact in some cases it is poor. The best performance is on instances obtained from CELAR09 and CELAR10, where the bounds obtained are good and can be computed in very short time. This is because these problems describe situation of old networks expansion and thus have positive mobility costs, with values comparable with those of the c_{ikjl}. This property is well exploited by our lower bounds, leading to the results presented.

Table 2 presents results obtained applying to some of the most difficult instances of Table 1 the lower bounds than can be obtained explicitly considering preogressively bigger *l*-tuples, as detailed in Section 3, with *l* ranging from 2 to 5. The columns with the bound values present the best result between the *a* and the *b* version of the corresponding bound, i.e., *LB3* = max {*LB3a*, *LB3b*}.

Table 2. High order lower bounds.

problem	nr lnk	nr constr	UB	LB2	sec	LB3	sec	LB4	sec	LB5	sec
CELAR 06-50	50	151	67	12	0	54	10	32	33	55	2 400
CELAR 06-60	60	190	155	106	0	143	10	127	34	149	2 403
CELAR 06-80	80	316	378	65	0	148	9	198	145	204	2 579
CELAR 06	200	1 322	3 437	421	4	725	91	935	2 349	*n.a.*	*n.a.*

CELAR 07-90	90	296	20 003	5 202	1	5 252	16	5 252	171	5 252	535
CELAR 07	400	2 865	343 594	55 658	13	101 785	211	152 948	3 120	*n.a.*	*n.a.*
CELAR 08-210	210	802	0	0	1	0	1	0	126	0	1 752.
CELAR 08	916	5 744	262	51	48	88	375	113	3559	*n.a.*	*n.a.*
CELAR 09-180	180	695	3 094	3 063	0	3 063	0	3 094	3	3 094	5
CELAR 09	680	4 103	15 571	9 989	6	13 275	13	13 479	86	14 781	1 372
CELAR 10-220	220	949	6 463	5 412	5	6 463	0	6 463	0	6 463	0
CELAR 10	680	4 103	31 516	28 556	3	29 525	4	31 021	7	31 024	14

It is possible to notice how the quality of the bound rapidly increases, and in fact in two cases (CELAR09-180 and CELAR10-220) it can prove the optimality of the available upper bound, but the CPU time to obtain them increases too. A trade-off analysis is required, to identify which bound is most proficient in the context of a specific optimization algorithm. Cells marked with *n.a.* represent instances for which the corresponding bound could not be obtained within the time limit of 3600 CPU seconds.

In Table 3 we show the results obtained by rules *R1*, *R2* and *R3* used as problem reduction rules on some PHILADELPHIA and COLORING problems. We show results only for instances where at least one link is eliminated. The columns show.

Problem: problem identifier.

Dimension: number of links (NL) and of constraints (NC) of the instance.

AF: number of frequencies (colors) available for use.

Rule R1: number of eliminated nodes (EN), number of eliminated arcs (EA) and CPU time in seconds (T) for *R1*.

Rule R2: number of eliminated nodes (EN), number of eliminated arcs (EA) and CPU time in seconds (T) for *R2*.

Rule R3: number of eliminated nodes (EN), number of eliminated arcs (EA) and CPU time in seconds (T) for *R3*.

Rule R1+R2+R3: number of eliminated nodes (EN), number of eliminated arcs (EA) and CPU time in seconds (T) for the successive application of rules *R1, R2* and *R3*.

Table 3. Reduction rules applied to PHILADELPHIA and COLORING problems.

