ELSEVIER

Discrete Applied Mathematics 119 (2002) 59–78

DISCRETE APPLIED MATHEMATICS

Generation of lower bounds for minimum span frequency assignment

S.M. Allen[a], D.H. Smith[a,*], S. Hurley[b]

[a] *Div. Maths. and Computing, University of Glamorgan, Pontypridd, Mid Glamorgan CF37 1DL, UK*
[b] *Department of Computer Science, University of Wales, Cardiff, P.O. Box 916, Cardiff CF2 3XF, UK*

Received 4 July 1999; received in revised form 8 June 2000; accepted 12 August 2000

Abstract

Minimum span frequency assignment problems require lower bounds for the span in order to assess the quality of individual assignments, and to effectively compare different meta-heuristic algorithms. The generation of good lower bounds requires the identification of critical subproblems. These subproblems are subgraphs of a constraint graph. Sometimes clique subgraphs lead to good lower bounds. However for some problems the clique must be modified in order to obtain the best possible bound. This can be time consuming, requires manual intervention and is dependent on the initial clique obtained and the specific problem. For practical use, a simple bounding technique is needed that can be universally applied to all problems without modification. Two algorithms based on meta-heuristics are presented that aim to find a subproblem that gives the best possible lower bound. This avoids the need for clique detection routines and manual intervention, and gives a robust and automatic method for generating lower bounds for all minimum span problems. These algorithms use mathematical programming formulations of the frequency assignment problem. Extensive testing on a wide range of problems shows that bounds from the new algorithms match those using previous techniques, and in some cases are significantly better.

Keywords: Radio frequency assignment; Lower bounds; Subgraphs

1. Introduction

Given a radio network, the minimum span frequency assignment problem is to assign frequencies to the transmitters of the network, so that

(1) a given level of service quality is met at all possible reception points.

* Corresponding author.
E-mail addresses: stuart.m.allen@cs.cf.ac.uk (S.M. Allen), dhsmith@glam.ac.uk (D.H. Smith), s.hurley@cs.cf.ac.uk (S. Hurley).

0166-218X/02/$ - see front matter
PII: S0166-218X(01)00265-7

(2) the spectrum used, indicated by the difference between the largest frequency used and the smallest frequency used (the *span*) is minimised.

This is a computationally hard problem. For networks with more than about 30 transmitters, exact algorithms take a prohibitive amount of time. Meta-heuristics have been used successfully to generate good quality frequency assignments [4,5]. Although a meta-heuristic algorithm cannot be guaranteed to find a best possible solution, it will ideally find a near optimal solution quickly. Therefore, in order to assess the quality of individual assignments, and to effectively compare different algorithms, it is important to be able to generate good quality lower bounds for the span of an assignment. The lower bounding techniques described in [10,8,2] have been found to be good, giving tight results in many cases. However, they depend on being able to identify a critical subproblem in the network that determines the span of the full network. This paper demonstrates how the bounding techniques themselves can be used as the cost function of a meta-heuristic algorithm which determines the best subproblem to use. This process normally matches or improves the best bounds known and fully automates the determination of lower bounds. A refinement of the methods for cellular problems is also described.

2. Lower bounding techniques

2.1. Graph theoretic representation of the frequency assignment problem

The minimum span frequency assignment problem is commonly modelled using constraints concerning the frequencies that can be allocated to pairs of transmitters (referred to as binary constraints). This leads to a graph theoretic representation of the problem.

Definition 1. A *constraint graph* is a complete graph with vertices corresponding to the transmitters of a network. Each edge, ij of the graph is labelled with an integer c_{ij} that gives a frequency separation requirement between the corresponding transmitters.

Note that this definition of a constraint graph is equivalent to the definition of the modified constraint graph G' in [10]

Definition 2. A *frequency assignment* (or *channel assignment*) in a constraint graph G is a mapping $f : V(G) \to \{0, 1, 2, \ldots, K\}$ such that

$$|f(v_i) - f(v_j)| \geqslant c_{ij}$$

for all $v_i, v_j \in V(G)$. Such an assignment is also referred to as a *zero-violation assignment*. If some of the constraints are violated then f is an assignment with constraint violations. The value of K is the *span* of the assignment.

Definition 3. If K is a minimum over all zero-violation assignments, then the assignment is a *minimal assignment*. The minimal value of K is the *minimum span* of G, denoted $sp(G)$.

2.2. Lower bounding techniques for minimum span frequency assignment

For a full description of lower bounding techniques for minimum span frequency assignment see [10,9]. This paper will only make use of the most successful bounds.

2.3. Bounds from the travelling salesman problem

It was noted in [7] that Hamiltonian paths (paths that include all vertices exactly once) in a constraint graph are related to minimum span frequency assignments. In particular, any zero-violation frequency assignment can be used to construct a Hamiltonian path in G whose cost is at least that of the minimum such path. This gives the following result.

Theorem 4. *If G is a constraint graph, and $H(G)$ is the value of a minimum weight Hamiltonian path in G, then $sp(G) \geqslant H(G)$.*

Proof. See [10,9,8]. □

Finding $H(G)$ is also known as the open symmetric travelling salesman problem (TSP) and so this bound is referred to as the *TSP bound*. Any Hamiltonian path in a constraint graph can be used to construct a frequency assignment as follows:

- Relabel the vertices so that $\{v_1, \ldots, v_{|V(G)|}\}$ is the Hamiltonian path of minimum weight.
- Let

$$f(v_1) = f_0,$$

$$f(v_i) = f(v_{i-1}) + c_{i\ i-1} \quad \text{for } i = 2, \ldots, |V(G)|.$$

However, this assignment may contain constraint violations between vertices that are not consecutive in the path. This can be seen, for example, for the assignment obtained from the path v_1, v_2, v_3, v_4 shown in Fig. 1:

Although the TSP bound can be used successfully for many frequency assignment problems [8,11,5], it has two drawbacks that prevent it from generating tight bounds in all situations.

