Annals of Operations Research 107, 285–301, 2001

Characteristics of Good Meta-Heuristic Algorithms for the Frequency Assignment Problem

D.H. SMITH * dhsmith@glam.ac.uk
Div. Mathematics, University of Glamorgan, Pontypridd, CF37 1DL, Wales, UK

S.M. ALLEN and S. HURLEY {Stuart.M.Allen, S.Hurley}@cs.cf.ac.uk
Dept. of Computer Science, University of Wales, Cardiff, PO Box 916, Cardiff CF2 3XF, Wales, UK

Abstract. Most writers on frequency assignment algorithms have described the details of a single algorithm, and evaluated the algorithm on selected data sets. There has been relatively little emphasis on describing the common features that are important if an algorithm is to have good performance. This paper describes the key features, with particular emphasis on algorithms for weighted fixed spectrum problems. The use of algorithms handling weighted constraints has become increasingly common in recent years. The advantages and disadvantages of weighting constraints are demonstrated.

Keywords: radio frequency assignment, algorithms, weighted constraints

1. Types of frequency assignment problem

Many authors have proposed algorithms for *the radio frequency assignment problem*. In fact there is not a single problem, but a variety of problems which can be addressed. For most networks the quality of service is limited by the level of interference within the network. As the modelling of this interference becomes more accurate, the formulation of the problem becomes more complex. Generally, the problems to be solved are of two types:

1. Given a specified set of transmitters T and a specified set of frequencies F, find an assignment $f : T \rightarrow F$ which minimises some measure of interference. Such problems will be referred to here as *fixed spectrum problems*.

2. Given a set of constraints which restrict the permissible interference in some way, find an assignment $f : T \rightarrow F$ which minimises the amount of spectrum used according to some measure. If the measure of spectrum is the difference between the largest frequency used and the smallest frequency used, then the problem is usually referred to as a *minimum span problem*. Other such problems exist, such as minimum order problems, where the measure of spectrum used is the number of occupied channels

* Corresponding author.

A fixed spectrum algorithm can be used to solve minimum span problems as follows:

1. *Generate a zero-violation assignment, using a sequential (greedy) method for example. Assume this assignment has span* q *and label the frequencies* $1, 2, \ldots, q+1$.
2. *Assign a random frequency (or lowest cost frequency) in* $\{1, 2, \ldots, q\}$ *to all transmitters assigned frequency* $q+1$. *This will normally introduce constraint violations.*
3. *Perform a fixed spectrum algorithm to eliminate the constraint violations introduced in step* 1. *This may find a zero-violation assignment with span* $q-1$.
4. *If a zero-violation assignment is found in step* 1, *set* $q := q-1$ *and go to step* 1. *Otherwise output* q *as the best span obtained, and its assignment.*

Figure 1. Application of fixed spectrum algorithms to minimum span problems.

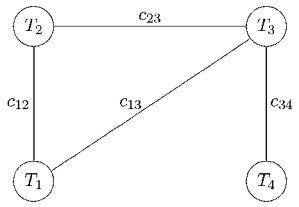
In general most operators have to solve the fixed spectrum problem. Minimum span problems may need to be solved by regulatory authorities and for planning purposes. Such algorithms also have a role in the evaluation of algorithms. Algorithms designed for fixed spectrum problems can be applied to minimum span problems, for which good lower bounds are generally available [1,7,17]. It can then be inferred that if an algorithm for the fixed spectrum problem performs well when applied to a minimum span problem as in figure 1, giving a span equal to or close to a lower bound, then it will perform well for fixed spectrum problems. This inference is not completely justified. It is conceivable that an algorithm that performs well in removing the small number of constraint violations usual when applying the procedure of figure 1, could be much less effective in minimising the large number of constraint violations that can occur in fixed spectrum problems. Thus this paper will focus on fixed spectrum problems. It may be required to minimise the number of constraint violations, but increasingly operators find it better to minimise the sum of certain weights associated with violated constraints [8]. Thus we will formulate the weighted form of the problem; a non-weighted form can be obtained by setting all weights to be 1.

Definition 1. Suppose that we are given a set $T = \{T_1, T_2, \ldots, T_N\}$ of transmitters, a set $\{c_{ij}\colon\ i < j,\ i, j \in \{1, 2, \ldots, N\}\}$ of constraint values bounded above by $c_{\max}$, a set $\{w_{ijk}\colon\ i < j,\ i, j \in \{1, 2, \ldots, N\},\ k \in \{1, 2, \ldots, c_{\max}\}\}$ of constraint weights and a set $F = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_N\}$ of frequency domains. It is required to find an assignment $f : T \to \bigcup_{i=1,2,\ldots,N} \mathcal{F}_i$ which satisfies $f(T_i) \in \mathcal{F}_i$ and which minimises:

$$\sum_{i,j,\ i<j} \sum_{k=1,2,\ldots,c_{\max}} w_{ijk} s_k\big(|f(T_i) - f(T_j)|\big), \tag{1}$$

where

$$s_k\big(|f(T_i) - f(T_j)|\big) = \begin{cases} 1 & \text{if } |f(T_i) - f(T_j)| = c_{ij} - k, \\ 0 & \text{otherwise.} \end{cases}$$



$$\mathcal{F}_1 = \mathcal{F}_2 = \mathcal{F}_3 = \{0, 1, 2\}, \quad \mathcal{F}_4 = \{0, 2\}$$
$$c_{12} = 1,\ c_{23} = 3,\ c_{13} = 1,\ c_{34} = 2,\ c_{\max} = 3$$
$$w_{121} = 0.1, \quad w_{231} = 0.2, \quad w_{23k} = 1.4\ (k = 2, 3)$$
$$w_{131} = 0.2, \quad w_{34k} = 100.0\ (k = 1, 2)$$

Figure 2. Example of a weighted frequency assignment problem.There is an assignment of cost 0.2.

