Problem Decomposition for Minimum Interference Frequency Assignment

G. Colombo and S.M. Allen
School of Computer Science, Cardiff University, Cardiff, UK E-mail: g.colombo@cs.cf.ac.uk

***Abstract*— This paper applies a problem decomposition approach in order to solve hard Frequency Assignment Problem instances with standard meta-heuristics. The proposed technique aims to divide the initial problem into a number of easier subproblems, which can then be solved either independently or in sequence respecting the constraints between them. Finally, partial subproblems solutions are recomposed into a solution of the original problem. Our results focus on the COST-259 MI-FAP instances, for which some good assignments produced by local search meta-heuristics are widely available. However, standard implementations do not usually produce the best performance and, in particular, no good results have been previously obtained using evolutionary techniques. We show that problem decomposition can improve standard heuristics, both in terms of solution quality and runtime. Furthermore, genetic algorithms seem to benefit more from this approach, showing a higher percentage improvement, therefore reducing the gap with other local search methods.**

I. Introduction: the fixed frequency assignment problem

In a wireless network transmitters and receivers communicate via signals encoded on specific frequency channels. When adjacent transmitters use similar channels they may cause unacceptable interference. Thus a channel separation based on some interference measure is required for transmitters which are geographically close. The *fixed frequency assignment problem* (FAP) is an optimization problem which assigns frequencies to transmitters in as efficient way as possible, either in terms of interference (*fixed spectrum* FS-FAP), or range of frequencies used (*minimum span* MS-FAP). The FAP can be expressed in graphic theoretical terms and is a known NP-hard problem, which can be solved exactly only for problems with a limited number of transmitters. An updated classification of the different methods, models and formulations that the literature provides on the topic can be found in [4]. For large problems several meta-heuristic techniques have been proposed in the literature to produce near-optimal solutions in a reasonable runtime, although their performance tends to degrade with the size or the complexity of the problem [13]. To remedy this problem we propose that a decomposition approach is applied.

The aim of this work is to show that problem decomposition can be used in order to improve the effectiveness of meta-heuristics on solving large or complex benchmark test-problems.

In particular, we consider the well-studied Siemens *minimum interference* MI-FAP (a version of the FS-FAP) instances of the COST-259 benchmarks [9]. The rest of the paper is organized as follows. The next section formally describes the MI-FAP problem and presents a survey of the application of decomposition techniques to the FAP. Section III gives a description of the decomposition algorithm we propose and its implementation. Section IV describes in detail the proposed decomposition approach in combination with the meta-heuristics considered while Section V shows results for the MI-FAP in comparison with other decomposition algorithms. Finally, Section VI outlines future work and concludes the paper.

II. Problem decomposition for FAP

In the rest of the paper we will focus on the MI-FAP, which involves the presence of *hard constraints*, which must be satisfied by the optimal solution, and *soft constraints*, assigned as penalties which constitute the actual interference to be minimised. Formally, this type of FAP can be modelled by a unordered weighted graph $G(V, E)$, called the interference graph, which consists of a finite set of vertices V, representing transmitters, and a finite set of edges $E \subseteq \{ \{u, v\} | u, v \in V\}$ joining unordered distinct pairs of vertices. Each edge e has an associated weight vector $c_e = \{c_e^{hard}, c_e^{coch}, c_e^{adj}\}$. Given an allocation of allowed channels $\{1, 2, \ldots, K\}$, the aim of the MI-FAP is to produce an assignment $f : V \rightarrow \{1, 2, \ldots, K\}$ that minimises:

$$c_G(f) = \sum_{e \in E(G)} c(f, e)$$

$$c(f,e) = \begin{cases} 1000 & if \ |f(u) - f(v)| \leq c_e^{hard} \\ c_e^{adj} & if \ |f(u) - f(v)| = 1 \\ c_e^{coch} & if \ |f(u) - f(v)| = 0 \\ 0 & otherwise \end{cases} \quad (1)$$

The penalties on the edges are defined as follows.

c_e^{hard} are hard constraints which represent the channel separation necessary to avoid undesirable interference. They are associated with very high values, usually in the order of thousands, in order to be necessarily satisfied by the heuristic search. A solution which violates any of these constraints is than classified as an invalid, or infeasible, solution.

c_e^{coch} and c_e^{adj} are soft constraints defined in terms of penalty factors which may be violated by feasible solutions. Their values represent a probabilistic measure of interference between pair of transmitters that use respectively the same (*co-channel* interference) or adjacent transmission channels (*adjacent-channel* interference).

1-4244-1340-0/07/$25.00 ©2007 IEEE

Although in many combinatorial approaches, the *decomposition-recombination* planning problem (DR) is a current object of research, it has been scarcely applied to meta-heuristics to solve the FAP. In fact, as described in the next subsection, decomposition techniques have been primarily used for this problem in combination with exact optimization methods, such as branch-and-cut and combinatorial enumeration. Moreover, they have been essentially applied to simpler formulations of the FAP or to produce lower bounds rather than full solutions.

A. Exact solution methods

In Mannino and Sassano [19] the idea of firstly solving a partial problem restricted to a hard subgraph (called the *core*) is applied to an enumerative algorithm for some *minimum blocking* MB-FAP instances. In Aardal et al. [3] preprocessing based on cliques in the interference graph is used with a branch-and-cut algorithm to reduce the size of *minimum order* MO-FAP problems by usually ten to fifteen percent. Similarly, a clique bound has been extended and generalised to provide lower bounds for both the MO-FAP and MS-FAP.