(1) *Calculation*. Finding a minimum weight Hamiltonian path is an NP-hard problem and so it is not always possible to find a solution in reasonable time.
(2) *Tightness*. Since the TSP bound ignores frequency constraints between non-consecutive vertices, a frequency assignment constructed from a minimum weight

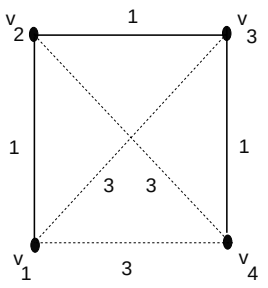

Fig. 1. If a frequency assignment is constructed from the path v_1, v_2, v_3, v_4 then the frequencies assigned to v_1 and v_3 will differ by 2, violating the constraint between them.

Hamiltonian path may contain constraint violations as indicated above. These ignored constraints mean that for some problems the TSP bound is not tight.

2.4. Bounds using mathematical programming

The drawbacks of the TSP bound can be addressed by considering an integer programming formulation of the bound. The graph G_0 is constructed from G by adding a dummy vertex v_0 joined by a single edge of weight 0 to all other vertices of G. A Hamiltonian circuit in G_0 is then equivalent to a Hamiltonian path in G. The variables x_{ij} specify the edges contained in the minimum weight Hamiltonian circuit:

$x_{ij} = 0$ if edge ij is not in the circuit,

$x_{ij} = 1$ if edge ij is in the circuit.

The TSP bound can be formulated as an integer program (TSP IP) as follows:

$$\text{Minimize} \quad \sum_{ij \in E(G_0)} c_{ij} x_{ij}$$

$$\text{subject to} \quad \sum_{j:ij \in E(G_0)} x_{ij} = 2; \quad i \in V(G_0) \quad \text{(degree constraints)},$$

$$\sum_{i \in S, j \in V(G_0) \setminus S} x_{ij} \geqslant 2; \quad S \subset V(G_0) \quad \text{(subtour elimination constraints)},$$

$$x_{ij} \in \{0, 1\}; \quad ij \in E(G_0). \quad \text{(integrality constraints)}.$$

The problems with the TSP bound can be addressed as follows:

(1) *Calculation*. Relaxations of the TSP can be used to generate bounds more quickly. The relaxations used here are the linear programming relaxation obtained by replacing the integrality constraints by $0 \leqslant x_{ij} \leqslant 1$ and the perfect two matching relaxation (PTMP) obtained by omitting the subtour elimination constraints and therefore

allowing a union of several circuits. These relaxations make the calculation easier, but the bounds obtained may potentially become worse. The combination of these two relaxations gives the PTMP LP bound.

(2) *Tightness*. Extra constraints can be added to the integer program to take account of the frequency assignment constraints ignored by the original formulation. Although this may improve the bound obtained where possible, the time taken to obtain solutions may be significantly increased.

In [2] it is shown that a solution to PTMP is a good approximation to the TSP bound for frequency assignment problems. The PTMP has the advantage of being easier to solve; a polynomial time algorithm exists [6]. It is also guaranteed that solutions to the linear programming relaxation will have all variables taking half-integral values (i.e. each x_{ij} takes the value of either 0, $\frac{1}{2}$ or 1). The formulation of PTMP is

$$\text{Minimize} \quad \sum_{ij \in E(G_0)} c_{ij} x_{ij}$$

$$\text{subject to} \quad \sum_{j:ij \in E(G_0)} x_{ij} = 2; \quad i \in V(G_0) \quad \text{(degree constraints)},$$

$$x_{ij} \in \{0,1\}; \quad ij \in E(G_0). \quad \text{(integrality constraints)}.$$

In [2] extra constraints are described that improve the TSP bound by considering the frequency separation constraints between non-consecutive vertices on the Hamiltonian path. These constraints are based on additional *excess* variables for each edge of the path. An assignment can then be constructed from the path using both the edge weights and the excess variables to allow constraints to be satisfied with some slack. Again, the vertices are relabelled so that $\{v_1, v_2, \ldots, v_{|V(G)|}\}$ is the Hamiltonian path of minimum cost. Then let

$$f(v_1) = f_0,$$

$$f(v_i) = f(v_{i-1}) + c_{i\,i-1} + e_{i\,i-1} \quad \text{for } i = 2, \ldots, |V(G)|,$$

where e_{ij} is the value of the excess variable for edge ij. Constraints between non-consecutive vertices can now be satisfied in the example of Fig. 1 by assigning an excess value of 1 to the edge v_2v_3 to obtain the assignment:

transmitter	v_1	v_2	v_3	v_4
frequency	0	1	3	4

Constraints involving the excess variables can be formulated by considering paths P in the constraint graph. Let