Clearly only c_{ij} different weights w_{ijk} need to be defined for each (i, j), and these weights might be expected to increase with k. The use of different weights for different values of k allows a more accurate interference cost to be represented. An example of a small problem using this formulation is given in figure 2. Note that sometimes certain of the constraints can be chosen as either hard or soft. For example, a frequency may be treated as *illegal* for transmitter T_i, and so is not included in the domain $\mathcal{F}_i$. This is a *hard* constraint. Alternatively, a very high weight w_{if} may be defined and a term w_{if} is added to the cost function if frequency f is assigned to transmitter T_i. This is a *soft* constraint. Sometimes the problem is represented in cellular form, with T_i representing a transmitter site or sector, and a demand value d_i representing the number of frequencies required for T_i. This may have computational advantages, but may also have practical disadvantages. For example, it does not accurately model the higher weights required for a control carrier than for a traffic carrier.

If a weighted fixed spectrum algorithm is to be effective in minimising interference, the weights must be chose to accurately reflect the undesirability of the interference associated with an assignment. This will be considered in section 2.

This paper is not intended to include a comprehensive survey of the literature on frequency assignment algorithms, nor does it aim to demonstrate the superiority of meta-heuristic algorithms. These methods are the primary class referred to in the paper simply because of their publicized success in solving various aspects of the frequency assignment problem. The main purpose of the paper is to identify important features in the formulation, algorithmic detail and choice of cost function in algorithms which have been demonstrated by many authors to produce effective solution frameworks. It should be noted that the formulation and method of selecting weights given here are widely used in the mobile telephone industry. Also detailed in the paper is computational experience of the effect of using weighted cost functions against thresholded cost functions.

2. Weighted frequency assignment and modelling

Ideally, the cost function used to model a fixed spectrum frequency assignment problem will accurately reflect some measure of the service quality within the network, for exam-

ple, the average number of dropped calls. In most cases this measure will be based (either directly or indirectly) on the level of interference within the network. The construction of weights described here is typical of the procedure employed for GSM networks. However, it applies equally to any radio communications network with a predefined set of reception points at which a service is to be provided. At a reception point r, the interference from an interfering transmitter T_I on a wanted signal from a transmitter T_S only becomes unacceptable when the signal-to-interference ratio (SIR) is below some limit σ, either defined by equipment limitations or by standard definitions. If the SIR is below this service threshold then a dropped call is likely to occur. For example, in a GSM network, the requirement is that the SIR be above a value σ, which is typically chosen between 9 dB and 14 dB. Similar values apply in other radio networks. Most networks are planned and managed using a propagation prediction tool. These calculate the received signal from each transmitter at any point in the network. Constraints for frequency assignment are typically derived by considering propagation predictions for the serving and interfering signals at each reception point r. Let S_r^S denote the predicted signal strength received by a mobile at r from the transmitter T_S of the serving sector and let S_r^I denote the predicted signal strength from the cochannel transmitter T_I of the interfering sector. The predicted SIR at r is then:

$$C_r = \frac{S_r^S}{S_r^I}.$$

Whenever C_r is below the service threshold σ, a cochannel constraint $|f(T_S) - f(T_I)| \geqslant 1$ can be added. The weight of the constraint could simply be set to the number of points r whose predicted SIR is below the threshold if the constraint is violated. However, since the propagation predictions are subject to an error, it is better to consider the probability of an unacceptable SIR at r given the predicted value. A cumulative normal distribution is typically assumed as in figure 3. The exact nature of the distribution will reflect confidence in the predicted values. Summing the probability of unacceptable interference from T_I across all reception points served by T_S gives a weight

$$w'_{T_S T_I c_{T_S T_I}} = \sum_{r \in \{\text{sector } T_S\}} \text{prob}(C_r < \sigma)$$

representing the expected fraction of the area of the sector served by T_S suffering unacceptable cochannel interference as a result of interference from T_I. This process can be repeated to generate weights for adjacent channel constraints by including an appropriate attenuation factor in the interfering signal strength when calculating the SIR at each reception point. Thus S_r^I would be replaced by θS_r^I where typically $\theta = 1/64$, and $w'_{T_S T_I (c_{T_S T_I} - 1)}$ would be obtained. Note that this technique leads to asymmetric weights in general, however, they can be added according to the formula $w_{ijk} = w'_{ijk} + w'_{jik}$ $(i < j)$ to give symmetric constraints. Higher separation constraints such as co-sector separation constraints are usually given an arbitrary high penalty, or treated as *hard*. It should also be noted that areas of unacceptable interference are multiply counted according to

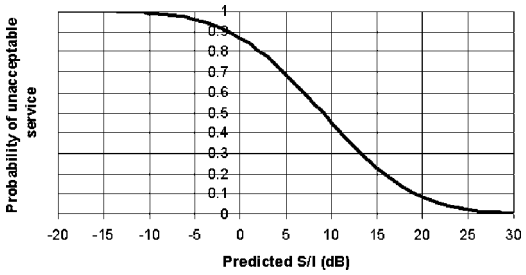


Figure 3. The probability of unacceptable SIR.

the actual number of interfering transmitters, so the measure of interference is only an approximation to the true level of interference in the sector.