However, for the more difficult MI-FAP exact methods have been essentially used to produce lower bounds. Koster et al. [17] observed that assigning frequencies to a cut-set of the interference graph decomposes the problem into two or more independent subproblems. Hence, they generated a sequence of such cut-sets by using a tree decomposition. This idea, in addition to the use of further several dominance and bounding techniques led to the solution of some small and medium size instances. For larger real-life instances, in which the dynamic programming algorithm proposed is impractical because of the width of the tree decomposition, the algorithm has been used iteratively to improve some known lower bounds. Finally, Eisenblatter [12] derived new lower bounds for the COST 259 MI-FAP instances by studying the semidefinite programming relaxation of the minimum k-partition problem. These bounds are based on the fact that the MI-FAP reduces to a minimum k-partition problem, which can be modeled as a semidefinite program, with the restriction of considering only co-channel interference.

B. Heuristics solving the FAP

Due to its difficulty, the majority of work on the FAP has proposed heuristic approaches; the adoption of decomposition techniques in these cases has been very limited. Moreover, they have been mainly combined with exact methods in order to optimize solutions locally inside system clusters of several cells, in the same fashion used for distributed channel assignment. In Montemanni et al. [20] a cell reoptimization is used in combination with a *tabu search* (TS) after a fixed number of iterations. Each cell is selected in sequential order and its assignment is optimized by an exact method, while those in the other cells are kept fixed. Mannino et al. [18] proposed a *simulated annealing* (SA) combined with dynamic programming to compute local optima. In their approach, they optimize assignments in cliques of vertices of multiple demand by reducing this problem to finding fixed cardinality stable sets in interval graphs. In both situations the local optimization procedure obtains very good results on some of the COST 259 instances, although this increases the complexity of the original heuristics used.

The same strategy can be extended to a larger scale by adopting a different approach in which a large problem is partitioned a priori into subsets thus generating a number of subproblems. Each of them is then solved by a meta-heuristic, either separately or in sequence respecting the constraints between different subsets. The resulting solutions are recomposed into a solution of the whole initial problem. This approach, which does not involve any exact local optimization algorithm, has been very seldom used in the literature. In [13] a decomposition based on cliques was used with SA to produce lower bounds and solutions for MS-FAP. Firstly, the procedure finds the maximum clique of the interference graph. The corresponding induced subgraph constitutes a sub-problem which is then solved by SA. Finally, the assignment is extended to one of the whole interference graph while keeping the assignment of the clique fixed. Alternatively, the extended assignment can be found iteratively by adding, one at a time, a selected set of vertices to the subgraph currently considered. In Colombo [8] an order-based *genetic algorithm* GA has been combined with two different decompositions, based on either the generalized degree of the corresponding graph or a more sophisticated graph partitioning algorithm, to solve both the MS-FAP and simple instances of the FS-FAP. The procedure starts by partitioning the interference graph into one or more subsets. The GA is applied to each of the subsets in turn to produce a sequence of partial frequency assignments. When the current subset is considered the algorithm keeps fixed the assignment of transmitters in the previously assigned subsets, and the current assignment aims to minimise the constraint violations within them. When all the subsets have been assigned, the algorithm returns a final assignment of the original whole problem.

Finally, a slightly different approach has been recently used in [15]. Here a clustering algorithm based on the generalised degree of the neighbour vertices is used to initially partition a fixed spectrum FAP instance, which is then solved by a GA. However, the representation used has the limitation of only considering co-channel interference.

III. Our approach: graph partitioning problem

As mentioned before, in this work we refer to the approach presented in [13], [8], and investigate its actual effectiveness on the harder COST-259 benchmarks. In particular we adopt a decomposition based on the *graph partitioning problem*, which extends that used in [8]. This is formally described in the rest of the section.

A. Graph partitioning for FAP (GPFAP)

In a simple unordered graph the *graph partitioning problem* is defined as dividing the vertices into disjoint subsets such that the number of edges whose endpoints are in

different subsets is minimized. Formally, given a collection of n disjoint subsets:

$$\{ V_1, V_2, \ldots, V_n \} : \bigcap_{i=1}^{n} V_i = 0 \wedge \bigcup_{i=1}^{n} V_i = V$$

we then define a set $E_n = \bigcup_{i=1}^{n} E_i \subseteq E$:

$$E_i = \{u, v\} \in E | u \in V_i, v \in V_j, \ 1 \leq j \leq n, \ j \neq i \quad (2)$$

The graph partitioning problem consists of minimizing the cardinality $\mid E_n \mid$. If we consider the *balanced graph partitioning* we have the additional constraint that the difference between the cardinality of different subsets is as small as possible, i.e either one or zero.

To adapt the graph partitioning to the MI-FAP data sets we will refer to the graph theoretical model introduced in the previous section. In addition we reduce the corresponding interference graph to a new unordered weighted graph whose edge weights are defined as a weighed combination of soft and hard constraints according to the expression below:

$$c_e = max\{\lambda_1 c_e^{coch} + \lambda_2 c_e^{adj}, \lambda_3 c_e^{hard}\} \quad (3)$$

in which λ_i are assigned weights to reflect their importance.