$$d(P) = c_{i_1 i_k} - (c_{i_1 i_2} + \cdots + c_{i_{k-1} i_k}),$$

where $P=\{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}$. $d(P)$ is called the *deficit* of the path and gives the amount by which the constraint between the end-vertices would fail if all other constraints were met exactly. Let $\mathscr{P}(G)$ denote the set of all paths P with positive deficit ($d(P)>0$) and let $\mathscr{P}_k(G)$ denote the subset of $\mathscr{P}(G)$ consisting of all paths with length k. The constraint for the path P with edge set $E(P)$ is then:

$$E_P \geqslant d(P)(X_P - (|E(P)| - 1)),$$

where $X_P = x_{i_1 i_2} + \cdots + x_{i_{k-1} i_k}$ and $E_P = e_{i_1 i_2} + \cdots + e_{i_{k-1} i_k}$. These constraints are referred to as *FAP constraints*. If all of the edges of P are chosen, $X_P = |E(P)|$ and the constraint becomes

$$E_P \geqslant d(P),$$

that is, enough excess must be added to overcome the deficit on the path. Otherwise E_P is unconstrained.

If a constraint is added to the TSP integer program for all paths in $\mathscr{P}(G)$, a complete mathematical programming formulation of the minimum span frequency assignment problem is obtained. However, generally it is only practical to obtain solutions when a subset of the FAP constraints are used. The subset $\mathscr{P}_2(G)$ is most practical computationally, and experiments over a wide range of frequency assignment problems suggest that it is often adequate for finding the best possible bound. In this case the constraints have the form

$$e_{ij} + e_{jk} \geqslant d(P)(x_{ij} + x_{jk} - 1).$$

Adding these to the TSP integer program would only make the solution harder to obtain. To obtain lower bounds in a reasonable time the PTMP LP bound is used with the extra constraints. This will be referred to here as PTMP FAP LP. The full formulation with integer constraints is:

$$\text{Minimize} \quad \sum_{ij \in E(G_0)} c_{ij} x_{ij} + \sum_{ij \in E(G)} e_{ij}$$

$$\text{subject to} \quad \sum_{j : ij \in E(G_0)} x_{ij} = 2; \quad v_i \in V(G_0) \quad \text{(degree constraints)},$$

$$E_P \geqslant d(P)(X_P - (|E(P)| - 1)) \text{ where } P \in \mathscr{P}(G),$$

(FAP constraints)

$$x_{ij} \in \{0, 1\}; \quad ij \in E(G_0),$$

$$e_{ij} \in \{0, 1, 2, \ldots\}; \quad ij \in E(G_0) \quad \text{(integrality constraints)}.$$

The results in [2] show that over a range of realistic data sets the PTMP LP bound is within 1% of the TSP bound, and the PTMP FAP LP is typically between 0.2% and 3% better than the TSP bound, depending on the problem type. The time taken to calculate the bounds is also significantly less than that of the TSP bound.

2.5. Bounds for cellular problems

Some problems, referred to here as cellular problems as they arise from a cellular structure, have several transmitters at a given site. All transmitters at site $\mathscr{C}_r$ have the same constraint label c_{rs} with transmitters at another site $\mathscr{C}_s$. Additionally, they have a (usually larger) constraint c_{rr} with the transmitters at the same site. Let $\mathbf{m} = (m_1, m_2, \ldots, m_{\text{no.sites}})$ be a requirements vector. Then m_r is the number of transmitters at site $\mathscr{C}_r$.

For non-cellular problems, the effect on the solution of the relaxation from an integer program to a linear program is generally small. This is true for both the PTMP and PTMP FAP bounds. For cellular problems, the relaxation is good for the PTMP bound. However, the relaxation for the PTMP FAP bound is particularly bad, and in most cases negates the effect of the additional FAP constraints. This is because all edges between any two cells have equal weights c_{ij} and so typically, the x_{ij} variables take non-integral values that avoid any non-zero excess values being incurred. By utilising the special structure of cellular problems the gap between the linear and integer programs can be reduced. The cellular structure gives rise to a *cellular constraint graph* $\bar{G}$ in which the vertices correspond to cells, and the edges correspond to the frequency separation constraint between the two cells. Specifically, each edge ij in the cellular constraint graph has a weight $\bar{c}_{ij}$ that is equal to the edge weight c_{rs} in the full constraint graph, where v_r corresponds to any transmitter in cell i and v_s corresponds to any transmitter in cell j. Whereas the full constraint graph is simple, the cellular constraint graph will have loops corresponding to the cosite constraint between transmitters in the same cell. Each vertex v_i of the cellular constraint graph is labelled with the required number m_i of transmitters, called the *demand* of the corresponding cell.

The integer program for finding a minimum weight perfect two matching can be simplified by using the cell constraint graph $\bar{G}$. Variables S_{ij} are introduced to represent the number of edges in a perfect two matching in G between vertices corresponding to cells i and j. As before, a dummy vertex v_0 is added to the cell constraint graph joined by a single edge of weight 0 to every other vertex to give the graph $\bar{G}_0$. The demand of the dummy vertex is taken to be 1.

$$\text{Minimize} \quad \sum_{ij \in E(\bar{G}_0)} \bar{c}_{ij} S_{ij}$$

subject to

$$2S_{ii} + \sum_{j : ij \in E(\bar{G}_0), i \neq j} S_{ij} = 2m_i; \quad i \in V(\bar{G}_0) \quad \text{(degree constraints)},$$

$$S_{ij} \in \{0, 1, \ldots, 2\min(m_i, m_j)\} \quad \text{for } ij \in E(\bar{G}_0).$$

(integrality constraints).

