International Journal of Applied Mathematics
& Statistical Sciences (IJAMSS);
ISSN (P): 2319–3972; ISSN (E): 2319–3980
Vol. 12, Issue 2, Jul–Dec 2023; 9–20
© IASET

International Academy of Science,
Engineering and Technology
IASET Connecting Researchers; Nurturing Innovations

A DYNAMIC ANT COLONY OPTIMIZATION FOR SOLVING THE STATIC FREQUENCY ASSIGNMENT PROBLEM

Khaled Alrajhi

Research Scholar, King Khaled Military Academy, Riyadh, Saudi Arabia

ABSTRACT

This study proposes dynamic ant colony optimization algorithm to solve the static frequency assignment problem. This approach solves 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. Several novel and existing techniques are used to improve the performance of this algorithm. One of these techniques is applying the concept of a well-known graph colouring algorithm, namely recursive largest first for each sub-problem. Furthermore, this study compares this algorithm using two visibility definitions. The first definition is based on the number of feasible frequencies and the second one is based on the degree. Additionally, we compare this algorithm using two trail definitions. The first one is between requests and frequencies. The second is between requests and requests. This study considers real and randomly generated benchmark datasets of the static problem and our algorithm achieved competitive results comparing with other ant colony optimization algorithms in the literature.

KEYWORDS: *Dynamic Ant Colony Optimization, Graph Colouring Algorithm, Frequency Assignment Problem*

Article History

Received: 18 Oct 2023 | Revised: 19 Oct 2023 | Accepted: 25 Oct 2023

INTRODUCTION

The frequency assignment problem (FAP) is related to wireless communication net-works, 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 literature that there are relatively few papers concerning the application of ACO to solve the FAP. However, existing ACO algorithms in the literature are unable to find a feasible solution in some instances of the MO-FAP. Hence, this study investigates whether ACO can be improved to be an effective solution method for the MO-FAP.

In this study, the dynamic ant colony optimization (DACO) is mainly designed to solve MO-FAP 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. Several novel and existing techniques are used in this study to improve the performance of DACO. One of these techniques is applying the concept of a well-known graph colouring algorithm, namely Recursive Largest First (RLF), which was proposed in [15]. RLF has not been used in ACO for the static FAP in the literature. Furthermore, this study compares DACO using two visibility definitions (see Section 5.3). The first definition is based on the number of feasible frequencies, which was previously used in ACO for the graph colouring problem (GCP) [10]. The second one is based on the degree, which was previously used in ACO for the GCP [12]. Additionally, we compare DACO using two trail definitions (see Section 5.4). The first one is between requests and frequencies, which was previously used in ACO for the static FAP [13]. Note that ACO in [13] decreases the level of trail for bad solutions, whereas we increase the level of trail for the unassigned requests for all available frequencies in order to be more attractive to be selected. This technique was previously used in ACO for the examination scheduling problem [11]. The second trail definition considered in this study is between requests and requests, which was previously used in ACO for the GCP [12].

This paper is organised as follows: the next section gives an overview of the static MO-FAP. Section 3 presents the modeling the static MO-FAP as a dynamic problem, section 4 shows the graph coloring model for the static MO-FAP. Section 5 presents the main components of our DACO algorithm for the static MO-FAP. Results of this algorithm are given and discussed in Section 6 before this study finishes with conclusions.

OVERVIEW OF THE STATIC MO-FAP

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

- A set of requestsR $= \{r_1, r_2, \dots, r_{\mathrm{NR}}\}$, where NR is the number of requests,
- A set of frequenciesF $= \{f_1, f_2, \dots, f_{\mathrm{NF}}\} \subset \mathbb{Z}^+$, 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 static MO-FAP has four variants of constraints as follows:

Bidirectional Constraints

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

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

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$d_{r_i r_j}$apart.These constraints can be written as follows:

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

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 toD_{r_i}.For the datasets considered in this study, there are 7 available domains.

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.

MODELING THE STATIC MO-FAP AS A DYNAMIC PROBLEM

In this approach, 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, $P_1, \dots, P_n$, 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 period0. After that, the next sub-problem $P_1$is considered at time period1 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 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).

An example of how a static MO-FAP is modeled as a dynamic problem is illustrated in Figure 1, where each node represents a request, each edge a bidirectional or interference constraint and each color a time period in which a request becomes known for the first time.