3. Evaluation of performance of existing algorithms

Any attempt to evaluate the relative performance of existing algorithms is certain to prove difficult, and this difficulty tends to increase rather than decrease as more papers are published. This arises partly because algorithms are often tested on unsuitable data sets. These are frequently not widely available, sometimes rather easy for any reasonable algorithm and in many cases are not typical of the problems that arise in real systems. Many authors have good practical data available, but are not able to share it for reasons of commercial confidentiality.

It is also necessary to consider what is meant by a "good" algorithm. In some applications, such as cell planning, it may be necessary to accept a very fast algorithm, which obtains solutions of only moderate quality. In other applications run times of many hours are acceptable, but the highest quality assignment possible is sought. In this paper we shall give precedence to solution quality over speed when assessing algorithms.

There are a number of sets of benchmark problems available. The CELAR problems [4] have been very influential in inspiring research on frequency assignment algorithms. From the current perspective they have three problems. Firstly, they contain certain "equality constraints" (the frequencies assigned to specified pairs of transmitters differ by exactly a fixed number of channels). Many algorithms designed for the formulation given in definition 1 do not address these constraints. Secondly, some authors have used problem specific structure to obtain their solutions. This can make it unclear how well particular algorithms will perform on other types of data. Thirdly, most instances have now been solved to optimality. Another set are the "Philadelphia" problems. The main variations of these minimum span problems were solved to optimality in 1996 [10,18]. Some authors have continued to invent new variations, or to attempt to obtain these optimal solutions more quickly. However, it should be recognised that these problems first appeared as long ago as 1973 [2], and they cannot be considered to be typical of modern systems. It should also be noted that some authors have used variations that

are really very easy. For example, if the minimum number of channels separation between two co-sited transmitters is increased to 7, the problem is not a challenging test of any algorithm. There are some indications that a number of groups are beginning to address the question of making a variety of types of practical and challenging data widely available (see, for example, [6]). Benchmark problems that address the real issue of minimising interference, rather than asking for minimum span for a given set of constraints, are beginning to circulate.

It is not possible to include all of the many different types of algorithm that have been proposed in a single framework. In this paper several types of algorithm will not be considered. The first such type are exact algorithms. Although it has recently been demonstrated that, with certain formulations, exact algorithms may be practical for surprisingly large problems, they must always suffer from a limitation on the size of problem that can be handled. Secondly, hill-climbing and sequential (greedy) algorithms are often outperformed by meta-heuristic algorithms [10] in terms of solution quality, but nonetheless produce reasonably good solutions quickly. In the past many authors have attempted to refine the orderings used in sequential algorithms, or attempted to develop schemes for updating the orderings in the light of the algorithm's previous performance.

Algorithms will not be considered here if they only address minimum span problems and are unsuitable for fixed spectrum problems. It may be possible to use them iteratively with weakened constraints. Thus if no zero violation solution is found within the available number of channels, weighted constraints can be removed according to some thresholding scheme (typically if $w_{ijk} < \psi$ for some fixed constant ψ then w_{ijk} is replaced by 0), and the algorithm is repeated. However, we shall see later in this paper that such thresholding can compare unfavourably with the use of the formulation in definition 1 with appropriate weights.

Frequency assignment algorithms have been proposed based on neural networks, but the authors are not aware of any that appear to improve the performance of widely publicised meta-heuristic algorithms on common benchmark problems. Other algorithms do not seem to fit in to any general classification or have not demonstrated good performance on a wide range of problem instances.

For these reasons and as a result of the experience of the authors, we consider meta-heuristic algorithms, in particular tabu search (TS), simulated annealing (SA), genetic algorithms, and multi-agent algorithms. However, the main emphasis will be on simulated annealing and, in particular, tabu search. The results and conclusions presented are indicated by a variety of weighted and unweighted fixed spectrum problems, as well as minimum span problems. They are based on the experience of the authors in developing a number of systems. The most fully documented system is FASOFT [10,20], a package containing many different algorithms and variations of algorithms. Other algorithms developed include two which address multiple interference problems (see, for example, [15,16,21]), a system which can handle intermodulation product and spurious emission and response constraints [19], and an algorithm which addresses the formulation given in definition 1. The algorithms have been extensively tested on many different

types of data sets; military radio links problems, problems of Philadelphia type, area coverage problems with randomly generated transmitter placement and real data from several mobile telephone operators, mostly GSM data.

