A Study of Permutation Operators for Minimum Span Frequency Assignment Using an Order Based Representation

Christine L. Valenzuela (Mumford)
School of Computer Science,
Cardiff University, CF24
3AA, United Kingdom
Email: C.L.Mumford@cs.cardiff.ac.uk

Published January 2001

Abstract

The genetic algorithm (GA) described in this paper breeds permutations of transmitters for minimum span frequency assignment. The approach hybridizes a GA with a greedy algorithm, and employs a technique called Generalized Saturation Degree to seed the initial population. Several permutation operators from the GA literature are compared, and results indicate that position based operators are more appropriate for this kind of problem than are order based operators. My offspring versus mid-parent correlation studies on crossovers show Pearson's correlation coefficient to be a reliable predictor of performance in most cases. Results presented herein represent improvements over previously published results.

Keywords Genetic algorithms, frequency assignment problem, minimum span, order based representation, permutation operators, greedy algorithms.

1 Introduction

The electromagnetic spectrum is a finite resource. In recent years the wider use of mobile communications has greatly increased demand for the spectrum and, as a result, there is now a growing interest in the development of techniques for using the spectrum efficiently. Allocation of the spectrum to companies and other users (for example the military) is normally the responsibility of national and/or local governments. Generally a band of frequencies is assigned to each organization, and it is then up to the individual organization to determine how best to use the range of frequencies which have been allocated to it.

If a number of transmitters and receivers are situated close together, interference will occur if the same or similar frequencies are used by two or more of the transmitters. Ideally it should be possible to allocate the frequencies to all the transmitters in such a way that no interference is suffered. In reality the huge demand on the spectrum usually makes this impossible and the primary objective is then to minimize the total interference.

The extent to which pairs of transmitters interfere with each other largely depends on the distance between them, the closer they are together the more they interfere with each other. Transmitters at the same site or within tens of metres away from each other generate *co-site interference.* When equipment is at a distance of several kilometres or more, *far-site interference* is produced.

Interference can be modeled using a two-dimensional constraint matrix showing, for each pair of transmitters, the minimum channel separation required to completely eliminate interference. (For the purpose of frequency allocation, the electromagnetic spectrum is usually modeled in terms of discrete channels situated at fixed points of the spectrum.)

Table 1: A typical constraint matrix

	1	2	3	4	5
1	n	n	1	c	3
2	n	n	3	0	0
3	1	3	n	2	0
4	c	0	2	n	n
5	3	0	0	n	n

Table 1 provides an illustration of a small constraint matrix for a 5 transmitter problem. The bold numbers heading the columns and rows represent the 5 transmitters. In the body of the Table n denotes that the transmitters identified by the row and column heading can be allocated on the same channel, c denotes required co-site separation such that:

$$|\mathrm{f}_i - \mathrm{f}_j| \quad \geq \quad c,$$

where f_i and f_j represent the frequency channels allocated to transmitters i and j respectively, and the integer values in the Table (i.e. 0, 1, 2 and 3) represent far-site interference, s, such that:

$$|f_i - f_j| \quad > \quad s.$$

The frequency assignment problem exists in three forms:

- Minimum Span Frequency Assignment,
- Minimum Order Frequency Assignment, and
- Fixed Spectrum Frequency Assignment.

The Minimum Span Frequency Assignment Problem (MSFAP) involves assigning channels to transmitters in such a way that interference is completely eliminated and the range of frequencies used is such that the *span* = largest channel - smallest channel, is minimized. The Minimum Order Frequency Assignment Problem (MOFAP) also requires that interference is completely eliminated. However, the objective here is to minimize the number of frequency channels used, i.e. the *order*. The Fixed Spectrum Frequency Assignment Problem (FSFAP) provides a fixed range of frequencies and the objective here is to assign channels within the given range in such a way that the total interference is minimized. Of the three forms the FSFAP is the most important commercially. However the MSFAP is the form most ideally suited to my hybrid approach and is therefore the form chosen for the present study. It is worth noting that it is possible to adapt good algorithms for MSFAP to produce good solutions for the FSFAP and the MOFAP. (The FSFAP will be addressed elsewhere).

To date most Genetic Algorithms (GAs) in the frequency assignment literature attack the problem directly by breeding lists of channels mapped in a 1-1 fashion to lists of transmitters [5] [11] [15] [4]. The present paper hybridizes the GA with a simple greedy algorithm and the GA itself focuses on breeding permutations of transmitters. The greedy algorithm works as a *decoder*, transforming the permutation list of transmitters into a channel assignment for each of the transmitters. It does this by stepping through the list of transmitters output by the GA allocating frequency channels to each of the transmitters in turn in a simple deterministic way. One very important advantage of the permutation approach over the direct approach is that the application of the greedy decoder to any permutation guarantees a legal channel allocation (i.e. one where there is no interference). On the other hand the direct approach produces illegal lists of channel allocations with great ease.

Use of a permutation based (i.e. an *order based*) representation with a decoder generally cuts down the number of possible solutions that the genetic algorithm will consider. In the case of the MSFAP illegal solutions are eliminated, as explained above. Regarding the optimum solution, however, the following question arises: "Is it possible to encode the optimum solution to the MSFAP using an order based representation?" The answer to this question is "yes", as will be proved in section 4.

