Dynamic Multiple Neighborhood Structures for the Static Frequency Assignment Problem

KHALED ALRAJHI
King Khaled Military Academy, Riyadh, SAUDI ARABIA

Abstract. This study proposes a dynamic multiple neighborhood structures to solve a variant of the frequency assignment problem known as the minimum order frequency assignment problem. This problem involves assigning frequencies to a set of requests while minimizing interference and the number of used frequencies. Several novel and existing techniques are used to improve the efficiency of this algorithm and make it different from other applications of multiple neighborhood structures in the literature. This includes solving the static problem by modeling it as a dynamic problem through dividing this static problem into smaller sub-problems, which are then solved in turn in a dynamic process using multiple neighborhood structures. Moreover, applying technique that aims to determine a lower bound on the number of frequencies required from each domain for a feasible solution to exist for each sub-problem, based on the underlying graph coloring model. These lower bounds ensure that the search focuses on parts of the solution space that are likely to contain feasible solutions. This study considers real and randomly generated benchmark datasets of the static problem and our approach achieved competitive results.

Keywords: multiple neighborhood structures, minimum order frequency assignment problem.

Received: October 5, 2022. Revised: September 5, 2023. Accepted: October 7, 2023. Published: November 2, 2023.

1 Introduction

The frequency assignment problem (FAP) is related to wireless communication networks, which are used in many applications such as mobile phones, TV broadcasting and Wi-Fi. The aim of the FAP is to assign frequencies to wireless communication connections (also known as requests) while satisfying a set of constraints, which are usually related to prevention of a loss of signal quality. Note that the FAP is not a single problem. Rather, there are variants of the FAP that are encountered in practice. The minimum order FAP (MO-FAP) is the first variant of the FAP that was discussed in the literature, and was brought to the attention of researchers by [1]. In the MO-FAP, the aim is to assign frequencies to requests in such a way that no interference occurs, and the number of used frequencies is minimized. As the MO-FAP is NP-complete [2], it is usually solved by meta-heuristics.

Many meta-heuristics have been proposed to solve the MO-FAP including genetic algorithm (GA) [3], evolutionary search (ES) [4], ant colony optimization (ACO) [5], simulated annealing (SA) [6] and tabu search (TS) [6, 7, 8, 9]. It can be seen from the literature that TS is a popular meta-heuristic for solving difficult combinatorial optimization problems. This generally applicable algorithm has proved to be an efficient way of finding a high quality solution for a variety of optimization problems e.g. [10]. However, existing algorithms in the literature are unable to find optimal solutions in some datasets for the MO-FAP.

In this paper, we present a dynamic multiple neighborhood structures (DMNS), one of which is used as a diversification technique. The concept of using multiple neighborhood structures is inherited from the variable neighborhood search algorithm, introduced by [11]. In contrast, [6, 7, 8, 9] implemented only a single neighborhood structure in their TS algorithms. Moreover, DMNS algorithm applies the good starting point strategy by starting with a good initial solution using a greedy heuristic associated with a descent method. This should lead to more efficient solution method [12]. In contrast, an initial solution is randomly generated in [7, 13]. Another technique used in DMNS algorithm is applying a lower bound on the number of frequencies that are required from each domain for a feasible solution to exist for each sub-problem, based on the underlying graph coloring model. These lower bounds ensure that the search focuses on parts of the solution space that are likely to contain feasible solutions. Experiments were carried out on the CELAR and GRAPH datasets, and the results show that our TS algorithm outperforms other algorithms in the literature.

This paper is organized as follows: the next section gives an overview of the MO-FAP. Section 3 explains how to model the static MO-FAP to a dynamic problem. Section 4 explains how the underlying graph coloring model for the MO-FAP can be used to provide a lower bound on the number of frequencies for each instance and how this information can then be used to assist the search. In Sections 5 and 6, the overall structure of the DMNS algorithm for the static MO-FAP is outlined. In Section 7, the results of this algorithm are given and compared with those of other algorithms in the literature before this study finishes with conclusions and future work.