3.1. Solution search and data structures

The algorithms under discussion all search a space of potential assignments for an assignment which minimises some cost function (such as expression 1 in definition 1). However well controlled the search, it is important that its implementation be fast in order to search as much of the space as possible. Apart from efficient data structures, it is particularly important that fast evaluation of the cost function is achieved. This makes it necessary to update an existing cost by considering only the changes that affect the cost, rather than re-evaluating the full cost function at each iteration. This was found to be especially important in [19] for problems with many intermodulation product constraints. Several implementations of algorithms known to the authors use some form of cost change table. A table of all possible moves and their cost implications is stored. It is then easy to evaluate a neighbourhood of solutions, but after a move is accepted the cost change table must be updated. A description of this technique for tabu search can be found in [5,9]. We carried out some experiments with and without such a cost table for a tabu search algorithm addressing a formulation based on that in definition 1. For some real GSM data with several hundred cells an increase in speed by a factor of about 16 was noted. Let N_{size} denote the mean number of possible moves to neighbouring solutions of the current solution during the search. Also let U_{size} denote the mean number of entries of the table that need updating when a move is made. For any particular algorithm, the actual speed increase will depend on $N_{\text{size}}/U_{\text{size}}$. When a cost table was considered for the algorithm addressing intermodulation product and spurious emission constraints described in [19], the complexity of updating the cost table appeared prohibitive.

3.2. Genetic algorithms

Genetic algorithms have been implemented in a number of forms. A genetic algorithm works with a population of solutions. At each step, existing solutions are replaced by new, hopefully improved solutions. This is done by *mutation* and *crossover* operators. A mutation operator is applied only to single members of the population. It can be used infrequently to introduce diversity. For example, if a situation evolves where all members of a population use the same frequency for a particular transmitter, the mutation operator makes it possible to introduce a new frequency for the transmitter [10]. Alternatively, a hill climbing mutation operator can be used repeatedly on solutions in the population to reduce their cost [13]. Each transmitter in the solution is changed to a frequency which gives the solution a lower cost until no further cost reductions can be achieved in this way.

A straightforward form of crossover operator for frequency assignment problems was presented in [10]. Let $(f_1, f_2, \ldots, f_N)$ and $(g_1, g_2, \ldots, g_N)$ be two solu-

tions from a population; in each case the vector represents the frequencies assigned to $(T_1, T_2, \ldots, T_N)$. One-point crossover produces two new solutions, $(f_1, f_2, \ldots, f_c, g_{c+1}, \ldots, g_N)$ and $(g_1, g_2, \ldots, g_c, f_{c+1}, \ldots, f_N)$. This type of crossover and some variations were tested for minimum span problems during the development of FASOFT [10] and were found to be relatively ineffective compared with SA and TS meta-heuristics. Another approach is to treat crossover of two solutions as a frequency assignment problem where the frequency domains $\mathcal{F}_i$ each have cardinality at most two, with the available frequencies being the frequencies assigned to transmitter T_i in the two parent solutions. Again, this was tried for minimum span problems during the development of FASOFT using either hill climbing, simulated annealing or tabu search to solve each crossover frequency assignment problem. The results were better than those with one-point crossover, but still not fully competitive with SA and TS meta-heuristics. More recently Kolen [13] has used the same type of crossover, with the crossover frequency assignment problem solved by a (0,1)-programming formulation. Excellent results are obtained for CELAR problems, but no results are given for other types of data set. The results of applying crossover and mutation operators to a population of solutions results in a *hyper-neighbourhood*. We are not aware of any general study of the relative merits of different types of hyper-neighbourhood in the solution of frequency assignment problems by genetic algorithms.

Another approach to using genetic algorithms in minimum span frequency was described in [3] and in [11]. In these two similar approaches, the mutation and crossover operators were applied to a transmitter ordering which is used in a sequential algorithm. Thus a mutation or crossover operator gives a new ordering and this ordering is used as the initial transmitter ordering in the sequential algorithm. The transmitter selected to assign next is always the next transmitter in this ordering and frequencies are tested in increasing order. After the sequential ordering is applied the fitness (e.g., span) of the assignment can be evaluated and associated with the new transmitter ordering. Good results for certain Philadelphia instances were reported in both cases. The application of this type of genetic algorithm to fixed spectrum problems is less certain. In [3] Beckmann and Killatt proposed the use of a thresholding scheme if the available spectrum is limited. Constraints with their measure of importance below some threshold are ignored. Again note that we shall see later in this paper that such thresholding can compare unfavourably with the use of the formulation in definition 1. An attempt was made to modify the algorithm described in [11] for the fixed spectrum problem described in [19]. The difficulty encountered concerned the choice of sequential algorithm to use for fixed spectrum problems, which is by no means unique. If no zero cost assignment can be found for a transmitter, it appears sensible to assign to the transmitter the frequency that gives the lowest cost. The consequence of this was that most transmitters had to be tested with all frequencies during the sequential assignment. Thus the relative slowness of this sequential algorithm made this type of genetic algorithm uncompetitive with SA or TS meta-heuristics for such fixed spectrum problems.