Problem	**Dimension**		**AF**	**Rule *R1***			**Rule *R2***			**Rule *R3***			**Rules *R1+R2+R3***		
	NL	**NC**		**EN**	**EA**	**T**	**EN**	**EA**	**T**	**EN**	**EA**	**CPU**	**EN**	**EA**	**T**
PHIL05	481	76 979	426	121	16 712	0	0	0	0	0	0	2	121	16 712	2
PHIL06	481	97 835	426	25	4 547	4	0	0	0	0	0	2	25	4 547	4
PHIL07	481	93 288	426	25	4 547	4	0	0	0	0	0	2	25	4 547	4

PHIL08	481	97 835	426	54	11 532	0	0	0	0	0	0	16	54	11 532	2
PHIL09	962	391 821	855	46	18 211	1	0	0	1	0	0	21	46	18 211	34
fpsol2.i.3	425	8 688	30	294	5 706	0	0	0	0	198	3 331	1983	339	6 762	0
fpsol2.i.2	451	8 691	30	318	5 649	0	0	0	0	224	3 334	4	365	6 765	0
le450_25b	450	8 263	25	142	1 716	0	0	0	0	3	8	11	156	2 023	17
le450_25a	450	8 260	25	165	1 973	0	0	0	0	5	14	11	186	2 429	13
mulsol.i.5	186	3 973	31	110	2 559	0	0	0	0	76	1 704	0	114	2 680	0
mulsol.i.4	185	3 946	31	108	2 500	0	0	0	0	74	1 645	0	112	2 621	0
mulsol.i.3	184	3 916	31	108	2 500	0	0	0	0	75	1 676	0	113	2 652	0
mulsol.i.2	188	3 885	31	113	2 500	0	0	0	0	79	1 645	0	117	2 621	0
mulsol.i.1	197	3 925	49	109	1 691	0	0	0	0	94	1 269	0	197	3 925	0
zeroin.i.3	206	3 540	30	102	2 092	0	0	0	0	105	144	0	206	3 540	0
zeroin.i.2	211	3 541	30	136	2 093	0	0	0	0	110	1 412	0	211	3 541	0
zeroin.i.1	211	4 100	49	149	2 333	0	0	0	0	115	1 323	0	211	4 100	0
anna	138	493	10	115	349	0	0	0	0	67	172	0	138	493	0
anna	138	493	11	119	385	0	0	0	0	67	172	0	138	493	0
david	87	406	10	65	275	0	0	0	0	27	73	0	87	406	0
david	87	406	11	67	288	0	0	0	0	27	73	0	87	406	0
huck	74	301	10	58	192	0	0	0	0	14	30	0	74	301	0
huck	74	301	11	68	288	0	0	0	0	14	30	0	74	301	0
jean	80	254	9	58	192	0	0	0	0	35	57	0	80	254	0
jean	80	254	10	66	233	0	0	0	0	35	57	0	80	254	0
miles250	128	387	7	90	222	0	0	0	0	19	47	0	120	359	0
miles250	128	387	8	100	256	0	0	0	0	19	47	0	128	387	0
miles500	128	1 170	20	84	645	0	0	0	0	4	39	0	128	1 170	0
miles750	128	2 113	31	68	938	0	0	0	0	6	115	0	128	2 113	0
miles1000	128	3 216	42	49	971	0	0	0	0	4	115	0	128	3 216	0
myciel3	11	20	4	6	15	0	0	0	0	0	0	0	11	20	0
myciel4	23	71	5	5	20	0	0	0	0	0	0	0	6	25	0
myciel5	47	236	6	5	25	0	0	0	0	0	0	0	5	25	0
myciel6	95	755	7	5	30	0	0	0	0	0	0	0	5	30	0
myciel7	191	2 360	8	5	35	0	0	0	0	0	0	0	5	35	1

Rule *R2* fails on all the presented instances because they have, for all i,j=1,..., n, the distances $d(i,j)$=1, thus condition (12) can never be satisfied. For similar structural reasons, rule *R3* always fails on the PHILADELPHIA instances.

In general *R1* obtains better results than *R3*, in fact it is able to eliminate a larger number of nodes of the graph on all the problems. Notice however that the set of nodes eliminated by *R3* is not a subset of those eliminated by *R1*.