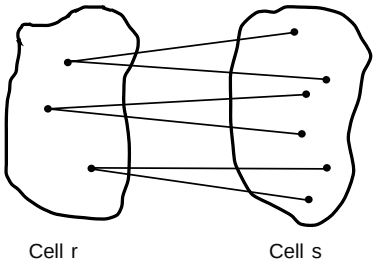

Fig. 2. The type of path used in the cellular formulation.

Any solution to this integer program can be used to construct a perfect two matching of minimum weight in the full constraint graph G. This can be done using a simple greedy algorithm. This cellular representation does not allow equivalent constraints for all paths in $\mathscr{P}(G)$ to be added conveniently. However, experimentation suggests that the most important constraints are based on the cosite constraints, and these constraints can be easily incorporated. Suppose $S_{rs} > m_r$ in a solution to PTMP IP (CELLULAR). Then any perfect two matching in G_0 constructed from this solution must contain at least $S_{rs} - m_r$ edge-disjoint paths of the form shown in Fig. 2:

If $\bar{c}_{ss} - 2\bar{c}_{rs} > 0$ then each of these paths in G_0 has positive deficit, and so requires an excess value of $\bar{c}_{ss} - 2\bar{c}_{rs}$. Since the paths are edge-disjoint, the total required excess on all paths must be at least $(\bar{c}_{ss} - 2\bar{c}_{rs})(S_{rs} - m_r)$. Hence if E_{rs} is the total excess required on edges between cells r and s, the following constraint must be satisfied:

$$E_{rs} \geqslant (\bar{c}_{ss} - 2\bar{c}_{rs})(S_{rs} - m_r).$$

Similarly, by considering $S_{rs} - m_s$, the following constraint is obtained:

$$E_{rs} \geqslant (\bar{c}_{rr} - 2\bar{c}_{rs})(S_{rs} - m_s).$$

Incorporating these constraints for each pair of cells into the PTMP IP (CELLULAR) gives the following integer program:

$$\text{Minimize} \quad \sum_{ij \in E(\bar{G}_0)} \bar{c}_{ij} S_{ij} + \sum_{ij \in E(\bar{G})} E_{ij}$$

subject to

$$2S_{ii} + \sum_{j : ij \in E(\bar{G}_0), i \neq j} S_{ij} = 2m_i; \quad i \in V(\bar{G}_0) \quad \text{(degree constraints)},$$

$$E_{rs} \geqslant (\bar{c}_{rr} - 2\bar{c}_{rs})(S_{rs} - m_r) \quad \text{for } \bar{c}_{rr} - 2\bar{c}_{rs} > 0,$$

$$E_{rs} \geqslant (\bar{c}_{ss} - 2\bar{c}_{rs})(S_{rs} - m_s) \quad \text{for } \bar{c}_{ss} - 2\bar{c}_{rs} > 0 \quad \text{(FAP constraints)},$$

$$S_{ij} \in \{0, 1, \ldots, 2\min(m_i, m_j)\} \quad \text{for } ij \in E(\bar{G}_0),$$

$$E_{ij} \in \{0, 1, \ldots\} \quad \text{for } ij \in E(\bar{G}_0) \quad \text{(integrality constraints)}.$$

3. Subproblems for generating lower bounds

3.1. Choice of subproblem: cliques

For most frequency assignment problems the constraint graphs contain large numbers of edges with weight zero. If the bounds described in Section 2 are applied to such a constraint graph, they will typically give a bound close to zero. However, better bounds can be obtained by considering a critical subgraph of the constraint graph. Clearly any lower bound for the span of a subproblem is also a lower bound for the span of the full problem. Cliques have been demonstrated [11,2] to be a good initial choice for determining bounds, particularly when information about the size of the frequency constraints is included.

Definition 5. A *level-p clique* in a constraint graph G is a complete subgraph in which each edge has label at least $p+1$, that is contained in no larger such subgraph.

If the bounding techniques are applied to level-p cliques ($p \geqslant 0$), then there are no edges with label 0, and the resulting bound is generally better than that for the full problem. Normally the best procedure is to choose the largest level-p clique in the constraint graph for each value of p, and find the best bound obtained for each of them.

Definition 6. A *maximal level-p clique* in a constraint graph G is the level-p clique that contains the largest number of vertices.

To find the maximal clique in a graph is an NP-hard problem, however for constraint graphs arising from frequency assignment problems it can often be solved in reasonable time. This can be done using an algorithm of Carraghan and Pardalos [3] together with orderings of the transmitters based on the set of constraints involving each transmitter [9]. Solutions can usually be obtained by this method for problems with up to about 800 transmitters. Although this may be too restrictive for most realistic cellular problems, by considering weighted cliques in the cellular constraint graph problems with up to about 800 cells can be handled [9].

Definition 7. A *maximal level-p weighted clique* in a cellular constraint graph $\bar{G}$ is a level-p clique such that all loops have labels at least $p+1$ (i.e. all cosite constraints are at least $p+1$), with the largest sum of demands.