3.3. Simulated annealing

A detailed account of simulated annealing applied to frequency assignment problems can be found in [10]. Given an existing assignment, of cost C, a change to the frequency assigned to a single transmitter T_i is identified. Let ΔC denote the change in cost associated with this change in frequency. Then the change to the new frequency is accepted if $\Delta C \leqslant 0$. In order to allow escape from local minima, it may also be accepted, with probability $e^{-(\Delta C/kt)}$, if $\Delta C > 0$. Here t is a variable referred to as temperature, which reduces as the run proceeds, and k is a constant which can be fixed as 1 as it essentially only scales the temperature. The probability is implemented by generating a pseudo-random number ρ between 0 and 1 and testing whether $\rho < e^{-(\Delta C/kt)}$. The full neighbourhood, where any transmitter T_i is selected and given a new frequency chosen from the domain $\mathcal{F}_i$, is called a single move neighbourhood.

It is also necessary to define a cooling schedule which describes how t reduces as the algorithm proceeds. A geometric cooling scheme can be used, where t reduces at each step in temperature according to the formula $t_{i+1} = \zeta t_i$, and the number of iterations at each temperature step increases according to the formula $N_{i+1} = N_i/\zeta$. Typical values for the parameters are $t_{\text{start}} = 1.0$, $\zeta = 0.95$ and $N_{\text{start}} = N$, the total number of transmitters. A finishing temperature t_{stop} can also be defined, typically with $t_{\text{stop}} = 0.0001$. However, for practical use the algorithm can be stopped whenever an assignment is required by the operator, and the best assignment found so far is used. Thus the definition of t_{stop} may be irrelevant. Other cooling schedules are described in [10].

3.4. Multi-agent algorithms

Multi-agent algorithms for frequency assignment have been attempted recently by several authors, see for example [14]. Although an exact evaluation of their relative performance with other meta-heuristics is not completely clear, it seems reasonable to claim that their performance is broadly comparable with other meta-heuristic algorithms. It has been suggested that it may be easier to tune the parameters of multi-agent algorithms than, for example, SA meta-heuristics. It is not clear that this claim can be substantiated. No study has been presented of the importance of neighbourhood choices in multi-agent algorithms; indeed the nature and definition of a neighbourhood can vary considerably between different algorithms. Consequently multi-agent algorithms will not be considered further here.

3.5. Tabu search

A detailed account of tabu search applied to frequency assignment problems can be found in [10]. In tabu search a system of short-term (and sometimes long-term) memory is used in order to guide the search out of local minima, to prevent cycling over the same set of moves and to diversify the assignments examined. At each iteration of tabu search, the algorithm holds the current assignment, a neighbourhood of possible moves,

the change in cost ΔC associated with each move, and definitions of which moves are forbidden. The neighbourhood may consist of all possible moves, or a random selection of moves. A move may be forbidden, or *tabu* because of a short-term memory condition. This specifies that where a transmitter has been moved to a frequency previously, it cannot be moved to that frequency again for the next r moves. A *recency list* holds the previous r moves and where a move is proposed, it is checked against this list to determine whether the move is tabu. The second source of tabu moves that can be used is the long-term memory. This represents the condition that a transmitter should not have its frequency changed too many times. The algorithm maintains a long-term memory list which has as many entries as there are transmitters in the problem. When a given transmitter is moved, the appropriate entry in this memory list is incremented. The value of this entry is then divided by the number of iterations to produce a long-term memory fraction. Where this fraction is larger than some pre-defined value β, then the move is marked as tabu. It will be seen later that long term memory can be counter-productive and is best omitted for certain types of neighbourhood.

Thus, the algorithm takes the set of candidate moves in the neighbourhood and marks those that are tabu. The move with the smallest value of C that is non-tabu and the move with the smallest value of C that is tabu are selected. If the best tabu move is better than the best non-tabu move, and gives a new best assignment, then the tabu move is taken. This condition allowing a tabu move to be taken is known as the *aspiration criterion*. Otherwise, the best non-tabu move is always taken. When a move is taken, it is added to the recency list, and the appropriate long term memory counter is updated.

The two memory conditions produce a requirement for two parameters for the algorithm, defining the length of the recency list and the long term memory fraction respectively. The recency parameter is usually defined as a fraction of the total number of transmitters in the problem:

$$r = \alpha |T|. \tag{2}$$

Also, if we introduce a parameter λ, where $0 \leqslant \lambda \leqslant 1$, we can define:

$$\beta = \frac{\lambda}{r} + \frac{1 - \lambda}{|T|}. \tag{3}$$

Thus, the two parameters for the control of the tabu search are α and λ. These two parameters are referred to as the *recency fraction* and the *lambda parameter*. They control parameters that the user can adjust. Typical values are 0.4 and 0.5, respectively.

The search may terminate when a zero cost solution is found. Alternatively, if the cost of the best assignment found is unchanged over a given number of iterations (known as the *frozen iteration number*), the algorithm terminates.

A further parameter associated with tabu search is the neighbourhood size fraction η, which, when multiplied by $|T|$, gives the number of assignments in a randomly selected neighbourhood. When no cost change table is used this parameter is crucial in determining good assignments quickly. A typical value of η is 0.25, but in certain circumstances, for example when using tabu search to improve an already good assign-

ment, it may be useful to increase η to 1 or more. When a cost change table is used a full neighbourhood may be the best choice.