The quality of the reduction varies widely for different instances: while in fact there are instances for which no node or arc can be eliminated, some other ones can be fully solved by means of the reduction rules alone.

For what concerns the computation time, we can observe that they are extremely low: in most of the cases less then 1 second.

Table 4 presents the results obtained by BBFAP on subproblems obtained from the CELAR benchmarks. The columns show:

Problem: problem identifier.

Dimension: number of links and of constraints of the instance.

BBFAP: optimal solution value (ZOPT), number of nodes generated in the tree search and time needed to solve the problem to optimality, in seconds.

Table 4. Results obtained by BBFAP on CELAR type subproblems.

Problem	Dimension		BBFAP		
	nr links	nr constr	ZOPT	nr nodes	time (sec)
CELAR06-40	40	70	11	20	0
CELAR06-50	50	151	67	90	7
CELAR06-60	60	190	155	144	25
CELAR06-70	70	251	155	149	42
CELAR06-80	80	316	n.a.	n.a.	n.a.
CELAR07-70	70	243	20 003	475	42
CELAR07-80	80	279	20 003	637	74
CELAR07-90	90	296	20 003	645	75
CELAR07-100	100	389	n.a.	n.a.	n.a.
CELAR08-50	50	135	0	25	0
CELAR08-90	90	240	0	45	1
CELAR08-130	130	412	0	65	3
CELAR08-170	170	570	0	85	6
CELAR08-210	210	802	0	105	11
CELAR08-250	250	1 032	n.a.	n.a.	n.a.
CELAR09-40	40	58	200	9	0
CELAR09-80	80	183	200	12	0
CELAR09-120	120	387	620	19	0
CELAR09-160	160	525	962	25	0
CELAR09-200	200	783	3 147	913	431
CELAR09-240	240	1 062	n.a.	n.a.	n.a.
CELAR10-60	60	127	2 000	6	0
CELAR10-100	100	265	2 200	10	0
CELAR10-140	140	425	2 603	16	0
CELAR10-180	180	695	5 026	19	0
CELAR10-220	220	949	6 463	212	3
CELAR10-260	260	1 158	8 565	90	2
CELAR10-300	300	1 327	9 970	96	2
CELAR10-340	340	1 553	9 975	1 007	708
CELAR10-380	380	1 807	9 981	1 585	701
CELAR10-420	420	2 017	10 198	3 362	1 555
CELAR10-460	460	2 346	n.a.	n.a.	n.a

Notice in Table 4 how the performance of BBFAP decreases unevenly with the increase of the size of the instance to solve. While in fact smaller instances can be solved in negligible CPU time, the addition of a comparatively small number of links can lead to a significant decrease in the effectiveness of the approach. This testifies the possibility of being deceived by a poor quality bound, which could be effective on a given subproblem but becomes loose after the addition a small number of problem elements.

Table 4 confirms also an observation already introduced for Table 1, that is that on instances obtained from CELAR09 and CELAR10, which represent expansions of existing networks, the good quality of the bound induces a good effectiveness of BBFAP.

In Table 5 we group the results obtained by BBFAP on some span minimization problems obtained from PHILADELPHIA family and modeled according to F1. We present only results about problems on which our algorithm have completed the visit of the search tree. The columns show:

Problem: problem identifier, with the convention by which PHILxx-yy is the problem obtained form PHILxx considering for each cell *i* of the problem PHILxx (see Smith et al. 1999) only *m[i]/*yy links, where *m[i]* is the original connection request for cell *i*.

Dimension: number of links and of constraints of the instance.

Nr available frequencies: number of frequencies available at each link.

BBFAP: flag indicating the existence or not of a feasible solution with the given number of frequencies (*feasible sol*), number of nodes generated in the tree search and time needed to solve the problem to optimality, in seconds.