2 Overview of the MO-FAP

The main concept of the MO-FAP is assigning a frequency to each request while satisfying a set of constraints and minimizing the number of used frequencies. The MO-FAP can be defined formally as follows: given

- a set of requests $\mathrm{R} = \{r_1, r_2, \ldots, r_{\mathrm{NR}}\}$, where NR is the number of requests,
- a set of frequencies $\mathrm{F} = \{f_1, f_2, \ldots, f_{\mathrm{NF}}\} \subset \mathbb{Z}^+$, where NF is the number of frequencies,
- a set of constraints related to the requests and frequencies (described below),

the goal is to assign one frequency to each request so that the given set of constraints are satisfied and the objective function is minimized, where the objective function is minimizing the number of used frequencies. Note that the frequency that is assigned to requests r_i is denoted as f_{r_i} throughout of this study. The MO-FAP has four variants of constraints as follows:

1. Bidirectional Constraints: this type of constraint forms a link between each pair of requests $\{r_{2i-1}, r_{2i}\}$, where $i = 1,\ldots,NR/2$. In these constraints, the frequencies $f_{r_{2i-1}}$ and $f_{r_{2i}}$ that are assigned to r_{2i-1} and r_{2i}, respectively, should be distance $d_{r_ir_j}$ apart. In the datasets considered here, $d_{r_ir_j}$ is always equal to a constant value (238). These constraints can be written as follows:

$$\left|f_{r_i} - f_{r_j}\right| = d_{r_ir_j} \qquad \text{for } i = 1, \ldots, \mathrm{NR}/2 \qquad (1)$$

2. Interference Constraints: this type of constraint forms a link between a pair of requests $\{r_i, r_j\}$, where the pair of frequencies f_{r_i} and f_{r_j} that is assigned to the pair of requests r_i and r_j, respectively, should be more than distance $d_{r_ir_j}$ apart. These constraints can be written as follows:

$$\left|f_{r_i} - f_{r_j}\right| > d_{r_ir_j} \qquad \text{for } 1 \le i < j \le \mathrm{NR} \qquad (2)$$

3. Domain Constraints: the available frequencies for each request r_i are denoted by the domain $D_{r_i} \subset F$, where $\cup_{r_i \in R} D_{r_i} = F$. Hence, the frequency which is assigned to r_i must belong to D_{r_i}. For the datasets considered in this study, there are 7 available domains.
4. Pre-assignment Constraints: for certain requests, the frequencies have already been pre-assigned to given values i.e. $f_{r_i} = p_{r_i}$, where p_{r_i} is given value.

3 Modeling the Static MO-FAP as a Dynamic Problem

In the DMNS algorithm, the static MO-FAP is broken down into smaller sub-problems, each of which is considered at a specific time period. To achieve this, each request is given an integer number between 0 and n (where n is a positive integer) indicating the time period in which it becomes known. In effect, the problem is divided into $n + 1$ smaller sub-problems $P_0, P_1, \ldots, P_n$, where n is the number of sub-problems after the initial sub-problem P_0. Each sub-problem P_i contains a subset of requests which become know at time period i. The initial sub-problem P_0 is solved first at time period 0. After that, the next sub-problem P_1 is considered at time period 1 and the process continues until all the sub-problems are considered. In this study, we found that the number of sub-problems does not impact on the performance of the DTS approach for solving the static MO-FAP, so the number of sub-problems is fixed at 21 (i.e. $n = 20$).

Based on the number of the requests known at time period 0 (belonging to the initial sub-problem P_0), 10 different versions of a dynamic problem are generated. These versions are named using percentages which indicate the number of requests known at time period 0. These 10 versions are named 0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% (note that 100% means all the requests are known at time period 0 and so corresponds to the static MO-FAP).

4 Graph Coloring Model for the MO-FAP

The graph coloring problem (GCP) is an underlying model to the MO-FAP [14]. The GCP involves allocating a color to each vertex such that no adjacent vertices are in the same color class and the number of colors is minimized. The MO-FAP can be represented as a GCP by representing each request as a vertex and a bidirectional or an interference constraint as an edge joining the corresponding vertices.

One useful concept of graph theory is the idea of cliques. A clique in a graph can be defined as a set of vertices in which each vertex is linked to all other vertices. A maximum clique is the largest among all cliques in a graph. Vertices in a clique have to be allocated to a different color in a feasible coloring. Therefore, the size of the maximum clique acts as a lower bound on the minimum number of colors and therefore, by extension, as a lower bound on the number of frequencies for the MO-FAP. For example, the requests r_1, r_{200}, r_{871}, r_{872} and r_{899} form a clique in the CELAR 01 instance (see Figure 1). All of these requests are linked to each other by either a bidirectional (see Equation 1) or an interference constraint (see Equation 2).