The work described herein extends the work documented in previous papers in two ways. Firstly it expands the comparative study of the various permutation operators started in [18]. Secondly it presents excellent results for a larger set of problems than [19].

An interesting study by T. Clark and G. D. Smith [3] uses an order based approach to the FAP with a simulated annealing algorithm as the search engine. The greedy decoder consists of two sequential assignment algorithms, which are applied one after the other. Excellent results are reported for the FSFAP using this approach, indicating the versitility of an order based representation for frequency assignment.

2 Sequential Assignment Algorithms

Sequential assignment methods mimic the way that the MSFAP may be solved manually. The transmitters are considered one at a time, successively assigning allowable frequencies as we proceed, until either we have assigned all the transmitters or run out of frequencies. We may generate a series of assignment methods based on three components:

- initial ordering,
- choice of next transmitter, and
- assignment of frequency.

Initial ordering techniques usually order transmitters according to the number and extent of their constraints, so that the most heavily constrained transmitters appear earliest on the list. The simplest way to choose the next transmitter is sequentially, picking the next one on the list produced by the initial ordering. A more complicated method, which has proved more effective than sequential selection with the various initial ordering methods, is called *generalized saturation degree.* In this method the choice of the next transmitter is influenced by the constraints imposed by all those transmitters which have already been chosen. One could view the more complicated process as a mechanism for correcting those mistakes which have been made by the initial ordering technique.

The simplest assignment technique is to assign to the chosen transmitter *the smallest feasible channel*, i.e. the lowest numbered channel which can be assigned without violating any constraints. Variations upon this technique attempt to assign channels that have already been used in favor of those that have not been used as the sequence of transmitters is stepped through. A detailed description of sequential assignment methods can be found in [11].

In the present paper a genetic algorithm is used to search the state-space of initial orderings. The choice of the next transmitter is made sequentially, and the smallest feasible channel is assigned to each chosen transmitter. Initial populations of permutations are produced in two different ways:

- at random, and
- by seeding the population.

GAs incorporating a range of permutation operators for testing are tried on both random and seeded initial populations. Random permutations are produced using a pseudo-random number generator which effectively builds up each permutation list one transmitter at a time, selecting the next one for the list from a uniform distribution of those not yet chosen for the list. The seeded population is produced by first generating a population of random permutations, and then applying the generalized saturation degree (GSD) algorithm to each of the permutation lists, to produce a new set of permutation lists. Details of the GSD algorithm are given below.

2.1 Generalized Saturation Degree Algorithm

Let V be a set of transmitters and V_c be the transmitters of V already assigned frequencies. Frequency n is said to be denied to the unassigned transmitter v if there is a transmitter u in V_c assigned to frequency n such that transmitter u and v would interfere, i.e., assuming an edge exists between u and v in the constraint graph then there is insufficient frequency separation between them. If frequency n is denied to transmitter v, the *influence* of frequency n, denoted by I_{nv}, is the largest weight of any edge connecting v to a transmitter assigned to frequency n. The number

$$\sum I_{nv}$$

(where the sum is taken over all frequencies n denied to v) is called the *generalized saturation degree* of v. The technique for selecting the next transmitter is as follows: Select a transmitter with maximal generalized saturation degree (break ties by selecting the transmitter occurring earlier in the initial ordering).

3 The Genetic Algorithm

Algorithm 1 Procedure GA1

```
generate $N$ random permutations {N is the population size}
if appropriate apply the GSD algorithm to each structure
evaluate the span produced by each structure and store each one
store best-so-far
repeat
  for each member of the population do
    this individual becomes the first parent
    select a second parent using roulette wheel selection on ranks
    apply crossover to produce offspring
    apply mutation to offspring
    evaluate span produced by offspring
    if offspring better than weaker parent then
      it replaces it in population
    if offspring better than best-so-far then
      it replaces best-so-far
until stopping condition satisfied
print best-so-far
```

The simple genetic algorithm (GA) used here is based on that which has appeared in [17]. It is derived from the model of [10] and is an example of a 'steady state' GA (based on the classification of [16]). It uses the 'weaker parent replacement strategy' first described by [2]. The GA outlined in Algorithm 1 applies the genetic operators to permutations of transmitters. The fitness values are based on the spans produced when the simple sequential assignment algorithm is applied to each permutation list produced by the GA. In 1 the first parent is selected deterministically in sequence, but the second parent is selected in a roulette wheel fashion, the selection probabilities for each genotype being calculated using the following formula:

$$\text{selection probability} = (\text{Rank}) \;/\; \Sigma\ \text{Ranks}$$

where the genotypes are ranked according to the values of the spans that they have produced, with the worse ranked 1, the second worse 2 etc.. and the best ranked highest.

The GA breeds permutations of transmitters and channels are assigned to the resulting lists of transmitters using the smallest feasible channel technique. Experimental results are presented which compare results obtained by application of the GA to *unseeded* and *seeded* populations. (I refer the reader back to section 1 for a description of the smallest feasible channel assignment technique and for details of how the initial population is produced.)

