An Efficient Heuristic Algorithm for Channel Assignment Problem in Cellular Radio Networks

Goutam Chakraborty

Abstract— **This paper presents an efficient heuristic algorithm for channel assignment problem in cellular radio networks. The task is to find channel assignment with minimum frequency bandwidth necessary to satisfy given demands from different nodes in a cellular network. At the same time the interference among calls within the same cell and from different neighboring cells are to be avoided, where interference is specified as the minimum frequency distance to be maintained between channels assigned to a pair of nodes. The simplest version of this problem, where only co-channel interferences are considered, is NP-complete. The proposed algorithm could generate a population of random valid solutions of the problem very fast. The best among them is the optimum or very near to optimum solution. For all problems with known optimal solutions, the algorithm could find them. A statistical estimation of the performance of the proposed algorithm is done. Comparison with other methods show that our algorithm works better than the algorithms that we have investigated.**

Keywords— **Cellular radio networks, Channel assignment problem, Heuristic algorithm**

I. Introduction

RECENT development of various forms of mobile communications and their increasing popularity assured its high growth rate. In mobile cellular network the geographical area covered by the network is assumed to be divided into a number of cells of hexagonal shape [1]. Communications to and from mobile hosts (MHs) in each cell are serviced by a base station (BS) located at the center of the cell. The whole bandwidth available for mobile users is divided into a set of carriers, sometimes loosely called channels. Generally, a carrier is subdivided into a number of channels by time-division multiplexing, and can be used by different users in that cell. But in this paper (and all the references cited) we assume that time division multiplex is not used, and use the word channel and carrier interchangeably to mean carrier.

Each channel can support a call. When a channel is allocated to a cell (BS), a mobile user in that cell can use that channel. A channel can simultaneously be used by multiple BSs, if their distances are more than the *minimum reuse distance*, i.e., the distance is enough such that there will be no interference. This model is same as discussed in [2] and is illustrated in Fig. 1. For example, in Fig. 1, a certain frequency distance between carriers used within cell 17 and between cell 17 and all its adjacent cells (9, 10, 16, 18, 20 and 21, marked by dashed lines) has to be maintained. Sometimes even further distant cells, i.e. cells 2, 3, 4, 8, 11, 15 and 19, may be within the interference range. The minimum physical distance after which the same channel could be used, is an important parameter for the mobile network design and is called *distance reuse*, as shown in Fig. 1. Here *distance reuse*, usually expressed in units of number of cells, is 3. The interference pattern between any two pair of cells of the mobile system is fixed and is specified by a $N \times N$ matrix, where N is the number of base stations.

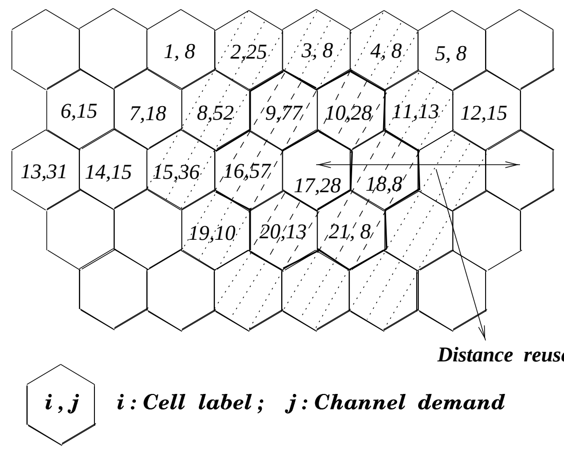

Fig. 1. A cellular network with 21 cells

Depending on the maximum number of possible simultaneous calls in a cell, the demand of channels for individual cells are set. Due to non-uniform distribution of business and social activities over the whole region of the cellular network, traffic density and channel demand vary from cell to cell. In Fig. 1, a 21-node mobile network is shown. The cell number and corresponding channel demand is printed inside the cell. The traffic demand is non-uniform with cell 9 having the highest traffic demand of 77 channels, and channel demand is only 8 for cells 1, 3, 4 etc.