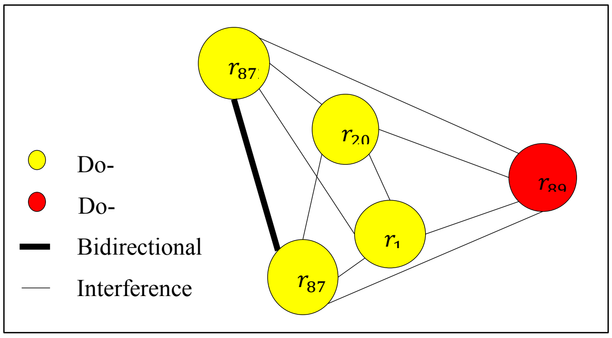

Fig. 1. An example of a clique in the CELAR 01 instance in the graph coloring model.

Figure 1 shows 5 different requests forming a clique, so at least 5 different frequencies are required. Note that r_1, r_{200}, r_{871} and r_{872} belong to domain 1, while r_{899} belongs to domain 3. As the requests belong to different domains, the graph coloring model for each domain can be considered separately and then a lower bound on the number of frequencies that is required from each domain can also be calculated. Generalizing the clique problem concept for all datasets gives a lower bound of the number of frequencies which are required from each domain as well as an overall lower bound on the total number of frequencies for a whole instance.

A Branch and Bound algorithm is used to obtain the set of all maximum cliques for each domain within each instance. Table 1 gives the minimum number of frequencies that is required in each domain and the size of the maximum clique for the overall instance, together with the run time (in seconds) and the optimal number of used frequencies, which are known and are available from the FAP website[1].

Table 1. The lower bound of the number of frequencies for each domain.

Instance	Domain							Max. Clique	Run Time	Optimal Solution
	1	2	3	4	5	6	7			
CELAR 01	10	9	10	4	4	7	2	12	1.50	16
CELAR 02	10	0	10	0	0	0	2	14	0.02	14
CELAR 03	10	0	10	0	2	0	2	12	0.06	14
CELAR 04	10	0	10	4	2	0	2	44	0.34	46
CELAR 11	20	0	14	4	2	0	2	20	0.34	22
GRAPH 01	8	3	6	2	4	4	2	18	0.03	18
GRAPH 02	6	2	4	0	2	4	0	14	0.12	14
GRAPH 08	10	2	6	2	3	8	3	16	0.28	18
GRAPH 09	6	2	10	2	2	8	2	18	0.48	18
GRAPH 14	6	2	4	2	0	2	2	8	0.48	8

5 Overview of the Dynamic Multiple Neighborhood Structures

A key decision when designing the DMNS algorithm is the definition of the solution space and the corresponding cost function.

5.1 Solution Space and Cost Function

In most cases, it has been found to be relatively straightforward to find solutions that satisfy the bidirectional, the domain and the pre-assignment constraints, as well as to define a neighborhood operator that moves between such solutions. Here the solution space S is defined as the set of all possible assignments satisfying all of the bidirectional, the domain and the pre-assignment constraints. Note that the interference constraints are relaxed in S. Only the interference constraints are relaxed because these are the most difficult constraint to be satisfied. The cost function CF is defined as the number of broken interference constraints, also known as the number of violations. This configuration has been used previously in the literature e.g. [6, 8, 15]. One of the advantages of using this configuration is that, in effect, the number of requests is halved because each request is linked with another request based on the bidirectional constraints (see Equation 1). As a result, here requests and frequencies are considered as pairs (instead of individuals). A pair of requests is denoted as $\{r_{2i-1}, r_{2i}\}$, where $i = 1, \dots, NR/2$, and a pair of frequencies is denoted as $\{f_k, f_k'\}$ throughout this study.

A further approach is also considered where the bidirectional constraints are not enforced and the solution space consists of solutions that satisfy only the domain and the pre-assignment constraints, while cost function counts the number of broken bidirectional and interference constraints. This configuration has been used previously in the literature e.g. [7, 13].

The solution space could have been defined as the set of all possible feasible assignments, that is, satisfy all of the constraints, and the corresponding cost function is the number of used frequencies. However, it may be difficult to move from one feasible solution to another. Furthermore, there is a weakness in the definition of cost function. This weakness can be seen when a large number of neighbor solutions with the same cost may differ greatly in their quality [16]. Therefore, this type of solution space is not considered in this study.

5.2 Second problem in the MO-FAP

Based on the definition of the above solution space which relaxes some constraints, a sub-problem is defined as minimizing the number of violations with a fixed number of used frequencies n_f to help us find a feasible solution. If a feasible solution is found, then the number of used frequencies is reduced to $n_f - 2$ in the creating violations phase (described in Section 6.3) and the sub-problem is reconsidered. The process is repeated until a feasible solution can no longer be found. This process is similar to [17] for the GCP, and [6, 8] for the MO-FAP.