Table 5. Results obtained by BBFAP on problems of type PHILADELPHIA

Problem	Dimension		nr available frequencies	BBFAP		
	nr links	nr constr		feasible sol	nr nodes	time (sec)
PHIL01-10	42	429	16	No	476 414	2 347
PHIL01-10	42	429	19	Yes	3 501	9
PHIL02-10	42	637	16	No	464 738	2 271
PHIL02-10	42	637	24	Yes	10 381	17
PHIL03-10	41	420	21	No	103 782	632
PHIL03-10	41	420	24	Yes	48	0
PHIL04-10	41	573	19	No	156 161	802
PHIL04-10	41	573	26	Yes	497	0
PHIL05-10	36	449	30	No	4 481	41
PHIL05-10	36	449	37	Yes	688 786	1 614
PHIL06-10	36	548	30	No	4 881	47
PHIL06-10	36	548	37	Yes	251	1
PHIL06b-10	36	548	42	No	11 383	184
PHIL06b-10	36	548	43	Yes	34	0

PHIL07-10	36	548	21	No	402 455	1 745
PHIL07-10	36	548	37	Yes	23 519	36
PHIL08-10	36	548	20	No	359 847	1 639
PHIL08-10	36	548	38	Yes	2 335	3
PHIL09-10	88	3 260	35	No	76 716	3 240
PHIL09-10	88	3 260	88	Yes	79	13

It is obviously computationally more challenging to discover that a problem has no solution than to find an optimal solution, when one exists. Moreover, the frequency domains, which are larger than in the CELAR case, make the branching rule proposed more inefficient: in fact generating an offspring for each possible frequency assignment for the link associated to the next level of the search tree becomes demanding on problem instances with many available frequencies and no mobility costs, such as these ones. This reflects on the fact that in Table 5 we have been able to obtain for each problem instance an upper and a lower bound on the minimal number of frequencies needed to find a zero cost solution, but only in one case (PHIL06b) we could narrow - under the given max CPU time constraint – the interval to a single frequency, thereby finding the optimal value.

Finally, Table 6 presents results obtained by BBFAP on some well-known coloring problem instances. As mentioned in Section 2, FAP is a generalization of coloring, thus an exact method for frequency assignment can also be used as an exact method for solving coloring instances. The particular objective function we consider in this work, minimization of penalties due to unsatisfied constraints given a frequency spectrum, can be used to model decision coloring problems, i.e. problems where given a graph and a number of colors the question is whether the graph is colorable with that many colors. The columns of Table 6 show:

Problem: problem identifier.

Dimension: number of links and of constraints of the instance.

Nr. colors: number of colors made available.

Feasible solution: whether or not a feasible solution exists.

BBFAP: flag indicating the existence or not of a feasible solution with the given number of frequencies (*feasible sol*), number of nodes generated in the tree search and CPU time needed by BBFAP to find a feasible solution for the problem, or to prove that none exists.

Table 6. Results obtained by BBFAP on coloring problem instances.