A tabu search with a full neighbourhood or a random neighbourhood generally has similar performance to simulated annealing. However, it will be shown in section 4.2 that with a carefully chosen neighbourhood, the performance of tabu search can be improved further.

4. Parameters and neighbourhoods for meta-heuristic algorithms

4.1. Simulated annealing parameters and cooling

Assume for the moment that simulated annealing is implemented with transmitter T_i randomly chosen from the set T of transmitters, and the new frequency for T_i is chosen randomly. Then the algorithm is essentially controlled by the starting temperature t_{start}, the nature of the cooling schedule and the parameters that control the cooling schedule. Experience with a wide variety of problems suggests that

(i) the cooling schedule described in section 3.3 is satisfactory if the parameters T_{start} and ζ are chosen satisfactorily;

(ii) other schedules appear to give at best only a marginal improvement in performance.

However, if simulated annealing is used to improve a given assignment, such as an assignment obtained by a sequential algorithm, t_{start} must be chosen to be much smaller than if a random starting assignment is used. Otherwise the characteristics of the good starting assignment are immediately lost. It should also be noted that it is often best to use several or many runs of SA from a variety of good starting assignments, rather than one very long run.

4.2. Violating neighbourhoods

Definition 2. A transmitter is a *violating transmitter* in a current assignment if it is involved in a constraint which is currently not satisfied.

Definition 3. The set of possible moves from the current assignment is referred to as the *neighbourhood* of the assignment. In many algorithms these moves consist of a change of frequency for a single transmitter. If the neighbourhood consists only of changes of frequency for violating transmitters, then it is referred to as a *violating neighbourhood*. Whether the neighbourhood is violating or not, it is possible to use either the full neighbourhood or a *random neighbourhood*, consisting of a random selection of moves from the full neighbourhood.

Good algorithms allow the implementation to quickly calculate the cost changes associated with all moves in the neighbourhood. Single changes of frequency allow this.

The results of tables 1 and 2, generated using the software described in [19] are typical of results indicating that in some circumstances the use of a violating neighbourhood is vital for tabu search. The advantages for simulated annealing are not as clear but still noticeable in some circumstances. The results were taken on a 333 MHz, 64 MB

Table 1
Simulated annealing runs of at most 1 minute.

Number of transmitters	SA with random start cost	SA with random start time (sec)	SA with sequential start cost	SA with sequential start time (sec)
Random neighbourhood, not violating				
100	0	30	0	15
300	225	60	40	60
600	2415	60	240	60
Random neighbourhood, violating				
100	20	60	10	60
300	170	60	10	60
600	1993	60	145	60

Table 2
Tabu search runs of at most 1 minute.

Number of transmitters	TS with random start cost	TS with random start time (sec)	TS with sequential start cost	TS with sequential start time (sec)
Random neighbourhood, not violating, no long term memory				
100	0	30	16	60
300	1442	60	97	60
600	57557	60	275	60
Random neighbourhood, violating, no long term memory				
100	0	22	0	11
300	348	60	20	60
600	42395	60	180	60
Random neighbourhood, not violating, long term memory				
100	34	60	30	60
300	472	60	165	60
600	40311	60	361	60
Random neighbourhood, violating, long term memory				
100	0	13	0	33
300	282	60	125	60
600	4165	60	305	60

Pentium PC and were implemented in Microsoft Visual C++ running under Windows 95. Long term memory in TS appears to be counter-productive. The use of a violating neighbourhood makes the algorithm concentrate on a relatively small number of hard to assign transmitters, whereas long term memory tends to prevent such concentration. The use of sequential starting assignments can improve the speed of the algorithms and the quality of the assignments obtained. The results presented are for a single run of at most 1 minute. In fact all of the problems described in tables 1 and 2 do have zero cost solutions. These are generally found most quickly by TS with a sequential start, a violating neighbourhood and no long term memory.

4.3. Short term memory for tabu search

It should be noted that it is possible to define recency in terms of *moves* as in section 3.5 or in terms of *transmitters*. Thus in the second case a transmitter is tabu if it has been involved in an accepted move in the last r' iterations. It is possible to use either or both types of recency list. Clearly if both are used it is necessary that $r > r'$ if the move recency is to have any effect. An experiment with a single class of data indicated that there is merit in using both types of recency list together, although this result may well be data dependent.

5. Neighbourhoods for weighted frequency assignment

Once it is noted that it is the use of a violating neighbourhood that particularly enhances the performance of tabu search, in comparison with other meta-heuristics, a study of the formulation in definition 1, taken together with the method of generating the weights w_{ijk} in section 2, indicates a potential problem. In weighted frequency assignment problems there may well be a large number of low weighted constraint violations in an optimal solution. Users may well retain constraints with a low weight which would be omitted in unweighted problems. This can mean that the violating neighbourhood becomes much too large to be effective, and performance reverts to the same level as when no violating neighbourhood is used.