5.3 Structure of the Dynamic Multiple Neighborhood Structures

the DMNS algorithm consists of three phases, namely the initial solution phase, the creating violations phase and the improvement phase. The initial solution phase (described in Section 6.2) generates an initial solution that we assume is feasible and uses n_f frequencies. Then, the creating violations phase (described in Section 6.3) reduces the number of used frequencies n_f by removing a pair of used frequencies $\{f_k, f'_k\}$ from the current solution. Then, all pairs of requests that are assigned to $\{f_k, f'_k\}$ are re-assigned to another pair of used frequencies, which may result in some violations. The improvement phase (described in Section 6.4) aims to find a feasible solution by reducing the number of violations to zero, using three neighborhood structures. If it results in a feasible solution within a specified number of iterations, then the creating violation phase is revisited to remove another pair of used frequencies. After that, the process continues until either, no feasible solution can be found, at which time the process is terminated, and the feasible solution in the previous iteration with $n_f + 2$ frequencies is returned or the optimal solution is found. Note that if the initial solution is not feasible, the violating phase can be omitted and the search moves immediately to the improvement phase.

The overall structure of the DMNS algorithm for the MO-FAP is illustrated in Figure 2.

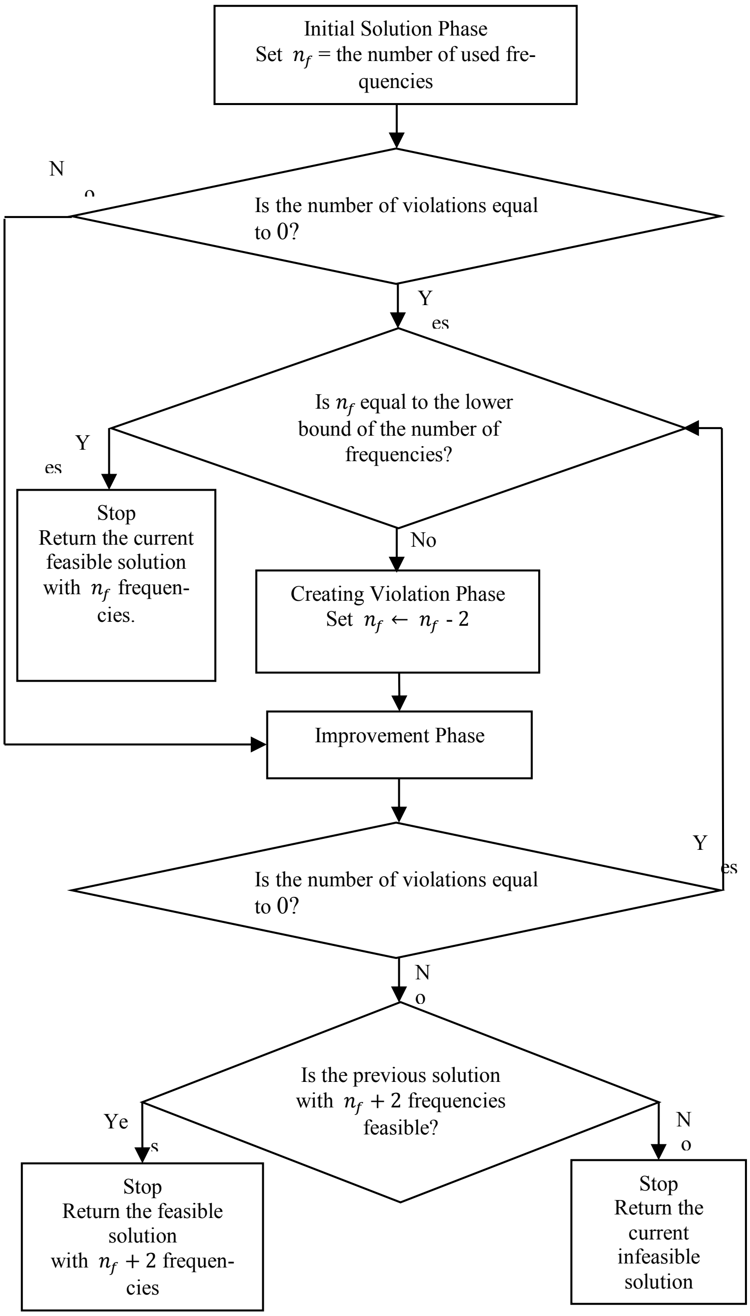

Fig. 2. Overall structure of the DMNS algorithm for the MO-FAP.