Consequently, the GPFAP can be formulised as minimizing the sum of the edge weights for all the edges whose endpoints are in different subsets. This objective was used in the partitioning in [8]. It is worth noticing that, however, the sum of the inter- connections between different subsets in a given partition is not the only important index of its effectiveness when it is subsequently used to solve FAP instances. In fact, other factors can be equally important, such as the number or sum of weights of the internal edges in the first subset (or the first group of subsets) in order to disallow the trivial solution of finding a partial assignment which does not present any interference. That is, we would like the first subset to contain the most "difficult" to assign transmitters, since the assignment of the first subset is not constrained by any fixed transmitters. This is the idea behind the clique and generalised degree decompositions used respectively in [13] and [8]. As a consequence, to solve the *balanced* GPFAP, we have defined the following objective:

$$O_{bal} = \sum_{e \in E(G)} c_e - \sum_{e \in I_1} c_e + \sum_{i=1}^{n} \sum_{e \in E_i} c_e \quad (4)$$

In (4)

$$I_i = \{ \{u, v\} \in E | u, v \in V_i, \ 1 \leq i \leq n\} \quad (5)$$

are the internal edges of the ith subset. Hence, the corresponding objective of optimization becomes minimizing the sum of the edge weights for all the edges whose endpoints are in different subsets while maximizing the internal connections in the first subset only. In addition, the objective in (4) can be normalized to the total sum of the edge weights for computational convenience: $O_{bal} := \frac{O_{bal}}{\sum_{e \in E(G)} c_e}$

We can also formulate an *unbalanced GPFAP* which removes the balanced constraint. For this problem better performance is obtained if we aim to minimise the external connections between pairs of different subsets while maximising the internal connections in each of the subsets. In fact, this encourages the size of each subset to be relatively balanced in order to prevent the tendency of including a very high percentage of transmitters in one subset only. For this purpose, we defined a different objective based on the analogy between graph partitioning and graph clustering. Cluster analysis is closely related to graph partitioning with the main difference being that in the latter the required partition sizes are specified in advance, whereas in the former the fact that the partition freely varies constitutes a very complicating factor. Other differences introduced in some graph clustering algorithms are the notions of overlapping and hierarchy. However, the effectiveness of a cluster strictly depends on some parameters, which are usually closely related to the number of clusters produced. As a consequence, clustering is quite often used only as a preprocessing step for the more clearly defined graph partitioning problem. A natural definition of graph clustering is the separation of sparsely connected dense subgraphs from each other. If we restrict the attention to only two subgraphs with variable size we can identify the simple objective used in [8] as the *minimum cut* of the graph. Given a cut $c(G) = (V_1, V \backslash V_1)$ and using the notation in (2), we can also define a *minimum quotient cut* problem consisting in minimizing:

$$\frac{\sum_{e \in E_1} c_e}{min(|V_1|, |V \backslash V_1|)} \quad (6)$$

The clustering problem in a weighted structured graph can then be seen as a mixture of the minimum cut and the minimum quotient cut problems [22].

The expression in (6) is also called the *expansion* of a cut. We can also define another index, very similar to the expansion, known as the *conductance* of a cut c, which, using the notations in (2), (5), is formulated as:

$$\phi(c) = \frac{\sum_{e \in E_1} c_e}{min(\sum_{e \in (E_1 \cup I_1)} c_e, \sum_{e \in (E \backslash I_1)} c_e)} \quad (7)$$

If a cut has a small conductance it means that its size is small relative to the density of either sides of the cuts, so such a cut can be seen as a bottleneck of the graph. The conductance of a graph $\phi(G)$ is the minimum conductance value over all cuts of G . Calculating the conductance of a graph is computationally expensive, usually raising to NP-hard problems [6].

Given a partition in n subsets $\{ V_1, V_2, \ldots, V_n \}$ we can then refer to the partition as a clustering $C(G)$ of the graph G and the V_i clusters. We can also identify the cluster V_i with the induced subgraph of G:

$$G[V_i] := (V_i, I_i)$$

in which I_i are the internal edges defined in (5). We call $\bigcup_{i=1}^{n} I_i = I$ the set of *intra-cluster* edges whereas $E_n = E \backslash I$

are the *inter-cluster* edges already defined in (2). We then define the *intra-cluster conductance* $alpha(C)$ of a a clustering $C(G)$ as the minimum conductance value over all induced subgraphs $G[V_i]$, while the *inter-cluster conductance* $delta(C)$ is the maximum conductance value over all induced cuts $c_i = (V_i, V \backslash V_i)$:

$$\alpha(C) = min(\phi(G[V_i])) \ and \ \delta(C) = 1 - max(\phi(c_i))$$

The larger the *intra-cluster conductance* of a clustering, the higher the quality, since small intra-cluster conductance means that there is at least one of the clusters V_i containing a bottleneck which can be then further decomposed into two subsets. A clustering with small inter-cluster conductance is also a low-quality one since there is at least a cluster with strong external connections. Optimising the indexes above is NP-hard as reported in [6]. However, given a clustering we can easily compute the conductance $\phi(c_i)$ over all induced cuts of the clustering. Since we want to minimize these values for all the cuts in a cluster we have then defined the following objective to solve the unbalanced GPFAP: $O_{unbal} = \sum_i \phi(c_i)$

$$O_{unbal} = \sum_i \frac{\sum_{e \in E_i} c_e}{min(\sum_{e \in (E_i \cup I_i)} c_e, \sum_{e \in (E \backslash I_i)} c_e)} \quad (8)$$

Finally, both objectives in (4) and (8) can be used for both types of GPFAP, although in our experiments we used those performing better on some test runs.

B. A memetic GA for GPFAP

To solve the GPFAP in all the formulations introduced in the previous subsection we have implemented a memetic GA. The aim is to obtain near optimal solutions in a reasonably short time rather than pursue the absolute optimal, since the partitioning only constitutes the preprocessing step of the the subsequent procedure which solves the FAP. We have implemented a standard version of SEAMO, originally proposed by Mumford, which, with the *order-based* representation here adopted, has been shown to be very effective in producing good nearly optimal solutions in a short time on some known test-bed problems [7].