4 Genetic Operators for Permutations

The standard mutation and crossover operators designed for bit-strings and parameter lists produce illegal offspring when applied to permutation lists. To combat this problem a number of special operators have been developed which produce legal permutation lists for offspring. Many of these operators were first developed and tested on the Traveling Salesman Problem (TSP), where the permutation represents a (circular) list of cities in a TSP tour. Some important features implicit in the TSP permutation lists are:

- adjacent cities in the TSP permutation list represent edges of the tour,
- absolute positions of cities in the list are irrelevant, and
- reversing the list does not change the tour.

For the MSFAP the positions of transmitters in the list determine the order in which they are allocated channels by the simple greedy algorithm. What is important in an ordering of transmitters that allows the greedy algorithm to produce a good channel allocation? Clearly the *absolute position* of a transmitter in the list is important. The closer a transmitter is to the front of the list, the earlier it will be allocated a channel. The identities of transmitters preceding a particular transmitter on the list are also important, since it is these which impose constraints upon which channel can be allocated to the current transmitter being processed. The features implicit in the permutation representation for the MSFAP are clearly very different from those previously identified for the TSP, for example:

- adjacent transmitters in the list *do not* represent edges in the solution
- absolute positions in the list *are relevant*, and
- reversing the list is *likely to change the solution considerably.*

We will now examine a number of genetic operators.

4.1 Crossover

Crossover involves combining elements from two parents into one or more children. Crossover operators can be classified in the following way:

- position based crossover,
- order based crossover, and
- edge based crossover.

Position based crossover maintains in the offspring *absolute positions* of transmitters occurring in the parents. Order based crossover maintains *relative positions* of transmitters and edge based crossover maintains *adjacency information*, so that transmitters which are adjacent in the parents' lists tend to be adjacent in the offspring list.

A very large number of permutation crossovers exist in the literature, many of which have been developed specifically for the TSP, which remains the most popular of the testbed problems involving permutations (for example see [9] [20] [13]). In the present study some non problem specific position based and order based crossovers are chosen for testing. Edge based crossovers are not considered here because transmitters which are adjacent to each other in a permutation list do not relate to edges in the resulting solution (i.e. channel allocation) in the way that adjacent cities in a TSP list map to edges in the resulting tour (see discussion above). Complicated variants of position and order based crossover which incorporate problem-specific knowledge relating to the TSP (or any scheduling problem not related to MSFAP) are also omitted. A recent review of permutation crossovers can be found in [21].

The crossovers chosen for study on the MSFAP are Cycle Crossover (CX), Partially Matched Crossover (PMX), Order Crossover (OX) [14] and Uniform Order Based Crossover (UOBX) [6]. CX and PMX are examples of position based crossovers, and OX and UOBX are order based. A brief description of each crossover used in this study is given below.

Cycle Crossover

The cycle crossover operator ensures that each position in the resulting offspring is occupied by a transmitter occupying the *same* position in one or other of the parents. As an example, suppose we have strings A and B below as our two parents:

A = 8 7 6 4 1 2 5 3

B = 2 5 1 7 3 8 4 6

We now start from the left and randomly select a transmitter from string A. Suppose we choose transmitter 6 from position 3, this is then copied to position 3 of the offspring we shall call A':

A'= - - 6 - - - - -

In order to ensure that each transmitter in the offspring occupies the same position as it does in either one or other parent, we now look in position 3 of string B and copy transmitter 1 from string A to the offspring:

A'= - - 6 - 1 - - -

Next we look in position 5 of string B and copy transmitter 3 from string A:

A'= - - 6 - 1 - - 3

Looking at position 8 in string B we find transmitter 6. This completes the cycle. We now fill the remaining positions in A' from string B thus:

A'= 2 5 6 7 1 8 4 3

B'= 8 7 1 4 3 2 5 6

The offspring B' is obtained by performing the complementary operations.

Partially Matched Crossover

Two strings are aligned and two crossing sites are picked uniformly at random along the strings. These two points define a *matching section* that is used to effect a cross using position by position exchange operations:

A = 8 7 6 4 1 2 5 3

B = 2 5 1 7 3 8 4 6

In each string 4 changes places with 7, 1 with 3 and 2 with 8 producing:

A'= 2 4 6 7 3 8 5 1

B' = 8 5 3 4 1 2 7 6

Order Crossover

Order crossover starts in the same way as PMX, selecting 2 crossing sites at random:

A = 8 7 6 4 1 2 5 3

B = 2 5 1 7 3 8 4 6

Order crossover uses a sliding motion to fill the holes left by transferring the mapped positions from one string into the other. For example transmitters 4, 1 and 2 will leave holes, marked by "H" in string B:

B' = H 5 H 7 3 8 H 6

These holes are then filled with a sliding motion that starts from the second crossing point:

B' = 7 3 8 H H H 6 5
The substring from string A is then inserted into string B. The final result of this cross and the complementary cross is:

A' = 4 1 2 7 3 8 5 6

B' = 7 3 8 4 1 2 6 5

Uniform Order based Crossover

This crossover maintains the *absolute* positions of the transmitters taken from one parent, and the *relative* positions of the transmitters taken from the other parent. It proceeds by first generating a bit string, S, which is the same length as the parents:

A = 8 7 6 4 1 2 5 3

B = 2 5 1 7 3 8 4 6

S = 0 1 1 0 0 1 0 1

Fill in some of the positions in child A' by copying them from parent A wherever the bit string contains a "1", and fill in the positions in child B' from parent B wherever the bit string contains a "0":

A' = - 7 6 - - 2 - 3

B' = 2 - - 7 3 - 4 -

Fill in the rest of the positions in child A' from the elements associated with a "0" in parent A. Permute these elements so that they appear in the same order they appear in parent B, and fill the gaps in child A' from this list of permutated elements. Carry out a similar process for child B':

A' = 5 7 6 1 8 2 4 3

B' = 2 8 6 7 3 1 4 5

4.2 Mutation

The main function of a mutation operator in a traditional GA is to maintain diversity. In addition mutation can work as a search operator in its own right, and it is thought to be particularly valuable in the latter generations of a genetic search when genetic diversity is much reduced and as a consequence the crossover operator is much less effective. We will consider three mutation operators for the MSFAP:

- position based mutation,
- order based mutation, and
- scramble mutation.

Position based mutation involves selecting two transmitters at random and placing the second transmitter immediately before the first in the list. For order based mutation two transmitters are selected, again at random, and their positions are interchanged. Scramble mutation [6], operates by selecting a sublist randomly and scrambling the order of the transmitters within that sublist.

4.3 Can we guarantee that the optimum solution can be encoded by the order based representation?

Can we be sure that a sequence of transmitters exists, which upon application of our smallest feasible channel algorithm, will produce an optimum solution? In [19] although optimal solutions for several instances of the MSFAP were produced using a GA and an order based encoding, we did not prove that it is *always* possible to encode optimal solutions in this way. A proof for this is outlined below. Recall that the *smallest feasible channel* decoder, which transforms the permutation lists into channel allocations, operates as follows:

1. Take the first transmitter on the list and allocate channel 1.
2. Take the next transmitter on the list and allocate the smallest feasible channel which will not generate interference with any transmitter previously allocated a channel.
3. Repeat from 2) until the list is exhausted.

The proof that follows verifies that it is always possible to encode a channel allocation of minimum span for an instance of the MSFAP.

Represent the transmitters by the vertices, $V(G)$ of a constraint graph G . Definitions 1, 2 and 3 below are based on definitions which appear in [1].

Definition 1 *A constraint graph G is a finite, simple, undirected graph in which each edge v_iv_j($v_iv_j \in$ V(G), i and j are positive integers) has a non-negative integer label ϕ_{ij}.*

Definition 2 *A feasible channel assignment (or frequency assignment) on a constraint graph G is a mapping $f : V(G) \to F$ (where F is a set of consecutive integers m,..., n) such that the constraints*

$|f(v_i) - f(v_j)| \quad > \phi_{ij}$.

are satisfied for all $v_iv_j \in$ E(G). This is referred to a zero-violation assignment. The elements of the set F, can be referred to as channels (or frequencies), where for each vertex $v_i \in$ V(G), $v_i \to f(v_i)$, represents a channel assignment of a vertex, v_i.

Definition 3 *If K = n - m (where m is the smallest channel used, and n is the largest channel used) is a minimum over all zero-violation assignments then the assignment is a minimum assignment. This minimal value of K is the minimum span of G, denoted sp(G).*

Definition 4 establishes the properties of a *smallest feasible channel assignment*; i.e. the properties possessed by a channel assignment which has been produced by the application of the smallest feasible channel algorithm to a sequence of vertices (transmitters).

Definition 4 *Let $S_{V(G)}$ be an arbitrary sequence, $v_0, v_1, \ldots\ldots, v_{|V(G)|-1}$, of all vertices in V(G). A smallest feasible channel assignment is a channel assignment, $g : S_{V(G)} \to F$ (where F is a set of consecutive integers m,....,n, as before) such that the following constraints are satisfied:*

Constraint 1: $|f(v_i) - f(v_j)| \quad > \phi_{ij}$. *for $v_iv_j \in$ E(G) : $j < i$ (where i and j represent positions in the sequence, $S_{V(G)}$).*

Constraint 2 :*in addition, for each vertex $v_i \in S_{V(G)}$, exactly one of the following statements must be true:*

statement 1: $|f(v_i) - f(v_j)| = \phi_{ij} + 1$, *for at least one edge, $v_iv_j \in$ E(G) with $j < i$, if at least one edge, v_iv_j, exists with $j < i$,*

statement 2: $f(v_i) = m$, *the smallest channel available, if $\forall$ j< i $v_iv_j \notin$ E(G).*

In other words a *smallest feasible channel assignment* will be the result produced by applying the smallest feasible channel assignment algorithm to a list of vertices (transmitters) in sequence. As the algorithm progresses through the sequence of vertices, it will assign the smallest channel possible to each one; i.e. the smallest channel which does not violate any constraints imposed by channel allocations to vertices which appear earlier in the sequence.