6 Components of the DMNS

This section presents the components of the DMNS algorithm in turn. Throughout, all the constraints except the interference constraints are regarded as hard constraints.

6.1 Neighborhood Structures

Three different neighborhood structures are considered, namely move, swap and diversification neighborhood structures. These are defined as follows:

- Move Neighborhood Structure (MNS): this structure is defined as the set of solutions obtained from the current solution by selecting a pair of requests $\{r_{2i-1}, r_{2i}\}$, where $i = 1, \ldots, \text{NR}/2$, and re-assigning them to a different pair of used frequencies $\{f_k, f_k'\}$ while satisfying all the hard constraints. Hence, this neighborhood investigates all the possible moves for all pairs of requests and used frequencies (the maximum possible number of such moves is $\text{NR} \times n_f$) and ensures that the number of used frequencies n_f does not increase. This structure is simple and most commonly used for TS algorithms in the literature e.g. [6, 7, 8].
- Swap Neighborhood Structure (SNS): this structure is defined as the set of solutions obtained from the current solution by swapping the frequencies of a pair of requests $\{r_{2i-1}, r_{2i}\}$, where $i = 1, \ldots, \text{NR}/2$. SNS proves to be quick as it contains a small number of neighbors (the maximum possible size is $\text{NR}/2$). Nevertheless, it can improve the solution quality.
- Diversification Neighborhood Structure (DNS): this structure, unlike the previous structures, is intended to diversify the search and ~~to~~ moves to a different part of the solution space. It consists of the set of solutions obtained from the current solution by replacing a pair of used (old) frequencies with a pair of unused (new) frequencies. Given a pair of old frequencies, another pair of frequencies is accepted if it can be assigned to all pairs of requests which were assigned to the old pair without breaking any hard constants, although violation may incur. After pairs of old and new frequencies are selected, then all pairs of requests that are assigned to the pair of old frequencies will be re-assigned to the pair of new frequencies. However, any re-assignment that causes the number of used frequencies to drop below the lower bound for some domains (see Section 3) is not considered. This is in fact unlikely as the pair of new frequencies have to be valid for all pairs of requests assigned to the pair of old frequencies. However, as some frequencies occur in more than one domain, the lower bounds on other domains may be breached.

6.2 The Initial Solution Phase

A greedy heuristic algorithm is used to generate an initial solution as follows: a pair of requests which has the smallest number of feasible pairs of frequencies is selected. Then, among those pairs of frequencies, the one which is feasible for most pairs of requests is assigned to the selected pair of requests. In case there are no feasible pairs of frequencies, then a pair of frequencies is randomly selected. If the initial solution is infeasible, then a descent method with MNS (described in Section 6.1) is used to reduce the number of violations.

6.3 The Creating Violations Phase

This phase aims to reduce the number of used frequencies in a feasible solution by removing a pair of frequencies based on the bidirectional constraints. The removed pair must satisfy the following conditions: firstly, neither of the frequencies should be required to satisfy any pre-assignment constraints. Secondly, the lower bound on the number of frequencies that are required from each domain based on the underlying graph coloring model must be satisfied after deleting these frequencies. If there is more than one candidate pairs of frequencies, then the one which is assigned to the least number of pairs is selected. If there is still more than one such pair, then one of them is selected randomly. After that, the pairs of requests which are assigned to the candidate pair of frequencies will be re-allocated to a feasible pair of used frequencies. If there is no feasible pair of used frequencies, then these requests will be re-allocated to an infeasible pair of used frequencies at random. If this process leads to a feasible solution, then a further pair is removed. Otherwise, the improvement phase is executed to attempt to find a feasible solution, which is described in Section 5.6. The concept of the creating violations phase was used previously in the literature e.g. [8].

6.4 The Improvement Phase

Ordering of Neighborhood Structures. The iterative procedure of the DMNS algorithm starts in the improvement phase. The objective of this phase is to find a feasible solution, i.e. a solution with zero violations. The improvement phase consists of three neighborhood structures (MNS, SNS and DNS). In MNS and SNS, only used frequencies are considered, while DNS considers only unused frequencies. MNS is explored first because it contains a large number of neighbors. The SNS structure, which covers a limited number of neighbors, is then considered. Therefore, this structure is intended to support the MNS. DNS aims to jump from the current position in the solution space to a new position by removing a pair of used frequencies and adding a new one from the set of pair of unused frequencies to the current solution. Therefore, DNS is intended to diversify the search rather than reduce the number of violations, which reflects the reason for leaving it as the last structure.