After breaking the static MO-FAP into smaller sub-problems, these sub-problems will be solved in turn.

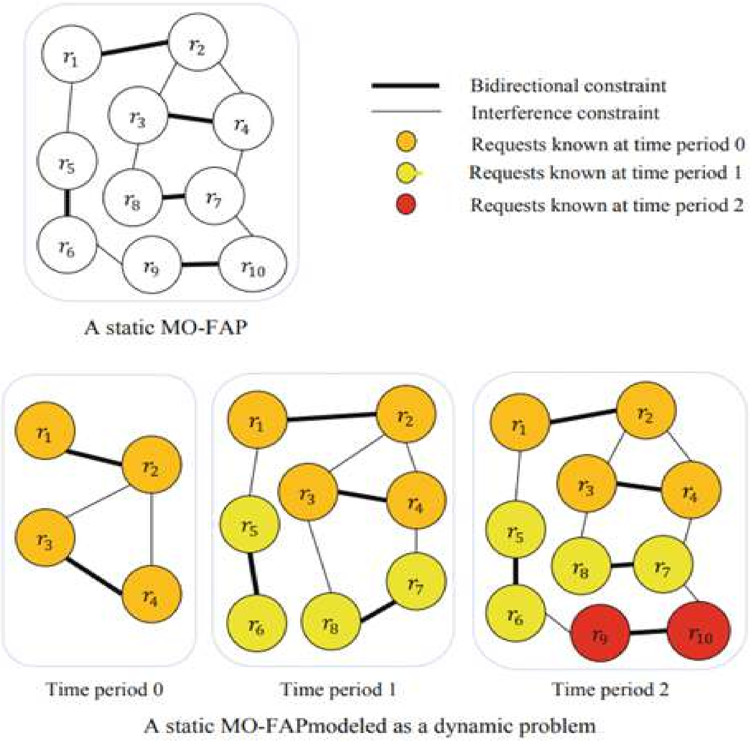


Figure 1: An Example of Modeling a Static MO-FAP as a Dynamic Problem Over 3 Time Periods.

GRAPH COLORING MODEL FOR THE STATIC MO-FAP

The graph coloring problem (GCP) can be viewed as an underlying model of the static MO-FAP [16]. 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 static 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.

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 be calculated. An overall lower bound on the total number of frequencies for a whole instance can also be calculated in a similar way. A branch and bound algorithm is used to obtain the set of all maximum cliques for each domain within each sub-problem.

OVERVIEW OF THE DYNAMIC ANT COLONY OPTIMIZATION

A key decision when designing DACO is how to choose the solution space and cost function, request and frequency selection, visibility definitions, trail definitions and descent method.

Solution Space and Cost Function

The solution space of DACO is defined as the set of all possible feasible assignments, that is, satisfying all of the constraints. The corresponding cost function is defined as the number of unassigned requests.

Request and Frequency Selection

DACO selects a frequency $f_j$greedily by selecting the one which can be assigned feasibly to the most requests. If there is more than one candidate frequency, then one of them is randomly selected. After that, the frequency $f_j$is sequentially feasibly assigned to all possible requests until no more can be feasibly assigned. The order of selecting requests from among those that are feasible for f_j is based on probability $p_{r_i f_j}$ given by Formula 3.

$$\frac{\tau_{r_i f_j}^{\alpha} . \eta_{r_i f_j}^{\beta}}{\sum} \quad p_{r_i f_j} = \begin{cases} \text{if } f_j \in G_{r_i} & 0 \qquad \text{otherwise} \end{cases} \tag{3}$$

where G_{r_i} is the set of frequencies which can be feasibly assigned by an artificial ant to the request r_i, The visibility $\eta_{r_i f_j}$of a request r_i to be assigned a frequencyf_j is defined in Section 3.3 and the trail$\tau_{r_i f_j}$is defined in Section 3.4.The parameters $\alpha, \beta \geq 0$ control the relative significance of the pheromone trail $\tau_{r_i f_j}$against the visibility $\eta_{r_i f_j}$.