Individuals are represented by a permutation of the transmitter set, and fitness evaluation consists of decomposing a given ordering into M subsets and then computing one of the objectives defined above. However, the actual implementation acts on single cells rather than single transmitters, in which each cell contains the transmitters belonging to the same site. This has been found to produce much better results since it automatically preserves constraints within cells. The number of subsets is an input parameter whereas the size of each subset in the decomposition can be either fixed (as $\frac{noCells}{noSubsets}$) or randomly chosen, if we want to remove the balanced constraint. In our experiments we used the same parameters' setting proposed in [8], i.e. a population of $N = 100$ individuals run for $G = 500$ generations. *Cycle crossover* was applied with a rate of 100% whereas one single *order-based mutation* was performed for each of the offspring generated. Finally fitness sharing was used as a niching method with the same implementation used in [8] in which this technique has produced very effective results. Similarly, the setting of the weights used was $\lambda_1 = \lambda_2 = 0.5$, and $\lambda_3 = 1$. Although it is not necessary to solve the GPFAP exactly, it has been found that better approximations in the decomposition lead to better FAP solutions. Consequently, to speed up the process and to improve solutions quality too, the GA has been hybridised with a local search procedure after each individual evaluation, in order to search for local optimality. We have implemented a standard SA, as described in [13], in which each move consists, as well as for the mutation operator for the GA, in a single cell swap between two different subsets. The SA was run for $I = 1000$ iterations for each of the offspring produced. Figure 1 gives the pseudocode of the memetic GA and Figure 2 gives a visual representation of a decomposition on a benchmark data set.

Procedure *Memetic GA for GPFAP*
Input: PopSize *N*, noSubsets *M*, noGen. *G*, noIt. *I*
Output Partitioning in *M* subsets
1: Generate a population of N individuals as permutations of the whole set of transmitters\cells representing a chromosome
2: Decompose each ordering in M subsets with either fixed or randomly variable size
3: Evaluate the fitness for each individual by using one of the objectives (4)and (8)
4: Store the *bestSoFar* fitness value
5: **while** Stopping condition not satisfied **do**
6: **while** next individual in the population **do**
7: This individual becomes the first parent
8: Select a second parent by applying roulette wheel selection
9: Apply cycle crossover to produce offspring
10: Apply order-based mutation to offspring
11: Evaluate fitness of offspring
12: Add fitness sharing to each of the offspring
13: Apply SA for I iterations to improve local optimality
14: **if** offspring better than either parent **then**
15: Replace the weakest parent
16: **else**
17: Randomly select another individual in the population and replace it if it is weaker
18: **end if**
19: Update *bestSoFar*
20: **end while**
21: **end while**
22: Return the decomposition representing the *bestSoFar* individual

Fig. 1. Pseudocode of the 'memetic' GA to solve the Graph Partitioning problem.

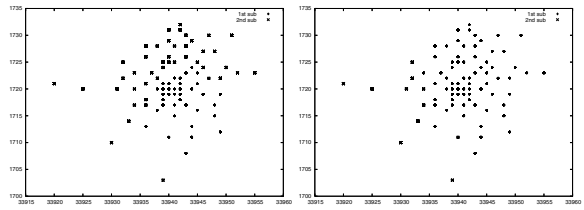

Fig. 2. Balanced (left) and Unbalanced (right) GPFAP decomposition for the Siemens2 COST-259 data set

IV. A DECOMPOSITION APPROACH FOR MI-FAP

Once a suitable decomposition into n subsets $\{ V_1, V_2, \ldots, V_n \}$ has been determined, the corresponding subproblems are solved in turn by applying a chosen meta-heuristic. For the ith subproblem the interference graph is then represented by the induced subgraph $G[V_i]$, and the assignment produced for a subset is considered fixed for the rest of the procedure. When the current subset is considered, the heuristic algorithm computes the cost of constraints involving the transmitters belonging to the subsets already assigned. This choice has been found as essential in order to produce feasible solutions to MI-FAP instances, since the presence of hard-constraints prevents the possibility of solving each of the subproblems independently. When all of the subsets have been examined the procedure returns a final assignment for all the transmitters included in the original data set. To improve the quality, the procedure can be repeated, with the meta-heuristic looping over the subsets more than once. Again, frequencies assigned to subsets other than the current one are kept fixed. In this work the chosen meta-heuristics were a standard SA implementation and a GA with *direct* representation. Figure 3 gives the pseudocode of the decomposition algorithm described above.

Procedure *Decomposed solution*
Input: Partition $\{ V_1, V_2, \ldots, V_n \}$ of G, $numLoops$
Output FrequencyAssignment f of G
1: **for** $i = 1$ to $numLoops$ **do**
2: **for** $j = 1$ to n **do**
3: Apply meta-heuristic to determine $f(v)\, \forall\, v \in V_j$ to minimise $c_G(f)$ (i.e. keep $f(u)$ fixed $\forall\, u \notin V_j$)
4: **end for**
5: **end for**

Fig. 3. Pseudocode of the decomposition algorithm

Simulated annealing has been successfully applied to the MI-FAP test problems considered even in its simplest implementations [1]. We have here implemented the standard implementation proposed in [13], in which each move corresponds to a frequency change to a randomly chosen transmitter. Frequencies are also chosen at random within the corresponding transmitter domains. Figure 4 gives the pseudocode of the SA implementation used in this work. Besides the standard SA we have applied the same decomposition approach to a genetic algorithm. However, the order-based GA which produced good results for some FS-FAP instances is unsatisfactory when solving the harder MI-FAP instances (for the reasons described in [8]). As a consequence the GA implemented for this work adopts the more straightforward *direct* representation, in which a chromosome is represented by a vector s whose components s_j simply represents the frequency assigned to transmitter v_j, in the same fashion used for SA described before. Here, the main difficulty arises from the choice of effective genetic operators since the standard ones produce poor performance. However, the fully exploration of the objective space is always permitted. Therefore some new problem specific operators have been suggested, although the results produced