3.2. Choice of subproblem: Extending cliques based on partial assignments

If the lower bound obtained from a clique matches the value of an assignment of the full problem then a tight bound has been obtained. If not, either the bound or the

assignment needs to be improved further. It is possible that the bound could be improved further by considering longer paths in the FAP constraints. However, the effect of paths of length 3 or more is usually negligible when linear relaxations are used. Another possibility is that the gap between the linear and integer programs is causing the difference. Both of these cases would require the solution of an integer program to improve the bound. Alternatively, it could be that the clique does not capture the full difficulty of the problem. Often the bound obtained from a clique is tight for the subproblem but not for the full problem. If the bound for the clique is not tight, a better subproblem can often be found by successively adding new vertices to the clique. A method of doing this was described in [11]. The method is sometimes successful, but not always; it can be time consuming to carry out and the exact rationale for the choices made is rather uncertain.

3.3. Heuristic generation of subproblems

An alternative method is to use a meta-heuristic algorithm that attempts to generate a subproblem that gives the best lower bound for the problem. Many meta-heuristics work on the principle of local search. An initial configuration is chosen and successive configurations are selected from a set of those similar to the current one. For example, for this problem, the neighbourhood of a subproblem may consist of all subproblems that can be obtained by adding or removing a single cell/transmitter. At each iteration a new configuration is selected from the neighbourhood of the current configuration, and tested against some criteria, for example, the value of a cost function. If the configuration is accepted the search continues from the new configuration, otherwise the new configuration is rejected and the search continues from the previous configuration. This process continues until some termination criteria are met, for example, when a fixed number of configurations have been tested, or if no improvement has been made for a given number of iterations.

For the heuristic generation of subproblems for the lower bounds, two types of neighbourhood have been considered. In the description that follows, "site" refers to a single cell in a cellular problem and to a single transmitter in non-cellular problems:

Random move. A site is chosen at random. If it is already in the subproblem it is removed, otherwise it is added to the current subproblem.

Connected move. A site is chosen at random from those sites that are either contained in the current subproblem or constrained with some site in the current subproblem. If it is already in the subproblem it is removed, otherwise it is added to the current subproblem.

The cost function used to evaluate each subproblem consists of a calculation of one of the bounding methods:

- PTMP LP
- PTMP FAP LP
- PTMP IP
- PTMP FAP IP

In practise the integer programming bounds are far too computationally expensive to be used.

The algorithms described can also be used with a cost function involving the size of the subproblem being considered, with the aim of encouraging the search to remove cells that have no effect on the bound. However, the effects on the bounds produced using this method appear to be insignificant. This type of cost function may be of use in finding large subproblems with large bounds. These can potentially be used as part of the assignment procedure.

To use meta-heuristic algorithms effectively it is essential to have a cost function that can be quickly and easily calculated; otherwise the total number of configurations that can be evaluated will be limited. The PTMP LP bound described earlier satisfies this property. For the results presented in this paper solutions to the linear programs were obtained using the commercial mathematical programming package CPLEX (see http://www.cplex.com/). The times taken in seconds (on a 200 MHz Pentium Pro, 64Mb RAM) to calculate the bounds for typical subproblems are:

	PTMP LP	PTMP FAP LP
32 cell clique in cellular problem	0.11	0.12
45 transmitter clique in non-cellular problem	0.23	1.41
45 randomly chosen transmitters in non-cellular problem	0.14	3.81

The non-cellular PTMP FAP LP bounds take considerably longer than equivalent sized cellular problems. This is due to the larger number of FAP constraints involved, for an N cell problem there are at most $N(N-1)$ FAP constraints, but for an N transmitter non-cellular problem there are at most $N(N-1)(N-2)/2$. This is also reflected in the time taken to formulate the constraints (the times shown above are the times for the solution to be obtained after it has been formulated). In both cases each possible constraint must be tested to see if the path has a positive deficit.

Using the PTMP LP bound as a cost function in a meta-heuristic to generate subproblems gives Algorithm 1, shown in Fig. 3. For cellular problems the PTMP FAP LP bound can be calculated quickly enough to allow it to be used in place of the PTMP LP bound to test every configuration.

For non-cellular problems the PTMP FAP LP bound is more time consuming and so for larger problems it is impractical to use as the cost function to test every configuration. Instead, the PTMP LP bound is used to guide the search. Additionally, whenever the PTMP LP bound for a configuration is within a specified percentage of

```
bestBound = 0                    // Reset overall bound and
currentRun = 0                   //   run counter.

WHILE ( currentRun < totalRuns )

  currentRun++                   // Increment run counter.
  subproblem.initialize          // Load initial configuration.

  WHILE (NOT meta-heuristic.isFinished )
                                 // Until termination criteria
                                 //   are met.
    subproblem.perturb           // Apply move and evaluate
    cost = subproblem.ptmp_lp    //   the cost function for the
    meta-heuristic.step( cost ) //   new configuration using the
                                 //   chosen meta-heuristic.
    IF ( meta-heuristic.isAccept )

      IF cost > bestBound        // Update best overall bound if
        bestBound = cost         //   necessary.
    END IF

    ELSE

      subproblem.reset           // Undo the last move

    END IF

  END WHILE

END WHILE
```

Fig. 3. Algorithm 1.

the best PTMP LP bound found so far, the PTMP FAP LP bound can be applied to the subproblem. This is justified by the strong correlation found experimentally between the PTMP LP and PTMP FAP LP bounds. This method is shown as Algorithm 2 in Fig. 4. Note that this algorithm will also give the best bound obtained by Algorithm 1 (as bestPtmpBound).