After that, a different frequency is selected in the same way and this process is repeated until all requests are feasibly assigned, if possible. This process is inherited from a well-known graph colouring algorithm, namely recursive largest first. In fact, applying recursive largest first aims to improve the performance of selecting frequencies and requests to be assigned. In contrast, ACO for the MO-FAP in the literature (see e.g. [13]) frequently selects a request based on probability and then assign it to a feasible frequency.

Visibility Definitions

The visibility gives some indication of the desirability of choosing a request based on the experience of previous ants. Hence, the visibility of a request acts as a greedy heuristic. In this study, two types of visibility definition are applied and compared. These two visibilities are defined as follows:

i) Visibility $\eta_{r_i f_j}$of a request r_i to be assigned a frequencyf_j is based on the number of feasible frequencies for r_i (NFF_{r_i}), which is given by Formula 4.

$$\eta_{r_i f_j} = \frac{1}{NFF_{r_i}+1} \tag{4}$$

This definition prioritises those requests that have fewer feasible frequencies. This type of visibility definition was previously used in ACO for the graph colouring problem (GCP) [10].

ii) Visibility $\eta_{r_i f_j}$of a request r_i to be assigned a frequencyf_j is based on the degree of r_i (DEG_{r_i}), which is defined as the numbers of unassigned requests that cannot be assigned feasibly to$f_j$and have a common interference constraint withr_i. This visibility is given by Formula 5.

$$\eta_{r_i f_j} = DEG_{r_i} + 1 \tag{5}$$

This visibility looks ahead and prioritises requests that have more constraints in common with other requests that cannot be assigned to the frequencies being considered currently. This visibility definition was previously used in ACO for the GCP [12].

A request from among those that are feasible for the selected frequency$f_j$is selected based on the probability given by Formula 3 Here, assume that the trail and the parameters α andβ in Formula 3 are set to one. Then, the probability of selecting a request based on the two visibility definitions would be calculated as follows:

i) The probability of selecting each request using the first visibility definition is given in Table 1. Note that the number of feasible frequencies of each request (NFF_{r_i}) is invented and cannot be deduced from Figure 1.

Table 1: Requests Selection Based on Probability using the First Definition of Visibility

	r_1	r_3	r_5	r_7	Σ
NFF_{r_i}	1	2	3	4	
$\frac{1}{NFF_{r_i}+1}$	1	1/2	1/3	1/4	25/12
$p_{r_i f_j}$	0.44	0.28	0.13	0.15	1

ii) The probability of selecting each request using the second visibility definition is given in Table 2. Note that the degree of each request (DEG_{r_i}) can be deduced from Figure 1.

Table 2: Requests Selection Based on Probability using the Second Definition of Visibility

	r_1	r_3	r_5	r_7	Σ
DEG_{r_i}	3	0	1	1	
DEG_{r_i}+1	4	1	2	2	9
$p_{r_i f_j}$	0.44	0.11	0.22	0.22	1

In both cases, once the probabilities have been calculated, one request is selected probabilistically.

Trail Definitions

The purpose of the trail within DACO is to provide information about previous construction solutions to influence future constructions. In this study, two different trails are defined, where the initial values of these trails are set to one. Moreover, evaporation and updating of these trails are discussed. The definitions of these trails are given as follows:

i) Trail between requests and frequencies (T_ARF): the key component of a solution is to decide to which frequency each request is assigned. Therefore, the most obvious trail definition is between each request and each frequency, which is also previously used in ACO for the static FAP [14]. The value of the trail indicates the quality of previous solutions when a request is assigned to a frequency.

ii) Trail between requests and requests (T_ARR): previous work on the graph colouring problem (GCP) in [12] found that a trail between nodes and nodes was more successful than a trail between nodes and colours. This is because the important aspect of a graph colouring solution is not in which colour each node is placed, as the colours are interchangeable. The important aspect is which nodes are placed together in the same colour class. When considering the static FAP, clearly the actual frequency to which each request is assigned is important. However, given the static FAP has the same underlying model as the GCP, we decided to investigate whether a trail based on which requests are assigned to the same frequencies could be advantageous.

This trail measures the success of previous solutions when requests are assigned to the same frequency using$AverageT_ARR$, which is the average trail between the prospective request and all requests already assigned to the candidate frequencyf_j, which is defined by Formula 6.

$$AverageT_ARR\{r_i\} \ = \sum_{r_j \in H, i \neq j} \frac{T_ARR\{r_i,r_j\}}{|H|+1} \qquad (6)$$

Where H is the set of requests already assigned to frequencies f_j.