The problem can be overcome by restricting the neighbourhood, utilising the weight of the violated constraints. Transmitters can be added to the neighbourhood whenever they are involved in violated constraints with weight above some threshold (either as a total for the transmitter or for individual constraints). Thus a full violating neighbourhood of transmitters with threshold ε would be

$$\left\{T_i \mid \exists j \in 1, \ldots, N,\ k > 0 \text{ with } w_{ijk} > \varepsilon \text{ and } \left|f(T_i) - f(T_j)\right| = c_{ij} - k\right\}.$$

The algorithm is applied in a number of stages, with a sequence $\varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots$ of progressively smaller thresholds. This ensures that the algorithm is continuously considering a relatively small neighbourhood of influential moves.

The potential disadvantage of such a method is that small violations that may be easy to eliminate may not be considered in the neighbourhood. More sophisticated

neighbourhood constructions can lead to improved performance, in particular when the structure of the neighbourhood is modified during the algorithm. However, any such method is dependent on the distribution of weights within a specific problem instance or class of problems. Experience suggests that such methods can be made to work very effectively. Although it is difficult to carry out fair comparisons of all the increasing number of published algorithms, our experience suggests that with these improvements, tabu search is probably at least as good as the best algorithms currently known for the problem formulated in definition 1.

Table 3
Results for 5 runs with thresholded constraints for 8 different thresholds and cost function equal to the number of violations.

	Cost = no. violations of constraint with weights above a defined threshold					
	Threshold = 0			Threshold = 50		
	Cost	Total weight	True total weight	Cost	Total weight	True total weight
run 1	607	36100000.00	36100000.00	23	1630000.00	1630000.00
run 2	582	37900000.00	37900000.00	21	1220000.00	1220000.00
run 3	595	35600000.00	35600000.00	23	1420000.00	1420000.00
run 4	597	35000000.00	35000000.00	27	1080000.00	1080000.00
run 5	608	38000000.00	38000000.00	24	1630000.00	1630000.00
	Threshold = 100			Threshold = 140		
	Cost	Total weight	True total weight	Cost	Total weight	True total weight
run 1	7	607742.00	629062.00	2	2622.41	33075.80
run 2	7	405417.00	428270.00	3	200405.00	228357.00
run 3	9	414413.00	436609.00	4	13956.60	42963.20
run 4	5	201946.00	224104.00	4	4636.47	33891.70
run 5	7	226681.00	249335.00	1	851.61	27903.70
	Threshold = 160			Threshold = 180		
	Cost	Total weight	True total weight	Cost	Total weight	True total weight
run 1	1	7742.46	40354.40	0	0.00	34247.60
run 2	1	1679.35	32356.60	1	2483.44	35231.20
run 3	0	0.00	31316.40	1	446.46	32658.10
run 4	2	200000.00	231426.00	0	0.00	35154.80
run 5	2	200000.00	229764.00	1	1050.88	34910.40
	Threshold = 200			Threshold = 400		
	Cost	Total weight	True total weight	Cost	Total weight	True total weight
run 1	1	2866.36	36881.50	0	0.00	50222.60
run 2	0	0.00	32337.40	0	0.00	49190.60
run 3	0	0.00	31559.40	0	0.00	49627.20
run 4	1	305.80	35755.70	0	0.00	50660.20
run 5	0	0.00	36942.70	0	0.00	49365.40

6. A comparison of weighted cost with thresholded cost

It has already been noted that the use of a weighted cost function has achieved increasing popularity with users. This cost function is used instead of simply minimising the number of violations or thresholding the constraints so a zero violation solution can be achieved. The results in tables 3 and 4 clearly illustrate the advantages of the weighted cost function. The weights were determined using methods similar to those described in section 2. Thus the numbers in the columns headed "true total weight" can be taken to be a reasonable measure of the interference in the network, whereas the number of constraint violations is certainly not. An effective tabu search algorithm suitable for solving the formulation in definition 1, using the enhancements described in section 5 and a cost change table, was used. The cost function used in table 3 is the sum of the weights of violated constraints that are above the threshold. It can be seen in table 3 that when minimising the number of constraint violations the best threshold to use here is about 140, giving a true total weight of 27,903.70 for the best run. A threshold of at least 160 is needed if a zero violation solution is to be obtained, but the true total weight of the solution is then much higher. When, as in table 4, the weighted sum of violations is minimised, a threshold of 0 should be used and a best true total weight of 18,968.00 is achieved. The superiority of the unthresholded, weighted approach (as in definition 1) is also supported by the distribution of costs over 5 runs.

Table 4
Results for 5 runs with thresholded constraints for 3 different thresholds and cost function equal to the total weight of violated constraints with weights above a defined threshold.

	Cost = total weight of violated constraints with weights above a defined threshold					
	Threshold = 0			Threshold = 140		
	Cost	No. violations	True total weight	Cost	No. violations	True total weight
run 1	18968.00	1842	18968.00	1841.84	10	29520.90
run 2	21044.40	1817	21044.40	1277.73	7	29443.70
run 3	19203.90	1835	19203.90	1873.81	11	31703.50
run 4	20291.80	1815	20291.80	1468.28	7	27079.80
run 5	20704.10	1806	20704.10	2538.14	12	29539.20
	Threshold = 200					
	Cost	No. violations	True total weight			
run 1	0.00	0	34890.00			
run 2	247.89	1	33809.90			
run 3	259.15	1	35603.20			
run 4	0.00	0	35898.70			
run 5	468.60	2	35996.80			

7. Conclusion

Meta-heuristic algorithms are now recognized to be effective in the solution of frequency assignment problems. This paper has concentrated particularly on algorithms for weighted fixed spectrum problems. The advantages of weighting these constraints without thresholding have been demonstrated. The need for a rational method of generating the weights is a potential disadvantage. The method described in section 2 leads to a cost function which is a reasonably good representation of the amount of interference in the network. However, ultimately it may be advantageous to minimise the amount of interference in the network more directly; an idea first presented in [12].