The problem of channel allocation is to assign carriers to cells so that, as far as possible, the traffic demand for the cell is met. This is to minimize the overall rate of blocked calls, where a call is blocked when the BS can not allocate channel to a MH within it. There are three types of carrier allocation strategies [3], (1) *static allocation*, where depending on an a-priori available static demand of different cells the carries are assigned [4], (2) *dynamic allocation*, where due to change in traffic pattern with time, the carrier demands at different cells change dynamically, and a free channel is allocated to a BS requiring it [5], and (3) *flexible allocation*, where each cell is allocated a fixed set of permanent carriers and a number of channels are set aside to be dynamically allocated depending on changing requests

The author is with the department of Software and Information Science, Iwate Prefectural University, Iwate ken, Takizawa Mura, Japan - 020-0193. E-mail: goutam@iwate-pu.ac.jp .

from base stations [6]. In this paper we will consider *static allocation.*

The fundamental problem in the operation and planning of a mobile network is the limited available frequency bandwidth and often poses severe limitations on the performance of the system. Assignment of the limited available frequency resource to the different base stations, such that the interference among the calls is below tolerable limit and at the same time meeting the traffic demands, is an important critical design step. Thus, this is a constraint optimization problem. We will consider the problem of finding minimum frequency bandwidth and the corresponding channel assignment, satisfying given channel demands for different cells without violating interference constraints. In addition, presently we will consider only static allocation as is done in [4] [7] [8] [9] [10] [11]. The algorithm could be extended for dynamic allocation too. That is discussed at the end in the conclusion section V.

In section II, we describe the channel assignment problem in a formal way and introduce the notations we will use throughout this paper. We will also briefly introduce the previous works in section II. In section III, the algorithm is proposed with explanations. The pseudocode of the algorithm with complexity analysis is available in the appendix. In section IV, simulation results and statistical analysis of the performance of the algorithm are reported. Section V is the conclusion, where we summarize this work and shed some light on further possible extensions of this algorithm.

II. Channel Assignment Problem in Cellular Radio Networks

The total bandwidth i.e. the number of carriers are the basic unit of resource allocation. It is assumed that the carrier frequencies are consecutive and evenly spaced, and therefore could be expressed as $f = f_0 + m \times \Delta f$, with $m > 0$ an integer. Thus there is an one-to-one correspondence between f and m, and we will represent different frequencies by $f_1, f_2, \ldots, f_m, \ldots, f_M$, where $M \times \Delta f$ is the total available bandwidth. In our problem M is not known, and we need to minimize M such that the required amount of channels are allotted to base stations fulfilling their respective traffic demands without causing any interference. The mobile network is defined as:

$\mathbf{X}$ *:* is the cellular network. N is the number of base stations (also called cells) in the network. The whole network $\mathbf{X}$ is the set of cells $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots \mathbf{x_N}\}$.

d_i *:* is the number of channels required in cell i to be able to handle its traffic, where $1 \leq i \leq N$. For the whole network it is represented by the demand vector $\mathbf{D} = \langle d_1, d_2, \ldots d_N \rangle$.

c_{ij} *:* is the frequency separation required to avoid interference between any two channels allotted to cell i and cell j respectively. Here, $1 \leq i \leq N$ and $1 \leq j \leq N$. We will denote this $N \times N$ constraint matrix, also called compatibility matrix [7], by $\mathbf{C}$. It is likely that $\mathbf{C}$ is a symmetric matrix, but we will make no such assumption in our algorithm.

Thus, the channel assignment problem is specified by a triple $(\mathbf{X},\ \mathbf{D},\ \mathbf{C})$. Depending on the values of c_{ij} there are more commonly used terms like,

1. *Cochannel constraint* when $c_{ij} = 1$, meaning that the same frequency can not be assigned to certain pairs of cells simultaneously.
2. *Adjacent channel constraint* when $c_{ij} = 2$, meaning that adjacent frequencies (e.g. f_i and f_{i+1}) can not be assigned to certain pairs of cells simultaneously.
3. *Cosite constraint* C_{ii} is the minimum distance of separation of frequencies of two carriers assigned to the same cell $\mathbf{x_i}$. Its value depends on the communication system used. Thus, for a particular system, c_{ii} s for all $\mathbf{x_i}$s are same. But it is not assumed in our algorithm.