Problem	Dimension		nr colors	BBFAP		
	nr links	nr constr		feasible sol	nr nodes	time (sec)
queen5_5	25	160	4	No	8	0
queen5_5	25	160	5	Yes	21	0
queen6_6	36	290	6	No	114 701	182
queen6_6	36	290	7	Yes	1 601	1
queen7_7	49	476	6	No	53 968	193
queen7_7	49	476	7	Yes	1 090	3
queen8_12	96	1 368	12	Yes	449 020	1 689
fpsol2.i.3	425	8.688	30	Yes	21	3
fpsol2.i.2	451	8 691	30	Yes	21	3
le450_4a	450	5 714	4	No	953	864
le450_5c	450	9.803	4	No	237	170
le450_5d	450	9 757	4	No	132	103
le450_25b	450	8 263	25	Yes	96	65
le450_25a	450	8 260	25	Yes	86	50
mulsol.i.5	186	3 973	31	Yes	3	0
mulsol.i.4	185	3 946	31	Yes	4	0
mulsol.i.3	184	3 916	31	Yes	3	0
mulsol.i.2	188	3 885	31	Yes	3	0
mulsol.i.1	197	3 925	49	Yes	0	0
zeroin.i.3	206	3 540	30	Yes	0	0
zeroin.i.2	211	3 541	30	Yes	0	0
zeroin.i.1	211	4 100	49	Yes	0	0
anna	138	493	10	No	0	0
anna	138	493	11	Yes	0	0
david	87	406	10	No	0	0
david	87	406	11	Yes	0	0
huck	74	301	10	No	0	0
huck	74	301	11	Yes	0	0
jean	80	254	9	No	0	0
jean	80	254	10	Yes	0	0
miles250	128	387	7	No	1 100	0
miles250	128	387	8	Yes	0	0
miles500	128	1 170	20	Yes	0	0
miles750	128	2 113	31	Yes	0	0
miles1000	128	3 216	42	Yes	0	0
myciel3	11	20	3	No	31	0
myciel3	11	20	4	Yes	0	0
myciel4	23	71	4	No	3 358	1
myciel4	23	71	5	Yes	4	0
myciel5	47	236	6	Yes	15	0
myciel6	95	755	7	Yes	29	0
myciel7	191	2 360	8	Yes	47	3

DIMACS problem instances not reported in Table 6 could not be solved in 1 hour CPU time. Table 6, in accordance with Table 5, shows the greater difficulty of proving that no solution exists than on finding one, when it exists. This permits to instantiate a binary search on the number of colors, to find the minimum number of colors needed to color a given graph, only on relatively small problem instances. It is barely the case to mention that state-of-the-art exact algorithms specific for the coloring problem represent a more efficient alternative than our binary search for solving coloring problem instances.

A second observation is about the instances which could be solved with 0 nodes: for them the preprocessing using reduction rules *R1*, *R2* and *R3* was sufficient to find a feasible solution, with a negligible CPU time.

7. Conclusions

The work presented in this paper starts from the mathematical formulation of the Frequency Assignment Problem due to Koster, van Hoesel and Kolen (1997) to define new lower bounds, reduction and dominance rules and an exact tree search algorithm.

We identified a family of lower bounds and presented some detail about four of them, *LB1a*, *LB1b*, *LB2a* and *LB2b*, which are actually versions of two basic bounds. The first two, the *LB1x*, are derived from simple considerations on the structure of the objective function and of feasible solutions, the second two, the *LB2x*, strengthen the former ones by means of considerations similar to those used by Christofides and Gerrard (1981) on the QAP. Further strengthening can lead to bounds *LBlx*, which are progressively tighter but heavier to compute. The lower order bounds (i.e., the *LB1x* and *LB2x*) bounds are efficient to compute but they can be loose on some problem instance; nevertheless, they represent the first computationally tested contribution on problems where the objective is the minimization of the interference cost.

We propose three reduction/dominance rules, *R1*, *R2* and *R3*. These can be used as reduction rules on some type of problem instances, in order to a priori reduce the size of the problem to solve, and in all cases as dominance rule in a branch and bound context to prune dominated branches. Computational results show the effectiveness of these rules, which are able alone to solve a number of benchmarks from the literature.

Finally, the last contribution is the first computationally tested exact algorithm for interference cost minimization problems. The algorithm, called BBFAP, includes all elements introduced above and has been able to solve a number of instances derived from literature benchmarks. Moreover, being FAP a generalization of coloring, BBFAP has also been tested on benchmark coloring instances.

References

CALMA project (Combinatorial Algorithms for Military Applications), Europe. http://www.win.tue.nl/math/bs/comb_opt/hurkens/calma.html

Graph Coloring Instances. http://mat.gsia.cmu.edu/COLOR/instances.html.

K. I. Aardal, A. Hipolito, C. P. M. van Hoesel, B. Jansen, C. Roos, T. Terlaky (1995). A Branch-and-cut Algorithm for the Frequency Assignment Problem. Technical Annex T-2.2.1.A of the report of EUCLID CALMA radio link frequency assignment project. ftp://ftp.win.tue.nl/pub/techreports/CALMA/T221A.ps.