The claim has been made that meta-heuristic algorithms in general and tabu search (incorporating the enhancements described in this paper) in particular, provides an effective method of solving frequency assignment problems. There is ample evidence to support this statement in [5,9,10,18–20] and elsewhere. These algorithms also have the advantage that a complete frequency assignment is always available from a very early stage in the run. However, an exact comparison of the best currently available algorithms must await the wider availability of a range of modern, practical, challenging datasets. A start on the process of providing such datasets has been made in [6], but a wider variety of data is necessary.

References

[1] S.M. Allen, D.H. Smith and S. Hurley, Lower bounding techniques for frequency assignment, Discrete Math. 197/198 (1999) 41–52.
[2] L.G. Anderson, A simulation study of some dynamic channel assignment algorithms in a high capacity mobile telecommunications system, IEEE Transactions on Communications 21 (1973) 1294–1301.
[3] D. Beckmann and U. Killat, A new strategy for the application of genetic algorithms to the channel-assignment problem, IEEE Trans. Vehicular Technology 48 (1999) 1261–1269.
[4] CELAR Instances, ftp://ftp.win.tue.nl/pub/techreports/CALMA/Instances/ (see also http://fap.zib.de/).
[5] R. Dorne and J.-K. Hao, Tabu search for graph coloring, T-colorings and set T-colorings, in: *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, eds. A. Voss, S. Martello, I.H. Osman and C. Roucairol (Kluwer Academic, 1999) pp. 77–92.
[6] FAP web: http://fap.zib.de/
[7] A. Gamst, Some lower bounds for a class of frequency assignment problems, IEEE Trans. Vehicular Technology 35 (1986) 8–14.
[8] P. Hansson, C. Lewer-Allen and J. Samuelsson, Breakthroughs in frequency planning, Mobile Communications International (December 97/January 98) 89–90.
[9] J.-K. Hao, R. Dorne and P. Galinier, Tabu search for frequency assignment in mobile radio networks, J. Heuristics 4 (1998) 47–62.
[10] S. Hurley, D.H. Smith and S.U. Thiel, FASoft: A system for discrete channel frequency assignment, Radio Sci. 32(5) (1997) 1921–1939.
[11] S. Hurley, D.H. Smith and C. Valenzuela, A permutation based genetic algorithm for minimum span frequency assignment, in: *Proceedings 5th International Conference on Problem Solving from Nature*, Lecture Notes in Computer Science, Vol. 1498, eds. A.E. Eiben, T. Bäck, M. Schoenauer and H.-P. Schwefel (Springer, 1998) pp. 907–916.

[12] S. Hurley, R.M. Whitaker and D.H. Smith, Channel assignment in cellular networks without channel separation constraints, in: *Proc. IEEE Vehicular Technology Conference* (Fall, 2000) pp. 1714–1718.
[13] A.W.J. Kolen, A genetic algorithm for frequency assignment, Technical report, University of Maastricht (1999). Available at http://www.unimaas.nl/~akolen.
[14] V. Maniezzo, A. Carbonaro and R. Montemanni, An approach to frequency assignment based on an ANTS heuristic, in: *Proceedings of the 3rd Metaheuristics International Conference, MIC '99*, Rio de Janeiro, Brazil (1999) pp. 311–316.
[15] D.H. Smith, S.M. Allen, S. Hurley and W.J. Watkins, Frequency assignment problems with multiple interference (submitted).
[16] D.H. Smith, S.M. Allen, S. Hurley and W.J. Watkins, Frequency assignment: methods and algorithms, in: *Proceedings of the NATO RTA SET/ISET Symposium on Frequency Assignment, Sharing and Conservation in Systems (Aerospace)*, Aalborg, Denmark, October 1998, NATO RTO-MP-13 (1999) pp. K-1–K-18.
[17] D.H. Smith and S. Hurley, Bounds for the frequency assignment problem, Discrete Math. 167/168 (1997) 571–582.
[18] D.H. Smith, S. Hurley and S.U. Thiel, Improving heuristics for the frequency assignment problem, European J. Operational Research 107(1) (1998) 76–86.
[19] D.H. Smith, R.K. Taplin and S. Hurley, Frequency assignment with complex co-site constraints, IEEE Transactions on Electromagnetic Compatibility 43(2) (2001) 210–218.
[20] S.U. Thiel, S. Hurley and D.H. Smith, Frequency assignment algorithms, Technical report, University of Cardiff/University of Glamorgan (1997).
[21] W.J. Watkins, S. Hurley and D.H. Smith, Evaluation of models for area coverage, Research report 98003, University of Cardiff (1998).