Procedure *SA implementation for FAP*
Input: initialTemperature *to*, noIterations $numLoop$ reductionIndex α, interference Graph $G[V_i]$
Output FrequencyAssignment *X*
1: Initialize the temperature $t \leftarrow to$
2: Generate a random assignment X_{old} of the set of transmitters
3: Evaluate the cost C_{old} of X_{old}.
4: **while** $t > t_{min}$ **do**
5: **for** $i = 1$ to $numLoop$ **do**
6: Generate a new configuration X_{new} by changing the freq. of a randomly chosen transmitter. Frequencies are chosen at random within the transmitter domains.
7: Calculate the new cost C_{new}
8: Calculate $\Delta\, C = C_{new} - C_{old}$
9: **if** $\Delta\, C < 0$ *or* $random < prob = e^{-\frac{C_{new}-C_{old}}{kT}}$ **then**
10: $X_{old} \leftarrow X_{new}$
11: $C_{old} \leftarrow C_{new}$
12: **end if**
13: **end for**
14: $reduce\, t\ (e.g.\ t = \alpha t)$
15: **end while**
16: Return the final assignment $X \leftarrow X_{old}$

Fig. 4. Pseudocode of the SA implementation for the FAP

on some known benchmark are not uniform [4]. In particular no good results have been previously published for the MI-FAP Cost-259 instances considered in this paper. In our implementation crossover and mutation rates were chosen after some test runs on the same group of data sets, performed without any decomposition applied. The crossover used was a variation of those proposed in [14], [10]. Its procedure can be outlined in the following steps:

1: Find a pair of transmitters u and v for which the constraint between them is satisfied, i.e $c(f, uv) = 0$ for any of the two currently selected parents
2: If no pair can be found in a fixed number of selections select only one transmitter which has no constraint violation in any of the currently selected parents.
3: Interchange the frequencies assigned to u and v and all the vertices belonging to their common neighborhood in the first parent with those assigned in the second parent.

The mutation used, as proposed in [14], consists of a number of simple frequency swaps between pairs of transmitters selected at random, according to a given mutation rate. Furthemore, to improve the GA performance an iterative 1-opt procedure (see [12]) has been added after offspring generations to search for local optimality. Note that a 1-opt procedure used as mutation operators produced good results in [16] for another group of MI-FAP benchmarks. The local search procedure (LS) implemented in our GA is outlined in the figure below. Finally, the experiments have been run as *multi-objective optimization*, a novel approach which can be seen as alternative to the introduction of penalty factors suggested in [14]. In particular, we have considered two different objectives constituted by the two different types of of constraint violations, which yield respectively to *co-channel* and *adjacent-channel* interference. Note that the LS 1-opt procedure outlined above operates on a global objective computed as the sum of each of the objectives of optimization. The framework for the multi-objective GA

Procedure *Iterative 1-opt procedure for FAP*
Input: FrequencyAssignment X_{old}
Output FrequencyAssignment X
1: Select a random permutation ord of the transmitters
2: **while** no more cost improvements **do**
3: **while** next transmitter in ord **do**
4: Reassign transmitters sequentially to the best frequency in the domain according to the ordering ord.
5: Update assignment X_{old}
6: **end while**
7: Select a new random permutation ord of the transmitters
8: **end while**
9: Return the final assignment $X \leftarrow X_{old}$

Fig. 5. Iterative 1-OPT implementation for the FAP

TABLE I
CHARACTERISTICS OF THE SIEMENS COST-259 DATA SETS

	cardinality	density %	avg. degree	max. clique	SA [5]
Siemens1	930	9.03	84.0	52	2.78
Siemens2	977	49.17	480.4	182	15.46
Siemens3	1623	9.18	149.1	78	6.75
Siemens4	2785	10.50	292.3	100	89.15

implemented for this purpose was a standard implementation of NGSAII proposed by Deb et al in [11]. A detailed description can be found in [11]. The only change introduced was the addition of the LS 1-opt procedure at the end of each generation to search for local optimality. The multi-objective choice presents the advantage of reducing considerably the problem of *premature convergence*, which was one of the main problems of the single objective approach, despite losing something in terms of speed. Moreover, NGSAII (a generational GA) was preferred to SEAMO (a steady-state GA) used in [8] because it appears able to produce a better spread of the *Pareto non-dominated set* [7].

V. MI-FAP RESULTS

The proposed decomposition approach has been tested for the two meta-heuristics described in the previous section on a subset of the widely available COST-259 MI-FAP benchmarks [9], namely the Siemens data sets for GSM 900 networks (see Table I), whose results for a restricted number of meta-heuristics is available from [1]. A complete summary table of their characteristics can be found in [12]. Table I also shows the best known published results obtained by a standard SA [5]. The parameters used for the meta-heuristics are summarised in the following and were set after some test runs performed without considering any decomposition. For SA we used an initial temperature $t_0 = 0.5$ and a number of iterations $numLoop$ equal to the number of transmitters in the current subset. The reduction index α was calculated in order to satisfy the total number of evaluations required, i.e. the configurations explored as a multiple of the total number of transmitters N. For the GA we used a population of 20 chromosomes and a rate of 80% for Crossover, 0.06% for mutation, and 90% for 1opt LS. The number of generations depended again on the total number of evaluations required, in order to compare the two algorithms for the same number of evaluations. The runtimes in all figures are based on a 3.000 GHz Intel Pentium 4.