Implementation of the Improvement Phase. Each iteration involves one of the three neighborhood structures by attempting them consecutively until some criteria are satisfied. The search begins with MNS. If this structure results in a better solution, then it is executed. Otherwise, it is repeated until the structure is executed for given number of times consecutively without improvement. Following this, the search enters SNS. If this structure leads to a better or equally good solution, then the search goes back to MNS. Otherwise, it appears there is little prospect of finding a better solution in the current region of the solution space and so the search enters DNS. A solution from DNS is selected and the search returns to MNS.

It was found that on occasions, it is not possible to find any diversification move using DNS because all options are tabu. This is because a significant number of diversifications will not be allowed due to the pre-assignment constraints as well as the information from the lower bound on the number of frequencies that are required from each domain based on the underlying graph coloring model. If this happens, the criteria of selecting a pair of new frequencies in DNS will be modified. A pair of frequencies is accepted as a pair of new frequencies if it can be allocated to at least one pair (instead of all pairs) of requests assigned to the pair of old frequencies. Although the pair of new frequencies will not be allowed to be removed because of the diversification tabu list, the pair of old frequencies will be allowed to return to the solution because of a limited number of neighbors in this structure. DNS is executed for a given number of times.

The output of the improvement phase can be a feasible or an infeasible solution. If it is a feasible, but not optimal solution, then the algorithm will direct the process to the creating violations phase. On the other hand, if the output is an infeasible solution, then the algorithm will return to MNS. This continues until either the stopping criteria are satisfied or the optimal solution is found.

6.5 Stopping Criteria

the DMNS algorithm has three different stopping criteria described as follows: (i) the feasible solution whose number of frequencies is equal to the lower bound is found, as this is the optimal solution, (ii) the number of iterations is equal to the maximum number of iterations, (iii) the DNS is executed for a certain number of times.

7 Experiments and Results

This section provides the results of the DMNS algorithm for the MO-FAP using CELAR and GRAPH datasets (available on the FAP website[2]). Moreover, the process of the DMNS algorithm is discussed and analyzed. Finally, the performance of the DMNS algorithm is compared with other algorithms in the literature.

Table 2 presents details of the MO-FAP datasets considered in this study including the numbers of requests and constraints for each instance.

Table 2. Details of the CELAR and the GRAPH datasets.

Instance	No. of Requests	No. of Bidirectional Constraints	No. of Interference Constraints	No. of Domain Constraints	No. of Pre-assignment Constraints	Total No. of Constraints
CELAR 01	916	458	5,090	916	0	6,464
CELAR 02	200	100	1,135	200	0	1,435
CELAR 03	400	200	2,560	400	0	3,160
CELAR 04	680	340	3,627	400	280	4,647
CELAR 11	680	340	3,763	680	0	4,783
GRAPH 01	200	100	1,034	200	0	1,334
GRAPH 02	400	200	2,045	400	0	2,645
GRAPH 08	680	340	3,417	680	0	4,437
GRAPH 09	916	458	4,788	916	0	6,162
GRAPH 14	916	458	4,180	916	0	5,554

Based on experimentations, the parameters of the DMNS algorithm are set as follows:

- The maximum number of iterations is 10,000.
- The maximum number of accepting worst solution consecutively in MNS is 100.
- The maximum number of executing DNS is 20.

In this study, the algorithm was coded using FORTRAN 95 and all experiments were conducted on a 3.0 GHz Intel Core I3-2120 Processor (2nd Generation) with 8GB RAM and a 1TB Hard Drive.

7.1 Results Comparison of the DMNS algorithm

This section provides the results of the DMNS algorithm for the MO-FAP. Five runs are performed for each instance, and each run uses a different random number stream. The results include the number of used frequencies in the best, the worst and the average solutions (with the optimal ones shown in bold), the average run time for each instance and the optimal solutions (known and available on the FAP website[2]). Note that the run time includes the run time of finding the lower bound of the number of frequencies for each domain (Table 1).

Table 3. Results of the DMNS algorithm for the MO-FAP.