Theorem 1 *Given a constraint graph G with a set of vertices V(G), there exists some sequence of vertices, $S_{V(G)}$, such that the application of the smallest feasible channel assignment algorithm to that sequence will produce a minimum assignment, where the difference between the smallest and largest channels used will be sp(G), the minimum span.*

Proof For any chosen minimum span assignment f of G generating a mapping $f : V(G) \to F$ such that F is a set of consecutive integers m,....,n, number the vertices in non-descending order of their channel assignment, $f(v_i)$ for v$_i \in V(G)$, to produce a sequence v_0, v_1,......,$v_{|V(G)|-1}$. Call this sequence S_{min}. Next, convert the *minimum channel assignment* to a *smallest feasible channel assignment* in the following way:

Starting at v_0, step through the sequence, v_0,v_1,....,$v_{|V(G)|-1}$, reassigning a vertex, v_i, to a smaller frequency channel where it is necessary to do so in order to avoid violating constraint 2 in Definition 4.

Using the smallest feasible channel procedure to allocate a smaller channel to some vertex, $v_i \in S_{min}$, when this is possible, avoids violation of constraints 1 and 2 of definition 4 for $v_iv_j \in V(G)$ when $j < i$ (given). However, it is necessary to prove that allocating a smaller channel to v_i cannot result in the violation of constraint 1 for $v_iv_j \in V(G)$ when $j > i$. If this were to happen we could finish with a channel assignment which is not a minimum channel assignment. (It does not matter if constraint 2 is temporarily violated for some edge, v_iv_j for $j > i$, when some vertex, v_i has a lower channel allocated to it. The violation will be corrected later on when the smallest feasible channel procedure allocates a new channel to v_j.)

By definition of S_{min}, $f(v_i) \leq f(v_j)$ for all $v_iv_j \in E(G)$ when $j > i$, so making $f(v_i)$ smaller, $f'(v_i)$ say, it follows that

$|f'(v_i) - f(v_j)| \geq |f(v_i)$ - $f(v_j)| > \phi_{ij}$. for all $v_iv_j \in E(G)$ when $j > i$,

thus establishing that constraint 1 holds at all times during the application of the smallest feasible channel procedure to the sequence, S_{min}.

The smallest feasible channel assignment on S_{min} will be an assignment of minimum span, because:

1. the span cannot be larger than $sp(G)$, as the smallest feasible channel procedure applied to the sequence S_{min} cannot assign a larger channel to any vertex in the sequence, without violating constraint 2 of Definition 4.
2. the span cannot be smaller than $sp(G)$ by definition.

5 Do the crossovers pass useful information from parents to offspring?

The starting point is an initial population of 1,000 individuals produced by applying the generalized saturation degree algorithm to 1,000 random permutations of the set of transmitters for the problem in question. One thousand pairs of parents are selected using the selection mechanism defined in 1, and from those parents 1,000 offspring are generated using the various crossovers and no mutation. It is important to establish that offspring resemble their parents and produce solutions with similar values for the span. If this turns out not to be the case, the GA is likely to perform even worse than random search (because random search maintains diversity, and a GA does not).

Table 2 shows the values of Pearson's Product Moment Correlation Coefficient, r_{xy}, for offspring (y coordinate) versus mid-parent (x coordinate) values of the span for 1,000 samples. The value of r_{xy} is +1 for perfect direct correlation and -1 for perfect inverse correlation. Values of r_{xy} close to zero indicate that there is no direct relationship between the x and y values. (I refer the reader to a statistics text, for example chapter 8 of [7], for more details of r_{xy}.) Results are shown for eight problems with the numbers of transmitters varying between 95 and 726. The values of 'c' in the third column represent the co-site constraints making up part of the constraint matrix.

Table 2: Offspring versus mid-parent correlation for test problems

Problem No.	transmitters	c	CX	PMX	OX	UOBX
P1	95	4	0.3460	0.2159	0.1653	0.1562
P1	95	5	0.2067	0.1823	0.2391	0.0805
P2	190	5	0.2089	0.1520	0.1229	0.1614
P3	225	–	0.0385	- 0.0184	- 0.0108	- 0.0051
P4	410	5	0.2801	0.1172	0.1117	0.1762
P5	481	–	0.2993	0.1504	0.1182	0.0091
P6	726	4	0.3512	0.2162	0.1463	0.3415
P6	726	5	0.2970	0.0468	0.0985	0.2850

The values for the correlation coefficient in the table indicate that CX is more successful than other crossovers in passing on useful parental characteristics to offspring. (Observe that values of r_{xy} are larger in the CX column than anywhere else.) A significance test shows that under CX the r_{xy} values are highly significant at the 0.0001% level for 7 out of the 8 problems, indicating that parental features that contribute to their span values are indeed passed on to their offspring. Results for the problem P3, however, would indicate that none of the crossovers is able to pass on useful features from parents to offspring in this instance, suggesting that the GA is unlikely to be effective in solving this particular problem. Uniform Order Based Crossover would appear to be almost as successful as Cycle Crossover for problem P6.

6 Results

Table 3 gives the characteristics of the test problems used. P1, P2, P4 and P6 are all computer generated realistic examples. Problem P5 is the so-called Philadelphia problem, a cellular phone problem used in [19] and previously studied by many other authors, particularly [8] and [15]. P3 is also a cellular phone example. This example can be generated from the ideas that appear in [12] (with $d_0 = 25$, $d_1 = 13$).