Meta-heuristics generally show a rapid improvement in the cost function initially, while improvement later takes longer. It is therefore beneficial to limit the time taken by the meta-heuristic on a single run and instead perform many runs, selecting the best bound overall. It is possible to use an initial configuration to start the search with a good subproblem, for example a clique. In practise, this requires a more careful tuning of the parameters for the chosen meta-heuristic. They must be set so that enough solutions are accepted initially to allow the search to move away from the local maximum; but the setting must also ensure that enough solutions are rejected so that the subproblems considered do not grow too large. Bounds for large subproblems take longer to compute and generally have a relatively large number of edges of weight zero, which leads to lower bounds. Better results are obtained by selecting no cells initially, as in this case the parameters can be set conservatively in order to prevent the subproblems growing

```
bestPtmpBound = 0              // Reset best PTMP LP bound (used
bestBound = 0                  // to guide the search), best
currentRun = 0                 // overall bound and run counter.

WHILE ( currentRun < totalRuns )

  currentRun++                 // Increment run counter.
  subproblem.initialize        // Load initial configuration.

  WHILE (NOT meta-heuristic.isFinished )
                               // Until termination criteria met.
    subproblem.perturb         // Apply move and evaluate
    cost = subproblem.ptmp_lp  // the cost function for the
    meta-heuristic.step( cost )// new configuration using the
                               // chosen meta-heuristic.
    IF ( meta-heuristic.isAccept )

      IF cost > bestPtmpBound  // Update best PTMP LP bound if
        bestPtmpBound = cost   // necessary.
      END IF

      IF ( subproblem.ptmp_lp > threshold
                               // If PTMP LP bound is within a
                               // specified percentage of the
                               // best so far, calculate the
                               // PTMP FAP LP bound for the
                               // new configuration and update
                               // as necessary.
        cost = subproblem.ptmp_fap_lp
        IF ( cost > bestBound )
            bestBound = cost
        END IF

      END IF

    ELSE

      subproblem.reset         // Undo the last move

    END IF
  END WHILE
END WHILE
```

Fig. 4. Algorithm 2.

too large. Although many of the runs will give a bound that is worse than that of the maximal clique, by selecting an appropriate number of runs a better bound will often be obtained as more of the search space will be explored.

Connected moves were found to give the best bounds in a given amount of time, particularly when starting with no initial subproblem. They also require less tuning of parameters to get the best results. Moves consisting of adding/removing multiple sites at a time were also considered but had no positive effect and for some problems had a negative effect. All results presented in this paper use a connected move, changing only one site in each move.

4. Implementation

4.1. Simulated annealing meta-heuristic

The simulated annealing meta-heuristic is based on the physical process of annealing. The description given here is appropriate to a maximization problem. A temperature is used to control the ability to move away from local maxima that are not global maxima. The temperature is given a high value initially that reduces during the course of the algorithm. At each iteration, the current configuration with cost C_{new} is tested against the last accepted configuration with cost C_{accept}. The current configuration is accepted if either

- $C_{\text{new}} > C_{\text{accept}}$, (that is, if the current configuration is an improvement)
- or $e^{(C_{\text{new}}-C_{\text{accept}})/kT_{\text{SA}}} > random$, where T_{SA} denotes the current temperature and random is a random number from $[0,1)$. The constant k is used to scale the probability of accepting a worse solution and should be set based on the problem data. That is, a worse configuration is accepted with a probability dependent on the temperature. As the temperature reduces, so does this probability. The probability is also dependent on the quality of the new configuration; the lower the cost of the new configuration, the less likely it is to be accepted. Accepting a configuration that does not improve the current configuration is referred to as a *tunnel* event.

After N_{trial} configurations have been tested at a temperature T_{SA} it is reduced to a new temperature $\phi(T_{\text{SA}})$. The value of N_{trial} and the function $\phi(T_{\text{SA}})$ depend on the cooling scheme used. After many experiments had been performed, a simple logarithmic cooling schedule was selected. This avoids the need to repeatedly spend time setting a suitable starting temperature. The total number of iterations tested is fixed as $N_{\log}$, with N_{trial} iterations at each of N_{drop} temperatures. Then $N_{\text{trial}} = \lceil N_{\log}^{(1-\gamma)} \rceil$ and $N_{\text{drop}} = \lceil N_{\log}^{(\gamma)} \rceil$, where $\gamma \in (0,1)$. The temperature at step i is defined as $width^{`N_{drop}/cold'-i}$, for $i = 1,2,\ldots,N_{\text{drop}}$, where $width$ and $cold$ are parameters of the cooling schedule.

4.2. Parameter tuning

All results presented in this paper used the logarithmic cooling schedule with parameters $N_{\log} = 3000$, $\gamma = 0.1$, $width = 2$ and $cold = 1$. Note that these parameters, obtained after some experimentation, only give three temperature steps. The number of runs performed should be at least 25, preferably between 50 and 100 if tight bounds are to be reliably obtained.