Instance	Best Found	Worst Found	Average Solution	Average Time	Optimal Solution
CELAR 01	**16**	**16**	**16**	3.63 min	16
CELAR 02	**14**	**14**	**14**	0.52 sec	14
CELAR 03	**14**	16	14.8	1.00 min	14
CELAR 04	**46**	**46**	**46**	54.34 sec	46
CELAR 11	38	40	38.4	8.81 min	22
GRAPH 01	**18**	**18**	**18**	5.43 sec	18
GRAPH 02	**14**	**14**	**14**	2.16 sec	14
GRAPH 08	**18**	**18**	**18**	24.28 sec	18
GRAPH 09	**18**	**18**	**18**	3.01 min	18
GRAPH 14	**8**	**8**	**8**	4.81 min	8

Table 3 shows that the DMNS algorithm achieved optimal solution for all the instances except CELAR 11 and the solutions were obtained in a reasonable time, mostly less than 5 minutes.

A further approach is considered where the bidirectional constraints are not enforced and the solution space consists of solutions that satisfy only the domain and the pre-assignment constraints, and cost function counts the number of broken bidirectional and interference constraints. This approach was tried but did not lead to good results compared with the former one. This shows that enforcing bidirectional constraints is an important factor in improving the search efficiency for this application.

7.2 Analysis of Implementation Process

In this section, the process of the DMNS algorithm is discussed and analyzed. Figure 3 shows the number of used frequencies and the number of violations during a run using the CELAR 01 instance.

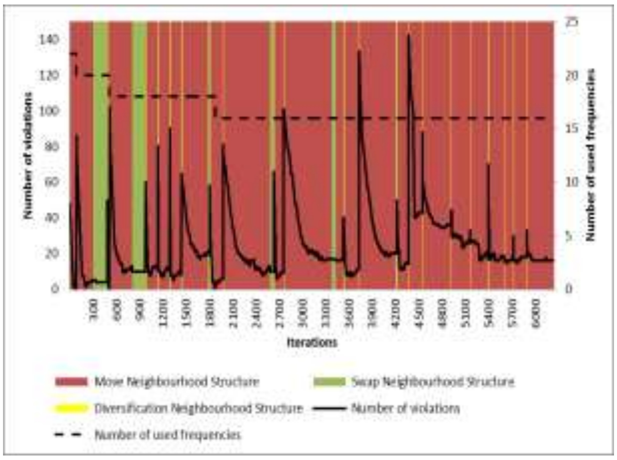

Fig. 3. The number of used frequencies and violations in each iteration in CELAR 01.

Figure 3 shows that the DMNS algorithm start with an initial feasible solution using 22 frequencies and this number was reduced to 16 frequencies. Although all neighborhood structures have been involved during the process of this algorithm, the most executed structure is MNS, which is represented by the red color. This justifies the fact that this structure is the most successful and commonly used. SNS came as a second most executed structure. This reflects the limitation of this structure and its objective, which is to support MNS. DNS is executed in a limited number of times and most of the times it results in an increase in the number of violations. This agrees with the aim of this structure, which is to diversify the search rather than optimize it.

7.3 Results Comparison with Other Algorithms

The results of the DMNS algorithm and other algorithms in the literature are compared. Table 4 shows the best found results; where the result shown in bold means these reach the optimal solution and a dash "-" means that the result is not available.

Table 4. Results comparison of the DMNS algorithm with other algorithms in the literature.

Instance	GA[3]	ES[4]	SA[6]	TS[6]	TS[9]	DMNS	Optimal Solution
CELAR 01	20	-	**16**	**16**	18	**16**	16
CELAR 02	**14**	**14**	**14**	**14**	**14**	**14**	14
CELAR 03	16	**14**	**14**	**14**	**14**	**14**	14
CELAR 04	**46**	-	**46**	**46**	**46**	**46**	46
CELAR 11	32	-	24	**22**	24	24	22
GRAPH 01	20	**18**	-	**18**	**18**	**18**	18
GRAPH 02	16	**14**	-	**14**	16	**14**	14
GRAPH 08	-	-	-	20	24	**18**	18
GRAPH 09	28	-	-	22	22	**18**	18
GRAPH 14	14	-	-	10	12	10	8

It can be seen from Table 4 that DMNS achieved competitive results compared with those of other algorithms in the literature. In fact, it achieved the optimal solution for all the instances except for CELAR 11and GRAPH 14. Moreover, it is the only algorithm in Table 4 that achieved the optimal solution for GRAPH 08 and GRAPH 09. Note that the results of GA [3] are less satisfactory than of the other algorithms, where only two instances obtained the optimal solutions. Overall, DMNS showed competitive results compared with those of other algorithms in the literature. Furthermore, this study suggests that solving the static problem in dynamic process by modeling it as a dynamic problem leads to competitive results by using multiple neighborhood structures.