P. M. Adjakplé, B. Jaumard (1997). Greedy and Tabu Search Heuristics for Channel Block Assignment in Cellular Systems. *Les Cahiers du GERAD G-97-45*.

L. G. Anderson (1973). A simulation study of some dynamic channel assignment algorithms in high capacity mobile telecommunications system. *IEEE Transactions on Communications 21*, 1294-1301.

R. Borndörfer, A. Eisenblätter, M. Grötschel, A. Martin (1998). Frequency Assignment in Cellular Phone Networks. *Annals of Operations Research 76*, 73-93.

N. Christofides, M. Gerrard (1981). A Graph Theoretic Analysis Of Bounds for the Quadratic Assignment Problem. *Studies On Graphs and Discrete Programming*. Amsterdam, North Holland 61-68.

D. Costa (1993). On the use of some known methods for *T*-colorings of graphs. *Annals of Operations Research 41*, 343-358.

D. de Werra, Y. Gay (1994). Chromatic scheduling and frequency assignment. *Discrete Applied Mathematics 49*, 165-174.

M. Fischetti, C. Lepschy, G. Minerva, G. Romanin-Jacur, E. Toto (1997). Frequency Assignment in Mobile Radio Systems using Branch-and-cut Techniques. *Submitted for publication*.

A. Gamst (1986). Some Lower Bounds for a Class of Frequency Assignment Problems. *IEEE Transactions on Vehicular Technology VT-35(1)*, 8-14.

A. Gamst , W. Rave (1982). On Frequency Assignment in Mobile Automatic Telephone Systems. *Proc. Globecom'82*, B3.1.1-B3.1.7.

P. Gilmore (1962). Optimal and Suboptimal Algorithms for the Quadratic Assignment Problem. *SIAM Journal on Applied Mathematics (10)*, 305-313.

A. I. Giortzis, L. F. Turner (1996). A Mathematical Programming Approach to the Channel Assignment Problem in Radio Networks. *Proceedings IEEE 46th Vehicular Technology Conference 2*, 736-740.

C. A. J. Hurkens, S. R. Tiourine (1995). Upper and Lower Bounding Techniques for Frequency Assignment Problems. ftp://ftp.win.tue.nl/pub/techreports/cosor/95-34.ps.

B. Jaumard, O. Marcotte, C. Meyer (1998). Mathematical Models and Exact Methods for Channel Assignment in Cellular Networks. *Les Cahiers du GERAD G-98-13.*

S. E. Karish, E. Çela, J. Clausen, T. Espersen (1998). A dual framework for lower bounds of the quadratic assignment problem based on linearizazion. *SFB Report 120, Department of Mathematics, Technical University Graz, Austria*.

M. G. Kazantzakis, P. P. Demestichas, M. E. Anagnostou (1995). Optimum Frequency Reuse in Mobile Telephony Systems. *International Journal of Communication Systems 8*, 185-190.

A. M. C. A. Koster, S. P. M. van Hoesel, A. W. J. Kolen (1997). The Partial Constraint Satisfaction Problem: Facets and Lifting Theorems. *Research Memorandum rm97-005*,. ftp://ftp.unimaas.nl/pub/grey_files/fdew/1997/rm97-005.ps.

E. Lawler (1963). The Quadratic Assignment Problem. *Management Science (9)*, 586-599.

V. Maniezzo, A. Carbobaro, R. Montemanni (1999). An approach to Frequency Assignment Problem based on an ANTS Heuristic. *MIC'99 III Metaheuristics International Conference*.

C. Mannino, A. Sassano (1999). An enumerative algorithm for the frequency assignment problem. *Discrete Applied Mathematics*, to appear.