The only parameter that remains to be set is k, the choice of which is critical. The parameter k controls the amount of tunnelling that occurs, for a fixed set of temperatures given by the cooling schedule. If k is set too large the subproblem size grows too large and the bound quickly reduces to zero. Note that once either Algorithm 1 or 2 accepts a subproblem with bound 0, it is very hard for the simulated annealing to guide the search back to a positive bound. This is because a move to a subproblem with equal cost is always accepted as a tunnel event, as $e^{(0/kT_{\text{SA}})} > random$. This means that vertices

Table 1
Table headings used in tables of results

Number of transmitters in maximal level-0 clique	The size of the largest level-0 clique. Calculated using the algorithm described in [3] using orderings of the transmitters or cells [9].
Extended clique bound	A bound obtained by applying the TSP bound to a clique or a clique that has been extended by the successive addition of vertices. Where noted, the PTMP FAP LP bound has been used instead of the TSP bound.
Algorithm 1 (*PTMP LP*)	The best bound obtained using Algorithm 1 over 50 individual runs.
Algorithm 2 (*PTMP FAP LP*)	The best bound obtained using Algorithm 2 over 50 individual runs.
Upper bound	The span of the best known assignment. All assignments are made using FASOFT [5], a package for frequency assignment based on several meta-heuristic algorithms.

are likely to be added, and accepted, that do not increase the bound. This continues until each subproblem consists of approximately half of the vertices. At this point the probability of a random move adding a site is approximately equal to the probability of a site being removed from the subproblem. Thus the size of the subproblem remains relatively constant. Using a connected move instead of a random move improves the situation, making the choice of k less critical. The results in Section 5 are all generated using a value of 0.001 for k. Although this may not be the best value of k for some problems, it has the advantage of being widely applicable in a robust manner.

Further information justifying these choices of parameters can be found in [1].

5. Results

The table headings used to present the results in this section are shown in Table 1:

5.1. Philadelphia data sets

The Philadelphia problems make up a well-studied set of benchmark problems. These are based on a theoretical cellular network around Philadelphia, PA., USA and have appeared extensively in the literature [11,12]. The cellular structure is shown in Fig. 5. Transmitters are assumed to be located at cell centres. The demand (number of transmitters) in each cell is given by one of the requirement vectors in Table 2. The constraints between each pair of transmitters are characterised by six distances $d_0, d_1, \ldots, d_5$. If d

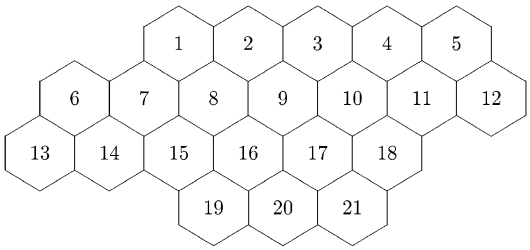

Fig. 5. The cellular geometry of the Philadelphia problem.

Table 2
Requirements vectors for the Philadelphia data sets

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	17	18	19	20	21											
D1	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
	20	20	20	20	20											
D2	5	5	5	8	12	25	30	25	30	40	40	45	20	30	25	15
	15	30	20	20	25											
D3	8	25	8	8	8	15	18	52	77	28	13	15	31	15	36	57
	28	8	10	13	8											

Table 3
Constraint sets for the Philadelphia data sets

	d_0	d_1	d_2	d_3	d_4	d_5
C1 [11]	$\sqrt{7}$	$\sqrt{3}$	1	1	1	0
C2 [11]	$\sqrt{12}$	$\sqrt{3}$	1	1	1	0
C3 [12]	$\sqrt{12}$	2	1	1	1	0

is the distance between transmitters i and j then

$$|f(v_i) - f(v_j)| \geqslant 0 \quad \text{if } d_0 \leqslant d,$$

$$|f(v_i) - f(v_j)| \geqslant k \quad \text{if } d_k \leqslant d < d_{k-1} \quad (k = 1, 2, \ldots, 5),$$

where the distance between adjacent cell centres is 1.

The constraint sets shown in Table 3 have appeared in the literature: The use of Algorithms 1 and 2 for all combinations of requirement vector and constraint set is shown in Table 4. For case D3, C3 the extended clique bound is obtained using PTMP FAP LP. Note that for all problems the best known bounds are matched or improved by Algorithm 2, with the exception of D1, C1. Here the best bound has recently been improved by H. Heller (private communication) and is obtained by a theoretical argument specific to this problem. Most of the upper bounds are found using the assignment methods described in [11].

Table 4
Results for Philadelphia data sets

Requirement vector/ Constraint set	Number of transmitters in maximal level-0 clique	Extended clique bound	Algorithm 1 PTMP LP/ Algorithm 2 PTMP FAP LP	Upper bound
D1/C1	140	179	177/177	179
D1/C2	240	239	239/239	239
D1/C3	240	239	239/239	259
D2/C1	180	252	252/252	252
D2/C2	258	257	257/257	257
D2/C3	258	258	290/300	324
D3/C1	275	426	426/426	426
D3/C2	360	426	426/426	426
D3/C3	360	524	483/524	524

Table 5
Results for real cellular problems

Data set	Number of transmitters in maximal level-0 clique	Extended clique bound	Algorithm 1 PTMP LP/ Algorithm 2 PTMP FAP LP	Upper bound
One	285	284	284/284	306
Two	282	336	336/336	336
Three	363	362	373/383	411

5.2. Practical cellular data sets

The results in Table 5 are for a set of real cellular problems supplied by a mobile telephone operator. The largest problem has 10,000 transmitters.

5.3. Radio links data sets

The results in Table 6 are for data sets constructed to represent military radio links problems. The extended clique bounds for these problems are derived from level-0 and level-1 cliques. The extended clique bound for T726b is obtained using PTMP FAP LP after extensive extension of a clique using a method similar to that described in [11]. The best bound from a number of trials was used.