Any other values of c_{ij}, say λ, implies that the distance of the carrier frequencies assigned to cell i and cell j should be at least λ, for interference free communication. In other words the m values (i.e. the subscripts) of assigned frequencies should differ at least by λ. A $c_{ij} = 0$ represents that there is no interference even when the two nodes i and j use the same carrier frequency. In reality, values of c_{ij}s depend on the communication system used, and the way the cells are geographically located i.e. the topography of the mobile network. In this paper we will assume that $\mathbf{C}$ matrix is given and the elements c_{ij} can take any arbitrary values.

In Fig. 1, let us assume that adjacent cells (e.g. 17 and 9, 17 and 10 etc. - shaded with dashed lines) have *adjacent channel constraints*, and cells at a cell distance of 2 (e.g. 17 and 2, 17 and 3 etc. - shaded with dotted lines) have *cochannel constraints*. Further if *cosite constraint* is assumed to be 7 for all cells, then the 17^{th} row c_{17} of the corresponding compatibility matrix $\mathbf{C}$ would be:

$$c_{17} = \ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 2\ 2\ 1\ 0\ 0\ 0\ 1\ 2\ 7\ 2\ 1\ 2\ 2$$

The basic resource i.e. the total frequency bandwidth available can be considered as a constraint, when a certain fixed bandwidth is allotted. Algorithms, where the required number of carriers is not reduced from the initial estimate, start with a heuristic estimate of the lower bound [7] of the number of frequencies required and try to find a valid solution [9] [11] [12] [13] [14]. They start with an initial channel allotment which does not satisfy the constraints, and eventually try to converge to a solution satisfying them. It is also possible that no valid solution could be found with the initial bandwidth estimate. On the other hand heuristic approaches [15] [16] [8] [10] try to find the minimum bandwidth required to solve the channel assignment problem. In this paper too, the bandwidth is considered as an optimization criterion and the algorithm tries to minimize the required bandwidth satisfying all the constraints. We think that this approach is more logical. With a pre-assigned bandwidth, there may not be any solutions satisfying all the constraints. However, if we start with more carriers, we may end up with a bandwidth inefficient solution that would be a waste of a very important resource. The algorithm we proposed could albeit handle both situations, as we will soon see.

The task of the channel assignment algorithm is to find the frequency allotment matrix $F_{(M \times N)}$. The elements of F, f_{mi}, can take values either 0 or 1. A 1 at f_{mi} indicates that frequency f_m is allotted to the i^{th} cell and a 0 indicates that it is not. Obviously, $1 \leq m \leq M$ and $1 \leq i \leq N$. The objective is that M has to be minimized, subject to satisfying the frequency separation constraints i.e. when $f_{mi} = f_{lj} = 1$,

$$|f_{mi} - f_l| \geq c_{ij}$$

for all m, l, i, j except for $m = l$ and $i = j$. For $i = j$, it becomes the *cosite constraint* c_{ii}.

The second constraint is the number of carriers required by different cells to handle the respective traffic. In this paper we considered that to be static and known a-priori. It is described by a vector $\mathbf{D}$ of n-elements. Element d_i denotes the number of carriers required by cell i, so that different cells could handle their respective traffic. Thus,

$$\sum_{m=1 \text{ to M}} f_{mi} = d_i$$

A. An Example

$$\mathbf{C_1} = \begin{bmatrix} 5 & 4 & 0 & 0 \\ 4 & 5 & 0 & 1 \\ 0 & 0 & 5 & 2 \\ 0 & 1 & 2 & 5 \end{bmatrix} \qquad \mathbf{D_1} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 3 \end{bmatrix}$$