Trail Evaporation

Both types of trail are evaporated after each generation by multiplying the trail by the evaporation parameter, which will be determined experimentally The trail evaporation can be defined by Formula 7.

$$\tau_{r_i f_j} \leftarrow \rho . \tau_{r_i f_j} \qquad (7)$$

Where the evaporation parameter ρ is in the range [0, 1).

Trail Updates

The trails are updated using two reward functions, namely Cost1 and Cost2, which are defined as follows:

- Cost1: Counts the number of used frequencies in the current solution. This is appropriate when a solution is feasible.
- Cost2: Counts the number of unassigned requests in the current solution. This is appropriate when a solution is infeasible.

The values of T_ARF could have been updated using Formula 8.

$$T_ARF\{r_i, f_j\} = T_ARF\{r_i, f_j\} + \frac{20}{Cost_1 + Cost_2 - Best + 1} \qquad (8)$$

Where $Best$ is the best minimum number of used frequencies found so far in the search. Note that $Cost_1 + Cost_2 - Best$ can be equal to 0 when $Cost_1 = Best$ and $Cost_2 = 0$. Thus, we add 1 to the denominator of the last term in Formula 8. A similar trail update function was previously used in ACO for the GCP [12].

Similarly, the values of T_ARR are updated using Formula 9.

$$T_ARR\{r_i, r_j\} = T_ARR\{r_i, r_j\} + \frac{20}{Cost_1 + Cost_2 - Best + 1} \qquad (9)$$

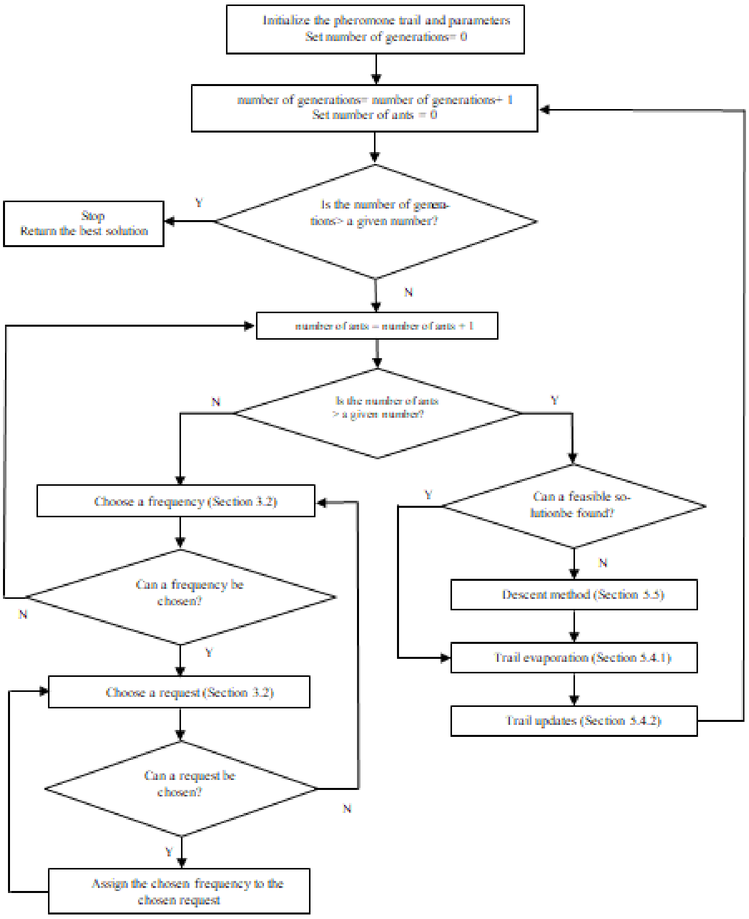
Another problem of trail updates is that only requests that have been assigned to frequencies are updated. Therefore, the trail values on any unassigned requests are not increased, meaning such requests are likely to be selected even later in the following construction processes. As we would prefer to consider them earlier in the construction process, the trail is increased on each unassigned request for all available frequencies. This idea was previously used in ACO for the examination scheduling problem [11].