Table 3: Test data characteristics

	No. transmitters	No. co-site constraints	No. far-site constraints	Edge density
P1	95	90	1124	0.27
P2	190	160	4882	0.28
P3	225	0	8163	0.32
P4	410	411	22346	0.27
P5	481	0	97835	0.85
P6	726	711	74595	0.29

6.1 Comparing the crossovers

All 4 crossovers (CX, PMX, OX and UOBX) are tested on the 726 transmitter problem P6 with co-site = 4. Populations of 1,000 are used and averages of 5 runs recorded with one mutation (order based) per individual, running for 500 generations. Figure 1 plots the best-so-far curves obtained using the 4 different crossovers with a starting population of random permutations. Figure 2 plots the best-so-far curves obtained using the same crossover operators on a starting population seeded with GSD orderings.

The relative performance of the GA using the four alternative crossover operators on a random starting population is much as would be predicted by the offspring verses mid-

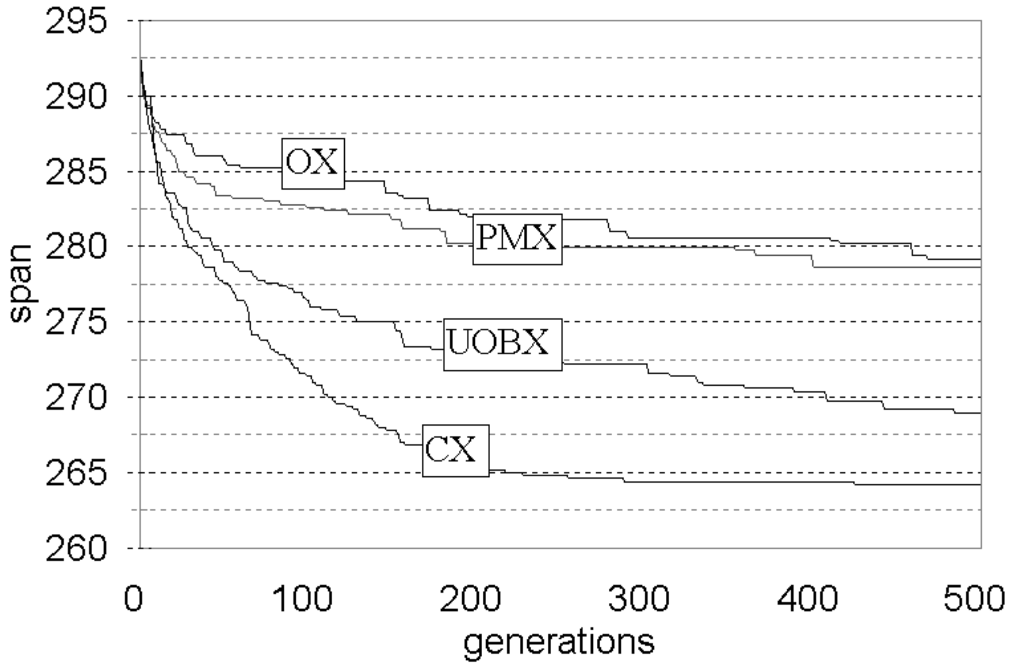

Figure 1: Permutation crossovers for problem P6 applied to random population

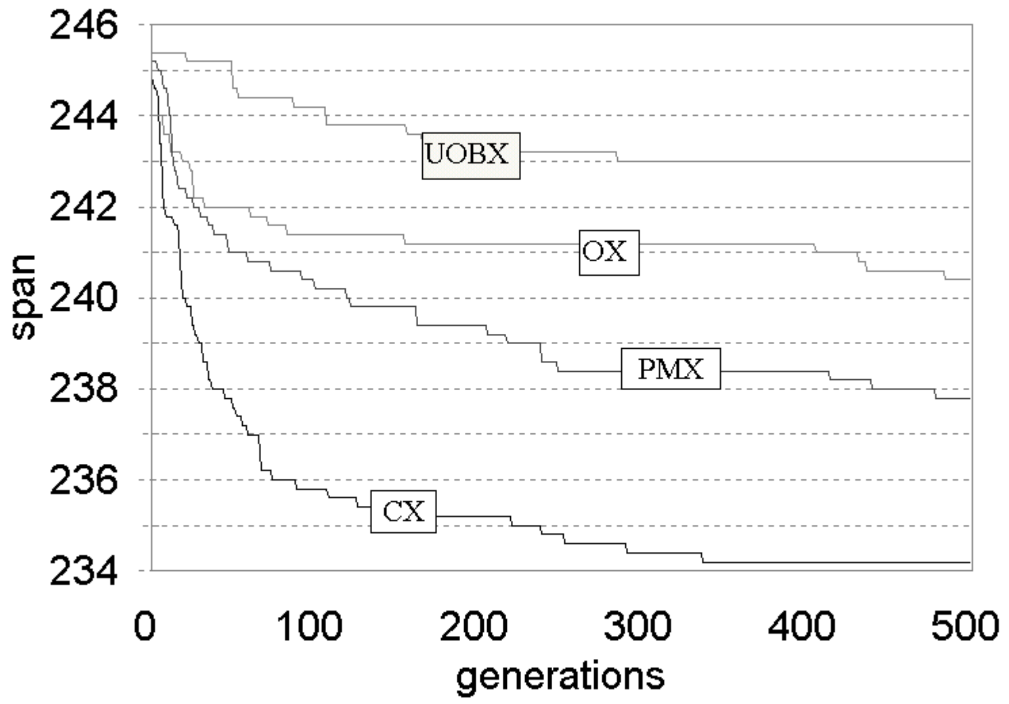

Figure 2: Permutation crossovers for problem P6 on a GSD seeded population

parent correlation coefficients. CX performs best, with UOBX second, PMX third and OX last. Using a GSD seeded population, however, the performance of the GA incorporating CX retains its superiority, but the relative performance of UOBX is exceedingly poor. For this set of experiments CX is best, with PMX second followed by OX, with UOBX last. One could speculate that perhaps the GSD algorithm and the UOBX crossover somehow work together to reduce the 'novelty value' produced during the genetic search. (Recall that the steady state GA used here replaces a 'weaker parent' with a 'stronger offspring'. Thus in situations in which the genetic search fails to generate any 'stronger offspring', no progress will be made by the GA, however promising the offspring versus mid-parent correlation coefficients may appear). This is worthy of further investigation.

6.2 Comparing the mutations

The three mutations described in section 4 are all tried on problem P6, $c = 4$, using CX as the crossover operator. Populations of 1,000 are used and the GA allowed to run for 500 generations with 500 x 1,000 individuals processed. One mutation is applied per individual. The maximum size of sublist scrambled in the third type of mutation is 8 items. The number of items scrambled at each mutation is selected from a uniform distribution on x: $2 \leq x \leq 8$. The results, presented in Table 4, indicate position based mutation produces the best results, with order based mutation a close second.

Table 4: Comparison of mutation operators

Mutation	Average of 5 runs	Best of 5 runs
Position based	233	231
Order based	234.2	233
Scramble sublist	240.6	238

6.3 Results for GA with CX on larger problem set

The GA with CX crossover and position based mutation performed best in the above tests and is thus chosen for experiments on the larger problem set. For the experiments a population size of 1,000 is used with one mutation per offspring. Cycle crossover is the recombination operator. The GA stops after 100 generations have elapsed with no improvement to the best-so-far.

The population is initially seeded with 1,000 random permutations. These permutations are then subjected to the Generalized Saturation Degree algorithm, and it is the new orderings produced by the application of this algorithm which form the starting population for the GA.