D. H. Smith, S. Hurley (1997). Bounds for the Frequency Assignment Problem. *Discrete Mathematics 167-168*, 571-582.

D. H. Smith, S.Hurley, S.U.Thiel (1998). Improving Heuristics for the Frequency Assignment Problem. *European Journal of Operational Research*, 107, 76-86.

D. H. Smith, S. Hurley, S. M. Allen (1999). A New Lower Bound for the Channel Assignment Problem. *University of Glamorgan.*

D.-W. Tcha, Y.-J. Chung, T.-J. Choi (1997). A New Lower Bound for the Frequency Assignment Problem. *IEEE/ACM Transactions on Networking 5(1)*, 34-39.

E. Tsang (1993). Foundations of Constraint Satisfaction. *Academic Press*.

J. P. Warners, T. Terlaky, C. Roos, B. Jansen (1997). A Potential Reduction Approach to the Frequency Assignment Problem. *Discrete Applied Mathematics 78*, 251-282.

Appendix

In this appendix we prove Theorem 3.3 and Theorem 3.4, that is the validity of lower bounds *LB2a* and *LB2b*, respectively.

Proof of Theorem 3.3

$$\sum_{\lambda\in C} \min_{\substack{k=1,..,F_{\lambda_1}\\ l=1,..,F_{\lambda_2}}} \left\{ m_{\lambda_1 k} + m_{\lambda_2 l} + c'''_{\lambda_1 k \lambda_2 l} + \sum_{\substack{h\in N;\\ h\notin\lambda}} \min_{m=1,..,F_h} \left\{ c''_{\lambda_1 khm} + c''_{\lambda_2 lhm} \right\} \right\} \le$$

$$\le \sum_{\lambda\in C} \min_{\substack{k=1,..,F_{\lambda_1}\\ l=1,..,F_{\lambda_2}}} \left\{ m_{\lambda_1 k} + m_{\lambda_2 l} + c'''_{\lambda_1 k \lambda_2 l} + \sum_{\substack{h\in N;\\ h\notin\lambda}} \left(c''_{\lambda_1 kha_h} + c''_{\lambda_2 lha_h} \right) \right\} \le$$

$$\le \sum_{\lambda\in C} \left(m_{\lambda_1 a_{\lambda_1}} + m_{\lambda_2 a_{\lambda_2}} + c'''_{\lambda_1 a_{\lambda_1} \lambda_2 a_{\lambda_2}} + \sum_{\substack{h\in N;\\ h\notin\lambda}} \left(c''_{\lambda_1 a_{\lambda_1} ha_h} + c''_{\lambda_2 a_{\lambda_2} ha_h} \right) \right) =$$

$$= \sum_{\lambda\in C} \left(m_{\lambda_1 a_{\lambda_1}} + m_{\lambda_2 a_{\lambda_2}} + c'''_{\lambda_1 a_{\lambda_1} \lambda_2 a_{\lambda_2}} + \sum_{\substack{j\in\Gamma(\lambda_1);\\ j\neq\lambda_2;\\ j>\lambda_1}} c_{\lambda_1 a_{\lambda_1} ja_j} + \sum_{\substack{j\in\Gamma(\lambda_2);\\ j\neq\lambda_1;\\ j>\lambda_2}} c_{\lambda_2 a_{\lambda_2} ja_j} \right) =$$

$$= \sum_{i=1}^{n} m_{ia_i} + \sum_{i=1}^{n} \sum_{j\in\Gamma(i);\, j>i} c_{ia_i ja_j} = z_{F1}.$$