8 Conclusions and Future work

In this paper, we presented a novel approach for solving the static MO-FAP by multiple neighborhood structures algorithm. This approach solves this problem by modeling it as a dynamic problem through dividing this problem into smaller sub-problems, which are then solved in turn in a dynamic process using DMNS algorithm. Several techniques have been used to improve the performance of this algorithm. These include using a lower bound for each domain based on the underlying graph coloring model. Moreover, based on the definition of the solution space which relaxes some constraints, a second problem of minimizing the number of violations is considered to find a feasible solution with a fixed number of used frequencies after the creating violations phase. Based on the results comparison, the DMNS algorithm show competitive results comparing with other algorithms in the literature. Clearly, there are many other variants of DMNS that could have been assessed. For example, a more advanced neighborhood structure could be used such as swapping pairs of requests with each other or forming chains similar to Kempe Chains in the GCP. Further investigations of these are left as future work.

References

[1]. Metzger, B. (1970), Spectrum Management Tech-nique, Presentation at 38th National ORSA meeting (Detroit, MI).

[2]. Garey, M. and Johnson, D. (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman W.H. and Company, San Francisco, California.

[3]. Kapsalis, A., Chardaire, P., Rayward-Smith, V. and Smith, G. (1995), The Radio Link Frequency As-signment Problem: A Case Study Using Genetic Al-gorithms, Lecture Notes on Computer Science, pp. 117-131.

[4]. Crisan, C. and Mühenbein, H. (1998), The Frequen-cy Assignment Problem: A Look at the Perfor-mance of Evolutionary Search, Lecture Notes in Computer Science, Vol. 1363, pp. 263-274.

[5]. Parsapoor, M. and Bilstrup, U. (2013), Ant Colony Optimization for Channel Assignment Problem in a Clustered Mobile Ad Hoc Network, International Conference on Swarm Intelligence (ICSI), Vol. 1, pp. 314-322.

[6]. Tiourine, S., Hurkens, C. and Lenstra, J. K. (1999), Local Search Algorithm for the Radio Link Fre-quency Assignment Problem, Telecommunication System, Vol. 13, pp. 293-314.

[7]. Bouju, A., Boyce, J., Dimitropoulos, C., Vom Scheidt, G. and Taylor, J. (1995), Tabu Search for the Radio Links Frequency Assignment Problem, In Applied Decision Technologies, London, [ADT'95]. UNICOM Conference.

[8]. Hao, J., Dorne, R. and Galinier, P. (1998), Tabu Search for Frequency Assignment in Mobile Radio Networks, Journal of Heuristics Vol. 4, pp. 47-62.

[9]. Bouju, A., Boyce, J., Dimitropoulos, C., Vom Scheidt, G. and Taylor, J. (1995), Intelligent Search for the Radio Links Frequency Assignment Prob-lem, Proceedings of the International Conference on Digital Signal Processing.

[10]. Glover, F. and Laguna, M. (1997), Tabu Search Applications, In Tabu Search, pp. 267-303.

[11]. Mladenovic, N. and Hansen, P. (1997), Variable Neighborhood Search,

[12]. Yu-Bin, Z., Yu-Cai, Z., Hui, X. (2009), A Tabu Search Algorithm for Frequency Assignment Prob-lem in Wireless Communication Networks, WiCO-M'09 Proceedings of the 5th International Confer-ence on Wireless Communications, Networking and Mobile Computing, pp. 2848-2851.

[13]. Hao, J. and Perrier, L. (1996), Tabu Search for the Frequency Assignment Problem in Cellular Radio Networks, Technical Report LG12P, EMA-EERIE, Parc Scientifique Georges Besse, Names France.

[14]. Hale, W. (1980), Frequency Assignment: Theory and Applications, Proc. IEEE Vol. 68, pp. 1497-1514.

[15]. Dorne, R. and Hao, J. (1996), Constraint Handling in Evolutionary Search: A Case Study on Frequen-cy Assignment, 4th International Conference on Parallel Problem solving from Nature, Lecture Note in Computer Science 1141, pp. 801-810, Springer-Verlag.

[16]. Dowsland, K., Thompson, J. (2008), An Improved Ant Colony Optimisation Heuristic for Graph Color-ing, Discrete Applied Mathematics, Vol. 156, No. 3, pp. 313-324.

[17]. Hertz, A. and de Werra, D. (1987), Using Tabu Search Techniques for Graph Coloring, Computing Vol. 39, pp. 345-351.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.