In Table 5 column 3 records the mean best span for the sets of 5 replicate runs, and column 4 gives the best result obtained by the GA in each case. Column 5 displays the best result from a set of control experiments performed by generating large numbers of random permutations and applying the GSD algorithm to each structure. The stopping criterion for the algorithm used for the GSD on random orderings (column 5) halts when 100 x 1,000 individuals have been processed in which no improvement has been observed to the best-so-far. The idea here is to make the stopping criterion for the random search the same as the stopping criterion for the GA. (The 100 x 1,000 individuals correspond with 100 generations of the GA with 1,000 trials in each generation.) Column 6 presents the best results obtained by trying all combinations of various state-of-the-art sequential ordering algorithms (see [11]). The previously best published results can be found in column 7. The numbered superscripts in column 7 refer to the following publications:

1: [15],

2: [18].

The solution of span = 47 obtained for problem P1 ($c = 4$, and $c = 5$) matches the best lower bound for the 95 transmitter problem described in [15].

Table 5: Genetic algorithm on a larger problem set using CX and position based mutation

Problem	c	Mean 5 runs	Best GA	Random + GSD	Best sequential	Previously best published
P1	4	47.6	47	49	51	471[1]
P1	5	47.8	47	51	54	482[2]
P2	5	75.4	74	79	87	762[2]
P3	—	42	42	41	43	—
P4	5	146.4	145	150	158	1542[2]
P5	–	426	426	462	449	4261[1]
P6	4	233	231	243	249	–
P6	5	238.2	237	247	255	–

Some experiments have been tried by using larger populations for problem P6, $c = 4$. The best solution I have obtained using the GA is span = 229, with a population of 8,000.

7 Conclusions and Future Work

The genetic algorithm which breeds permutations for minimum span frequency assignment has produced some promising results, beating results produced by state-of-the-art sequential assignment methods and also some previously best published results. The approach hybridizes a GA (for breeding permutations) with a greedy algorithm (for doing the channel assignment to the permutations). The technique would appear to be particularly successful when it incorporates a position based crossover, indicating that it is the *absolute* positions rather than the *relative* positions of transmitters on the list which are important for the greedy channel allocation algorithm. Results are improved if the initial population is seeded with 'good permutations' prior to invoking the GA, using a generalized saturation degree algorithm as a pre-processor for the population. It is noticeable, however, that the GA is not successful on all problems. It performs very badly for example on P3 (a problem in which the constraints are very evenly distributed). An offspring versus mid-parent correlation appears to be a good predictor of success in most instances, it predicts failure for P3, and success elsewhere, accurately forecasting that CX will out-perform the other crossover operators. The relatively high value of the correlation coefficient for UOBX on P6 ($c = 4$), however, (it is nearly as high as for CX on this problem), does not translate into a good performance for the GA when GSD seeding is used. In fact the UOBX performs worse than any other crossover, suggesting that other factors are also involved here. Thus although I recommend offspring versus mid-parent correlation as a very useful tool, some caution should be observed in its application.