■

Proof of Theorem 3.4

$$\sum_{\lambda\in C} \min_{\substack{k=1,..,F_{\lambda_1}\\ l=1,..,F_{\lambda_2}}} \left\{ m_{\lambda_1 k} + m_{\lambda_2 l} + c'''_{\lambda_1 k \lambda_2 l} + \frac{1}{2} \sum_{\substack{h\in N;\\ h\notin\lambda}} \min_{m=1,..,F_h} \left\{ c'''_{\lambda_1 khm} + c'''_{\lambda_2 lhm} \right\} \right\} \le$$

$$\le \sum_{\lambda\in C} \min_{\substack{k=1,..,F_{\lambda_1}\\ l=1,..,F_{\lambda_2}}} \left\{ m_{\lambda_1 k} + m_{\lambda_2 l} + c'''_{\lambda_1 k \lambda_2 l} + + \frac{1}{2} \sum_{\substack{h\in N;\\ h\notin\lambda}} \left(c'''_{\lambda_1 kha_h} + c'''_{\lambda_2 lha_h} \right) \right\} \le$$

$$\leq \sum_{\lambda \in C}\left(m_{\lambda_1 a_{\lambda_1}} + m_{\lambda_2 a_{\lambda_2}} + c'''_{\lambda_1 a_{\lambda_1} \lambda_2 a_{\lambda_2}} + \frac{1}{2}\sum_{\substack{h \in N;\\ h \notin \lambda}} \left(c'''_{\lambda_1 a_{\lambda_1} h a_h} + c'''_{\lambda_2 a_{\lambda_2} h a_h} \right) \right) =$$

$$= \sum_{\lambda \in C}\left(m_{\lambda_1 a_{\lambda_1}} + m_{\lambda_2 a_{\lambda_2}} + c'''_{\lambda_1 a_{\lambda_1} \lambda_2 a_{\lambda_2}} + \frac{1}{2}\left(\sum_{\substack{j \in \Gamma(\lambda_1);\\ j \neq \lambda_2}} c'_{\lambda_1 a_{\lambda_1} j a_j} + \sum_{\substack{j \in \Gamma(\lambda_2);\\ j \neq \lambda_1}} c'_{\lambda_2 a_{\lambda_2} j a_j} \right)\right) =$$

$$= \sum_{\lambda \in C}\left(\begin{array}{l} m_{\lambda_1 a_{\lambda_1}} + m_{\lambda_2 a_{\lambda_2}} + \frac{1}{2} c'''_{\lambda_1 a_{\lambda_1} \lambda_2 a_{\lambda_2}} + \frac{1}{2} c'''_{\lambda_2 a_{\lambda_2} \lambda_1 a_{\lambda_1}} + \\ + \frac{1}{2}\left(\begin{array}{l} \sum_{j \in \Gamma(\lambda_1);\, j \neq \lambda_2;\, j < \lambda_1} c_{j a_j \lambda_1 a_{\lambda_1}} + \sum_{j \in \Gamma(\lambda_1);\, j \neq \lambda_2;\, j > \lambda_1} c_{\lambda_1 a_{\lambda_1} j a_j} + \\ + \sum_{j \in \Gamma(\lambda_2);\, j \neq \lambda_1;\, j < \lambda_2} c_{j a_j \lambda_2 a_{\lambda_2}} + \sum_{j \in \Gamma(\lambda_2);\, j \neq \lambda_1;\, j > \lambda_2} c_{\lambda_2 a_{\lambda_2} j a_j} \end{array} \right) \end{array} \right) =$$

$$\leq \sum_{i=1}^{n}\left(m_{i a_i} + \frac{1}{2}\sum_{j \in \Gamma(i);\, j < i} c_{j a_j i a_i} + \frac{1}{2}\sum_{j \in \Gamma(i);\, j > i} c_{i a_i j a_j} \right) =$$

$$= \sum_{i=1}^{n} m_{i a_i} + \sum_{i=1}^{n}\sum_{j \in \Gamma(i);\, j > i} c_{i a_i j a_j} = z_{F1}.$$

Now, as all costs c_{ikjl}, $i,j=1,.., n$; $k=1,.., F_i$ and $l = 1,.., F_j$, are integer also the cost of the optimal solution must be integer, and it is thus possible to round to above ($\lceil\ \rceil$) the value of the bound.

■