Run times for Algorithm 2 are very variable, with the cellular problems much faster to solve than the non-cellular problems. For example for D3/C3 in Table 4, 50 runs were completed in less than 15 min. The bound of 524 was found on the third run. The bound of 524 was found 24 times, with a relatively poor bound of 380 found 26 times. For data set 2 in Table 5, which is much harder, 50 runs

Table 6
Results for military radio links problems

Data set	Number of transmitters in maximal level-0 clique	Extended clique bound	Algorithm 1 PTMP LP/ Algorithm 2 PTMP FAP LP	Upper bound
T95a	23	47	47/47	47
T95b	23	47	47/47	48
T252a	44	50	53/53	58
T252b	44	50	53/53	60
T282a	45	75	78/78	99
T282b	45	75	80/81	102
T410a	64	114	116/117	143
T410b	64	114	116/118	145
T450a	88	94	90/92	122
T450b	88	94	90/92	116
T490a	79	126	132/133	187
T490b	79	126	132/133	185
T726a	99	171	182/182	226
T726b	99	181	182/184	247

were completed in 45 min, with the best bound found only once on run 6. For the non-cellular problem T726a in Table 6, only two runs were completed in 2 h. The bound of 182 was found on the second run. None of the results quoted required more than an overnight run. Such run times are generally justified for frequency assignment problems.

6. Conclusions

The algorithms presented here can be considered to successfully provide an effective method to *assess* the quality of frequency assignments obtained by meta-heuristic or other methods. They have the following three properties:

- *Quality.* In all cases the bounds obtained by Algorithm 2 match or improve those obtained by applying a bound to a clique. The methods are also better than those obtained by manual extension of cliques. The only cases where any improvement has proved possible arise when a full integer programming search can be completed. This is not often practical for problems of a realistic size.
- *Ease of use.* Previously a clique detection routine was needed to derive the bounds. Although there are good algorithms for generating maximal cliques for constraint graphs [3,9], it is necessary to find several cliques to obtain the best bounds. In general, there is no way of determining in advance which level is required. Previous methods involving extension of cliques [11] required manual intervention and were very time consuming. Algorithm 2 gives good results without the need for clique detection or clique extension.

- *Robustness*. Algorithms 1 and 2 give good results applied to all problems without modification.

An implementation of Algorithms 1 and 2 has been extensively tested on a wide range of realistic problems. The main points arising from this experimentation are:

- Connected moves give better results faster than random moves.
- A single cell/transmitter should be changed in each move. Changing more than one cell needs careful tuning of the meta-heuristic parameters and provides no noticeable improvement.
- Algorithm 2 gives better results (where possible) than Algorithm 1, without a significant increase in run time. A threshold of approximately 95% should be used.
- No initial subproblem should be used, as this restricts the search too heavily.
- For the simulated annealing implementation described, the logarithmic cooling schedule gives good results while requiring less tuning of parameters. The parameters quoted were found to be successful for most problems. The value of 0.001 recommended for k is chosen to be robust, giving good results for all problems. For some problems better values of k may be found that encourage more tunnel events. This value can be from 0.005 to 0.5.

In some cases it can be seen that the bound obtained from PTMP FAP LP is significantly less than that of PTMP FAP IP (i.e. with integer programming). It is generally impractical to obtain a solution to full integer programs of PTMP FAP IP, however, better bounds may be possible by adding a selection of the integrality constraints. This may be either by identifying the critical integrality constraints or by using some random selection. Experimentation suggests that integrality constraints are the best way to improve these bounds further; extra frequency assignment constraints have little effect.

References

[1] S.M. Allen, Frequency assignment problems: subgraph generation for lower bounds, Technical Report UG-M-98-2, Division of Mathematics and Computing, University of Glamorgan, December 1998.

[2] S.M. Allen, D.H. Smith, S. Hurley, Lower bounding techniques for frequency assignment, Discrete Math. 197/198 (1999) 41–52.

[3] R. Carraghan, P. Pardalos, An exact algorithm for the maximum clique problem, Oper. Res. Lett. 9 (1990) 375–382.

[4] S. Hurley, D.H. Smith, Meta-heuristics and channel assignment, in: R. Leese (Ed.), Methods and Algorithms for Radio Channel Assignment, Oxford University Press, Oxford, to appear.

[5] S. Hurley, D.H. Smith, S.U. Thiel, FASoft: A system for discrete channel frequency assignment, Radio Sci. 32 (1997) 1921–1939.

[6] J.F. Pekny, D.L. Miller, A staged primal-dual algorithm for finding a minimum cost perfect two-matching in an undirected graph, ORSA J. Comput. 6 (1994) 68–81.

[7] A. Raychaudhuri, Intersection assignments, T-colourings and powers of graphs, Ph.D. Thesis, Rutgers University, 1985.

[8] D.H. Smith, S. Hurley, Bounds for the frequency assignment problem, Discrete Math. 167/168 (1997) 571–582.

[9] D.H. Smith, S. Hurley, S.M. Allen, A new lower bound for the channel assignment problem, IEEE Trans. Vehicular Technol. 49 (4) (2000) 1265–1272.

[10] D.H. Smith, S. Hurley, S.M. Allen, in: R. Leese (Ed.), Lower bounds for channel assignment, Methods and Algorithms for Radio Channel Assignment, Oxford University Press, Oxford, to appear.
[11] D.H. Smith, S. Hurley, S.U. Thiel, Improving heuristics for the frequency assignment problem, European J. Oper. Res. 107 (1998) 76–86.
[12] D-w. Tcha, Y-j. Chung, T-j. Choi, A new lower bound for the frequency assignment problem, IEEE/ACM Trans. Networking 5 (1997) 34–39.