A. Comparison of decomposition techniques

This section presents the results obtained for Siemens1-4 in order to compare the balanced and unbalanced graph partitioning, along with generalized degree decomposition and cliques. Graph partitioning was implemented using the algorithm in (6) with respectively the costs in (4) and (8) for the balanced and unbalanced decomposition. The generalized-degree and cliques decomposition are included as they have been shown able to obtain very good results for the MS-FAP (see [8], [13]). To find the maximum clique we implemented the weighted version of the algorithm proposed by Pardalos et al. in [21]. In order to define the problem with more than two subsets, the cliques were found sequentially after removing the subgraphs induced by those already found. The generalized-degree decomposition simply starts by ordering all the set of transmitters by their generalized-degree and then includes in each subset the transmitters which produce almost equal values of sums calculated under the graph generalized-degree versus transmitters. Tables II and III show the best and average over three runs (in brackets) cost obtained by SA after 100,000*N evaluations. The algorithm loops over the given partitioning twice.

The GPFAP appears superior to the cliques and generalised-degree decompositions, which for the hardest instances present difficulties in producing valid solutions. In particular clique based decomposition often found an interference free solution for the first subset. In fact, these small cliques are easy to assign with low cost, hence fixing their assignment unnecessarily restricts the assignment of subsequent subset. Note the effect of the second loop and how it is beneficial both in terms of mean and variance. In physical terms this is related to a locally different interference distribution in the subsets. For Siemens1, 3, and 4 the GPFAP technique performs better the traditional way of solving the original problem as a whole.

B. Results for balanced GPFAP decomposition

Tables IV and V show the best and average cost over three runs obtained by SA and the GA for a given number of 2,000,000*N evaluations. Using the parameters described earlier, each of the runs for both of the algorithms (either with or without decomposition) explores the same number of configurations. In both cases, for the 1st, 3rd and 4th problem instances the decomposition technique performs better than the normal approach shown in Table IV. Hence, problem decomposition can be used as a valid alternative to the addition of complicated local optimization procedures to standard meta-heuristics implementation. Furthermore, our approach improves considerably the performance of the GA. In fact its results for the problem solved considering all transmitters in one go are considerably worse than SA,

TABLE II
SA WITHOUT DECOMPOSITION - 100,000*N EVALUATIONS

Siem1	Siem2	Siem3	Siem4
3.38 (3.44)	16.75 (16.89)	8.14 (8.32)	91.43 (91.71)

TABLE III

SIEM1-4 FOR SA WITH DECOMPOSITION - 100,000*N EVALUATIONS

† at least 1 invalid solution * no valid solutions

ns	Bal. GPFAP	Unbal. GPFAP	Gen. Degree	Cliques
	first loop			
	second loop			
SIEMENS 1				
2	3.16 (3.28)	3.38 (3.44)	5.46 (5.72)	4,57 (4.64)
	3.14 (3.18)	3.30 (3.37)	4.24 (4.31)	3.49 (3.58)
3	3.36 (3.45)	3.35 (3.43)	5.89 (6.23)	3.51 (3.58)
	3.32 (3.36)	3.29 (3.34)	4.99 (5.05)	3.42 (3.52)
4	3.53 (3.58)	3.99 (4.20)	6.85 (6.99)	3.59 (3.67)
	3.52 (3.53)	3.30 (3.44)	5.35 (5.56)	3.58 (3.68)
SIEMENS 2				
2	17.91 (17.98)	18.24 (18.62)	23.30 (24.37)	18.38 (20.02)
	17.83 (17.91)	17.59 (17.86)	18.95 (19.58)	18.22 (18.42)
3	19.91 (20.23)	19.79 (19.94)	26.14 (26.26)	19.59 (2,018) †
	18.30 (18.57)	17.99 (18.09)	21.26 (21.36)	19.35 (19.44)
4	20.99 (21.10)	17.72 (17.93)	27.47 (27.52)	19.82 (2,686) †
	18.60 (18.87)	**17.31 (17.42)**	22.61 (22.76)	19.18 (19.33)
SIEMENS 3				
2	7.66 (7.84)	7.59 (7.69)	2,009 (3,350) *	9.14 (674.8) †
	7.58 (7.81)	7.31 (7.49)	9.15 (1,342) †	8.76 (8.82)
3	7.30 (7.45)	7.82 (7.99)	2,016 (3,646) *	8.48 (675.3) †
	7.24 (7.40)	7.79 (7.98)	11.30 (678.0) †	8.03 (8.27)
4	8.09 (8.28)	8.43 (8.57)	3,017 (3,684) *	8.61 (675.6) †
	7.98 (8.14)	7.92 (8.13)	12.40 (1,345) †	8.43 (8.48)
SIEMENS 4				
2	90.65 (91.50)	90.62 (90.88)	2,147 (3,145) *	93.04 (723.3) †
	90.65 (91.48)	89.54 (90.09)	2,120 (3,082) *	92.62 (93.75)
3	94.13 (94.85)	88.63 (89.12)	3,553 (3,823) *	2,254 (2,371) *
	92.88 (93.56)	**88.23 (88.43)**	3,131 (3,514) *	93.92 (95.54)
4	94.72 (95.68)	92.40 (93.63)	3,749 (3,983) *	2,651 (3,001) *
	93.19 (93.38)	92.07 (93.42)	3,535 (3,801) *	96.39 (97.46)

whereas the decomposition techniques tends to close the gap between the two algorithms (with GA even producing the best absolute cost for Siemens3). For Siemens2 the decomposition approach does not produce any quality improvement. However, this instance presents a particularly high connectivity of the associated interference graph (see Table I). Figure 6 gives an example of how the decomposition improves the runtime of the algorithms, even when they are run for the same number of evaluations, with and without decomposition in subsets. This is due to the smaller size of subsets, which requires a shorter time to produce partial incomplete assignments in the first subsets. It can be noticed that, in percentage, the GA benefits more in runtime terms.