Descent Method

This method is executed only when no feasible solution can be found by all ants in a generation for a sub-problem. In such generations, the descent method is executed only for one ant which constructs the infeasible solution with the minimum number of unassigned requests. First, these requests are assigned to the frequencies which lead to the least number of violations. Then, the descent method aims to reduce the number of violations with a fixed number of frequencies to find a feasible solution, if possible.

The DACO Algorithm Implementation

DACO solve each sub-problem through given number of generations, each of which contains a given number of ants, where each ant individually constructs a solution. Each ant starts constructing a solution by selecting a frequency to be assigned to all possible feasible requests. The process is repeated until no frequencies can be selected (see Section 5.2). After all ants in the current generation construct their solutions, if no feasible solution can be found, then the descent method (see Section 5.5) is used to attempt to achieve a feasible solution. Then, the trail is evaporated and updated (see Section 5.4.1 and 5.4.2). After that, the next generation is executed by the same process. The overall structure of the DACO algorithm is illustrated in Figure 2.



Initialize the pheromone trail and parameters
Set number of generations= 0
number of generations= number of generations+ 1
Set number of ants = 0
Is the number of genera-tions> a given number?
Y
Stop
Return the best solution
N
number of ants = number of ants + 1
Is the number of ants > a given number?
N
Y
Choose a frequency (Section 3.2)
Can a feasible so-lutionbe found?
Y
N
Can a frequency be chosen?
N
Descent method (Section 5.5)
Y
Trail evaporation (Section 5.4.1)
Choose a request (Section 3.2)
Trail updates (Section 5.4.2)
Can a request be chosen?
N
Y
Assign the chosen frequency to the chosen request

Figure 2: Overall Structure of Our DACO Algorithm for each Sub-Problem of the Static MO-FAP.

EXPERIMENTS AND RESULTS

This section presents and compared the performance of DACO in three sections. The first section gives the results of DACO for the static FAP. The second section compares the performance of DACO with existing ACO algorithms in the literature.

DACO is implemented in 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.

Results Comparison of the DACO Algorithm

In this study, the number of generations of DACO is 100, where this number is selected based on experiments. Moreover, the performance of DACO is compared based on several options of the following components:

- The number of ants,
- The trail definition,
- The visibility definition,
- The parameters $\alpha,\ \beta\ and\ \rho$

Different values of the number of ants, two options of the trail definition and two options for visibility definition are compared. For the parameters α, β and ρ, three values of each parameter are tested. By considering all these options, there are 756 versions of ACO to be compared. Moreover, each version is tested on 10 instances with 5 runs being performed on each instance. Therefore, considering all the versions of ACO take excessive time. Hence, the comparison is made for each component while fixing the others; i.e. first, different numbers of ants are compared while fixing the remaining components. After selecting the best number of ants, the two different trail definitions are compared. After that, two definitions of the visibility are compared and finally, different values of the parameters (α, β and ρ) are compared in the same way. Based on experiments, the best values of the parameters and number of ants given in Table 3.

Table 3: The Best Values of the Parameters and Number of Ants based on Experiments

α	β	ρ	$number\ of\ ants$
3	2	0.78	20

Moreover, the performance of DACO using $T_A RF$is better than using $T_A RR$. The performance of DACO using the two types of trail definitions is shown in Figure 3 (for the instances in which feasible solutions are found).

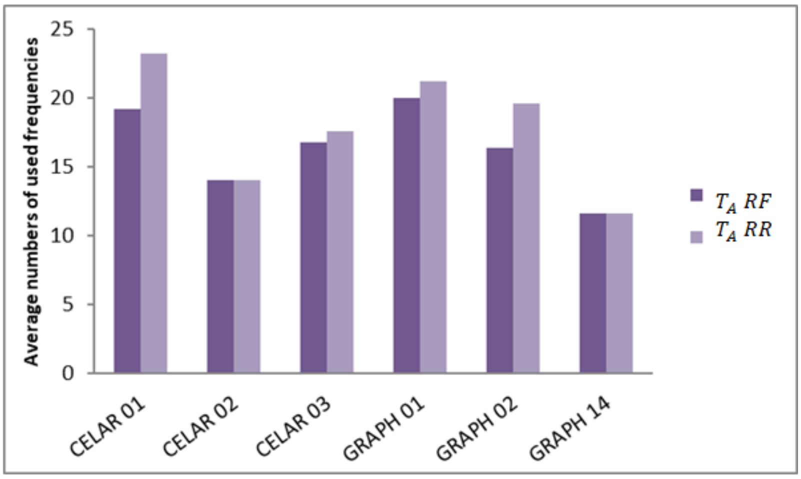


Figure 3: The Performance of DACO using Two Types of Trail Definitions.