Work is currently in progress to extend the approach in the following ways:

1. to try incorporating mixes of several of the most promising genetic operators in a single GA,
2. to solve larger problems,
3. to implement the algorithm on parallel hardware, and
4. to extend the technique to the Fixed Spectrum Frequency Assignment Problem.

Acknowledgments

I should like to thank Steve Hurley for providing the results for the best sequential algorithm quoted in Table 5 and for many useful discussions.

References

[1] Allen, S. M., Hurley, S. and Smith, D. H. *Lower Bounds for Frequency Assignment*, in Methods and Algorithms for Radio Channel Assignment, edited by R. A. Leese, (to be published by Oxford University Press).

[2] Cavicchio, D.J. *Adaptive search using simulated evolution.* Unpublished doctorial dissertation, University of Michigan, Ann Arbor.

[3] Clark, T., and Smith, G.D. *A Practical Frequency Planning Technique for Cellular Radio.* Proceedings of the Third International Conference on Artificial Neural Nets and Genetic Algorithms. Pages 312-316.

[4] Crisan, Christine and Mühlenbein, Heinz. *The Breeder Genetic Algorithm for Frequency Assignment.* Proceedings of the 5^{th} International Conference on Parallel Problem Solving from Nature (PPSN V), Amsterdam, The Netherlands, 1998. Pages 897-906.

[5] Crompton, W., Hurley, S. and Stephens, N.M. *A parallel genetic algorithm for frequency assignment problems.* Proceedings IMACS/IEEE Conference on Signal Processing, Robotics and Neural Networks, pages 81 - 84, Lille, France, 1994.

[6] Davis, L. *Order-Based Genetic Algorithms and the Graph Coloring Problem.* In Handbook of Genetic Algorithms, Van Nostrand Reinhold, pages 72 - 90.

[7] Elmore, Patricia B., and Woehlke, Paula L. *Basic Statistics*, Longman 1997.

[8] Gamst, A. *Some lower bounds for a class of frequency assignment problems.* IEEE Transactions on Vehicular Technology, 35: pages 8 - 14.

[9] Gorges-Schleuter, M. *ASPARAGOS an Asynchronous Parallel Genetic Optimization Strategy.* Proceedings of the Third International Conference on Genetic Algorithms. Hillsdale, NJ: Lawrence Erlbaum Associates.

[10] Holland, J.H. *Adaptation in natural and artificial systems.* Ann Arbor:The University of Michigan Press.

[11] Hurley, S., Smith, D. and Thiel, S. FASoft: *A system for discrete channel frequency assignment.* Radio Science **32** , pages 1921-1939.

[12] Leese, R. A. *A unified approach to the assignment of radio channels on a hexagonal grid.* IEEE Transactions on Vehicular Technology, Vol. 46, No 4, pages 968-980

[13] Nagata, Yuichi and Kobayashi, Shigenobu. *Edge Assembly crossover: A high-power genetic algorithm for the traveling salesman problem.* Proceedings of the 7^{th} International Conference on Genetic Algorithms. Morgan Kaufmann, 1997. Pages 450-457.

[14] Oliver, I.M., Smith, D.J., and Holland, J.R.C. *A study of permutation crossover operators on the traveling salesman problem.* Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms, pages 224-230.

[15] Smith, D., Hurley, S. and Thiel, S. *Improving heuristics for the frequency assignment problem.* European Journal of Operational Research, Vol. 107, No 1, pages 76-86, ISSN 0377-2217..

[16] Syswerda, G. *Uniform Crossover in Genetic algorithms.* Proceedings of the Third International Conference on Genetic Algorithms. Hillsdale, NJ: Lawrence Erlbaum Associates.

[17] Valenzuela, Christine L. *Evolutionary Divide and Conquer: a Novel Genetic approach to the TSP.* Unpublished Doctorial Thesis, Imperial College, University of London 1995.

[18] Valenzuela, C.L. ,Jones A. and Hurley, S. *Breeding Permutations for Minimum Span Frequency Assignment.* Proceedings of the Third International Conference on Artificial Neural Nets and Genetic Algorithms. Pages 308 - 311

[19] Valenzuela, Christine, Hurley, Steve, and Smith, Derek . *A Permutation Based Genetic Algorithm for Minimum Span Frequency Assignment.* Proceedings of the 5^{th} International Conference on Parallel Problem Solving in Nature, Amsterdam, The Netherlands 1998. Pages 907-916.

[20] Whitley, Darrell. *Scheduling Problems and Traveling Salesman: The Genetic Edge Recombination Operator.* Proceedings of the Third International Conference on Genetic Algorithms, pages 133-140, San Mateo, CA, Ed. J. D. Schaffer, Morgan Kaufmann.

[21] Whitley, Darrell, in *Handbook of Evolutionary Computation* , Part C: "Evolutionary Computation Models", section C3.3. Institute of Physics Publishing.