As for the shorter runs presented in the previous subsection, the unbalanced version of the GPFAP partitioning can sometimes produce better results. Note that the first subset runs for a longer time, since in this partitioning it usually includes a very high percentage of the whole transmitters. As a consequence the quality gain often happens despite losing some of the advantage in runtime. Moreover, although for reasons of space we do not present its results in this context, unbalanced GPFAP improves the performance especially when decomposition is not effective (see Table III for the shorter runs of Siemens2). An example of unbalanced GPFAP is shown in Figure 7.

Finally, for Siemens2, and in general when the partitioning does not improves the results quality, it can be seen how the GPFAP approach is still able to provide acceptable approximations in a shorter time. Note that for very large and complex problems good approximations of the solutions are more than acceptable, since it is impossible to find the optimal in a reasonable time. This is shown in Figure 8 for the GA, which, as already mentioned, generally shows more advantage than SA in terms of runtime.

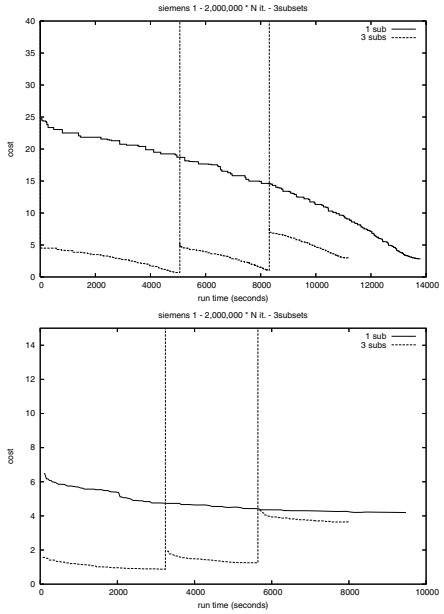

Fig. 6. Cost-time plot for Siemens1 solved by SA (top) and GA (bottom) with balanced GPFAP decomposition in 3 subsets after 2,000,000*N evaluations

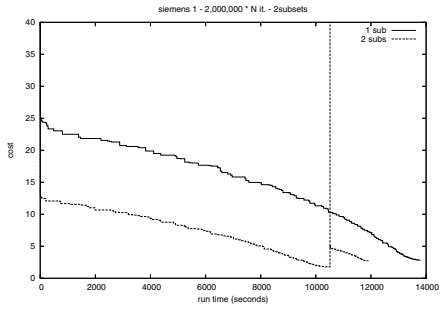

Fig. 7. Cost-time for Siemens1 solved by SA with unbalanced GPFAP decomposition in 2 subs. after 2,000,000*N eval.

VI. FUTURE WORK AND CONCLUSION

This paper proposed a problem decomposition approach for solving the FAP with meta-heuristics. Results show that problem decomposition can be used as a strategy to improve algorithm performance. However, this approach appears to be successful when the connectivity of the interference graph is not too high (although this is not the only important index to predict the effectiveness of a decomposition).

TABLE IV

SIEM1-4 WITHOUT DECOMPOSITION - 2,000,000*N EVALUATIONS

	Siem1	Siem2	Siem3	Siem4
SA	2.75(2.83)	15.59(15.79)	6.61(6.72)	85.59 (86.25)
GA	4.02(4.13)	18.00(18.14)	8.54(8.77)	105.69 (106.42)

TABLE V

SIEM1-4 FOR SA (TOP) AND GA (BOTTOM) WITH BALANCED GPFAP DECOMPOSITION - 2,000,000*N EVALUATIONS

noSub.	Siem1	Siem2	Siem3	Siem4
2	2.68 (2.75)	16.97 (17.04)	6.39 (6.46)	84.35 (84.80)
	2.60 (2.69)	**16.34 (16.73)**	**6.37 (6.44)**	**84.08 (84.39)**
3	2.95 (3.01)	19.54 (19.77)	6.53 (6.81)	89.50 (91.47)
	2.93 (2.95)	17.41 (17.60)	6.46 (6.58)	87.16 (89.51)
4	2.94 (3.01)	20.56 (20.87)	6.95 (7.02)	89.96 (90.71)
	2.90(2.98)	18.02 (18.27)	6.79 (6.89)	89.53 (90.02)

ns	Siem1	Siem2	Siem3	Siem4
2	3.53 (3.57)	18.88 (19.03)	6.11 (8.65)	98.31 (99.93)
	3.18 (3.27)	**17.83 (17.86)**	**6.08** (8.21)	**96.84 (97.22)**
3	3.31 (3.52)	20.70 (20.95)	6.46 (7.90)	103.61 (104.40)
	3.07 (3.14)	18.70 (18.89)	6.38 (7.63)	101.09 (101.93)
4	3.46 (3.49)	21.50 (21.79)	6.74 (6.84)	102.55 (103.60)
	2.96 (3.04)	19.10 (19.21)	6.28 (**6.48**)	101.84 (102.19)

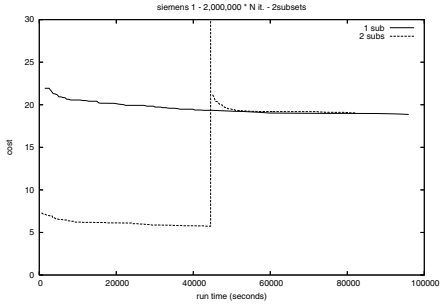

Fig. 8. Cost-time for Siemens2 solved by GA with balanced GPFAP decomposition in 2 subsets after 2,000,000*N evaluations

We have considered the Siemens COST-259 benchmarks problems. These instances represent fairly large and heavily connected real-life problems, which present further difficulties due to the contemporaneous presence of hard and soft constraints. As a consequence the performance of standard heuristics is insufficient for solving the corresponding MI-FAP problems. In particular, no good results have been previously published for these instances using any evolutionary algorithms. However, our problem decomposition obtained by solving the balanced\unbalanced partitioning problem improves the solution quality in three out of four instances. We obtained an average improvement of of 7.26% for SA after 100,000*N evaluations, 2.41% for SA after 2,000,000*N evaluations, and 20.99% for GA after 2,000,000*N evaluations. This technique improves the previous published results for a standard SA and hugely improves the GA performance, therefore reducing the gap between the two algorithms. In one of the problems, which presents very high connectivity, the decomposition does not produce satisfactory results in terms of quality. However, the partitioning approach can still provide a good approximation in a shorter time.