It is found by the Wilcoxon signed-rank test at the 0.05 significance level that there is a significant difference between the performances of DACO using $T_A RF$ and$T_A RR$.

Moreover, the performance of DACO using the first definition of visibility better than the second one.

Results Comparison with Existing ACO Algorithms

The performance of our DACO is compared with existing ACO in the literature. To the best of my knowledge, only one published research [13] applied ACO for the MO-FAP using CELAR and GRAPH datasets. Table 4shows the results in the form given in [13], i.e. in the form of (y) where y is the number of violations. Note that y is equal to 0 means a feasible solution is found.

Table 4: Results of DACO and Existing ACO Algorithm in the Literature

Instance	ACO [13]	Our DACO
CELAR 01	(0)	(0)
CELAR 02	(0)	(0)
CELAR 03	(0)	(0)
CELAR 04	(8)	(0)
CELAR 11	(2)	(1)
GRAPH 01	(0)	(0)
GRAPH 02	(0)	(0)
GRAPH 08	(0)	(0)
GRAPH 09	(0)	(0)
GRAPH 14	(0)	(0)

Table 4 shows that both of the algorithms struggled to find a feasible solution for CELAR 11. Moreover, ACO in [13] could not achieve a feasible solution for CELAR 04, whereas our DACO could. Overall, our DACO algorithms performing better than ACO in [13].

CONCLUSIONS

In this study, the DACO was introduced to solve the static MO-FAP 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. Several novel and existing techniques have been used. One of the techniques was applied to improve the performance of DACO is the recursive largest first. In fact, this technique aims to improve the performance of selecting frequencies and requests to be assigned. Moreover, DACO was compared using two trail definitions and two visibility definitions. It was found that using the trail between requests and frequencies led to better performance than the other trail definition. Moreover, using the visibility definition based on the number of feasible frequencies resulted in better performance than another visibility definition. Furthermore, several values for the parameters α, β, ρ were compared.

DACO is combined with a descent method to achieve better results when no feasible solution can be found in a generation. In such generations, the descent method is executed for only one ant which constructs the infeasible solution with the minimum number of unassigned requests. Overall, our DACO algorithm performed better than ACO in the literature.

REFERENCE

1. *Metzger, B. (1970), Spectrum Management Technique, 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 Assignment Problem: A Case Study Using Genetic Algorithms, Lecture Notes on Computer Science, pp. 117-131.*
4. *Crisan, C. and Mühenbein, H. (1998), The Frequency Assignment Problem: A Look at the Performance 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 Frequency 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 Problem, Proceedings of the International Conference on Digital Signal Processing.*
10. *Costa, D. and Hertz, A., 1997. Ants can colour graphs. Journal of the Operational Research Society, 48(3), pp.295-305.*
11. *Dowsland, K.A. and Thompson, J.M., 2005. Ant colony optimization for the examination scheduling problem. Journal of the Operational Research Society, pp.426-438.*
12. *Dowsland, K.A. and Thompson, J.M., 2008. An improved ant colony optimisation heuristic for graph colouring. Discrete Applied Mathematics, 156(3), pp.313-324.*
13. *Maniezzo, V. and Carbonaro, A., 2000. An ANTS heuristic for the frequency assignment problem. Future Generation Computer Systems, 16(8), pp.927-935.*
14. *Parsapoor, M. and Bilstrup, U., 2013. Ant colony optimization for channel assignment problem in a clustered mobile ad hoc network. In Advances in Swarm Intelligence, pp. 314-322. Springer Berlin Heidelberg.*
15. *Leighton, F.T., 1979. A graph coloring algorithm for large scheduling problems. Journal of Research of the National Bureau of Standards, 84(6), pp.489-506.*
16. *Hale, W.K., Frequency assignment: Theory and applications. Proceedings of the IEEE, 1980. 68(12): p. 1497-1514.*