Future research may investigate experiments of larger size MI-FAP benchmarks (in the order of ten thousand transmitters) and interference models that are more realistic and complex than the binary model used here, e.g *multiple interference*. In the latter case, because of its computational complexity, the use of decomposition techniques becomes necessary at a smaller problem size. Finally, some work still need to be completed in order to define a criterion for which, given a data sets, we can predict which is the more effective decomposition to be used.

ACKNOWLEDGEMENTS

We acknowledge the Cardiff University Condor Pool which has provided the resources for the experiments published in this paper [2].

REFERENCES

[1] *FAP web - A website about Frequency Assignment Problems*, , accessed on 1st June 2007, url = "http://fap.zib.de/".

[2] *Cardiff University Condor Pool*, accessed on 1st June 2007, url='http://www.cardiff.ac.uk/insrv/it/condor/index.html".

[3] K.I. Aardal, C.P.M. van Hoesel, and B. Jansen, *A branch-and-cut algorithm for the frequency assignment problem*, R.M. 96\11, 1996.

[4] K.I. Aardal, S.P.M. van Hoesel, A.M.C.A. Koster, C. Mannino, and A. Sassano, "Models and solution techniques for Frequency Assignment Problems," *Annals of Operations Research*, to appear.

[5] D. Beckmann, and U. Killat, *Frequency Planning with respect to Interference Minimization in Cellular Radio Networks*Vienna, 1999.

[6] U. Brandes, M. Gaertler, and D. Wagner, "Experiments on Graph Clustering Algorithms,", *Proc. of the 11th Annual European Symposium on Algorithms (ESA'03)*, Budapest, 2003, pp. 568–579.

[7] G. Colombo, and C.L. Mumford "Comparing Algorithms, Representations and Operators for the Multi-objective Knapsack Problem," *Proc. of the 2005 IEEE Congress on Evolutionary Computation (CEC20052)*, Edinburgh, Scotland, 2005, pp. 1268–1275.

[8] G. Colombo, "A Genetic Algorithm for frequency assignment with problem decomposition," *International Journal of Mobile Network Design and Innovation*, vol. 1-2, pp. 102–112, 2006.

[9] L.M. Correia, Editor, *Wireless Flexible Personalised Communications*, Chichester, UK: Wiley Europe, 2001.

[10] C. Crisan, and H. Muhlenbein, "The breeder genetic algorithm for frequency assignment," *Lecture Notes on Computer Science*, vol. 1498, pp. 897–906, 1998.

[11] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEE Trans. on Evolutionary Computation*, vol. 6, pp. 182–197, 2002.

[12] A. Eisenblatter, *Frequency Assignment in GSM Networks: Models, Heuristics, and Lower Bounds*PhD thesis, Technische Universitat Berlin, Berlin, Germany, 2001.

[13] S. Hurley, and D. Smith, "Meta-Heuristics and channel assignment," in *Methods and algorithms for radio channel assignment*, edited by S. Hurley and R. Leese, Oxford, UK: Oxford University Press, 2002.

[14] A. Kapsalis, P. Chardaire, V. J. Smith, and G. D. Smith, "The radio link frequency assignment problem: A case study using genetic algorithms," *Lecture Notes on Computer Science*, vol. 993, pp. 117–131, 1995.

[15] N. Karaoglu, and B. Manderick, "FAPSTER - a genetic algorithm for frequency assignment problem,", *Proc. of the 2005 Genetic and Evolutionary Computation Conference*, Washington, D.C., USA, 2005.

[16] A.W.J. Kolen, "A genetic algorithm for frequency assignment," *Statistica Neerlandica*, vol. 61-1, pp. 4–15, 2007.

[17] A.M.C.A. Koster, C.P.M. van Hoesel, and A.W.J. Kolen, "Solving partial constraint satisfaction problems with tree decomposition," *Networks*, vol. 40-3, pp. 170–180, 2002.

[18] C. Mannino, G. Oriolo, and F. Ricci, *Solving Stability Problems on a Superclass of Interval Graphs*, T.R. n. 511, Vito Volterra, 2002.

[19] C. Mannino, and A. Sassano, "An enumerative algorithm for the frequency assignment problem," *Discrete Applied Mathematics*, vol. 129-1, pp. 155–169, 2003.

[20] R. Montemanni, J.N. Moon, and D.H. Smith, "An improved Tabu Search algorithm for the Fixed-Spectrum Frequency-Assignment problem," *IEE Trans. on Vehicular technology*, vol. 52-3, pp. 891–901, 2003.

[21] P. Pardalos, J. Rappe, and M. Resende, "An exact parallel algorithm for the maximum clique problem,," in *High Performance Algorithms and Software in Nonlinear Optimization*, edited by P.P.R. De Leone, A. Murl'i, G. Toraldo, Kluwer, Dordrecht, 1998.

[22] S. van Dongen, *A cluster algorithm for graphs*, Technical Report INS-R0010, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, 2000.