Fig. 2. A 4-cell Channel allocation problem *Example 1*

SOLUTION 1				Freq	SOLUTION 2			
1	0	1	0	f_1	1	0	0	1
0	0	0	0	f_2	0	0	0	0
0	0	0	1	f_3	0	0	1	0
0	0	0	0	f_4	0	0	0	0
0	1	0	0	f_5	0	1	0	0
0	0	0	0	f_6	0	0	0	1
0	0	0	0	f_7	0	0	0	0
0	0	0	1	f_8	0	0	0	0
0	0	0	0	f_9	0	0	0	0
0	0	0	0	f_{10}	0	0	0	0
0	0	0	0	f_{11}	0	0	0	1
0	0	0	0	f_{12}				
0	0	0	1	f_{13}				
				f_{14}				

TABLE I

EXAMPLES OF VALID SOLUTIONS FOR PROBLEM *Example 1*

In Fig. 2 we present a channel allocation problem for a 4-cell network, where the inhomogeneous channel requirements for the four cells are given by vector $\mathbf{D_1}$, and the frequency interference between cells is described by the compatibility matrix $\mathbf{C_1}$. In future we will refer to this problem as *Example 1*.

In Table. 1 we have given two solutions of this problem. In SOLUTION 1 the required number of channels is 13, whereas in SOLUTION 2 it is 11 (which is actually an optimum solution).

B. Previous Works

In [16] it was shown that the channel assignment problem is equivalent to graph coloring problem, when only cochannel constraints are considered, that is the entries in $\mathbf{C}$ matrix are either 0 or 1. The graph coloring problem, which is a simpler version of this channel assignment problem, is a well known NP-complete problem. Thus the complexity of searching for the optimum solution, in case of a channel assignment problem, grows exponentially with the number of cells and is NP-complete.

References [4] to [23] are only a partial list of different algorithms proposed during the last two decades to solve the channel assignment problem. These algorithms could be classified into following three categories. References are in order of publication dates.

- Heuristic approaches [15] [16] [4] [7] [8] [10], published during 1978 to 1991.
- Neural Network or simulated annealing based approaches [9] [11] [17] [12] [13] [14], published during 1991 to 1997.
- Genetic Algorithm (GA) or Evolutionary Computation (EC) based approaches [18] [19] [20] [23], published during 1995 to 1998.

Heuristic approaches usually adopt ideas from previously introduced heuristic algorithms for solving graph coloring problems [21] [22] [8] [10]. All neural network based algorithms use some modified version of Hopfield type neural network. The recent reports are mainly evolutionary computation based approaches.

Most of the algorithms for the channel assignment problem assume that the number of carriers is fixed (say K) and is known a-priori. This is true for references [4] and [7] to [23]. K is calculated using some heuristics and may be more or less than the optimum value. The neural network or genetic algorithm approaches initially start with solution matrix F whose size is $K \times N$, which remains same throughout the execution of the algorithm. The initial solution is made such that some constraint, e.g. co-site constraint, which is easy to set, is initially satisfied. In case of using Hopfield neural network, the connection weights among neurons are made such that, at the stable energy minimum state, the output voltages of the neurons represent the optimum solution satisfying all the constraints. Neuron voltages are initially set at random. When the network is relaxed to settle down to energy minimum, the solution is obtained. Due to multimodal energy function, it is possible that the network settles down to a local minimum, and the solution may not be optimal or valid. For GA, the optimum solution satisfying all constraints is found after evolving the population through generations. If the solution could not be found, the algorithm is repeated with increased value of K. Sometimes the success of GA approach, as in [23], strongly depends on the proper choice of the value of mutation probability. The effective value of this probability is different for different problems and depends on the size of the network and constraints. The type of cross-over they used in [23] severely restricts the search space. In [19], in addition to crossover and mutation probabilities, the fitness function uses two extra parameters to bias the co-site and co-channel constraint terms. The success of [19] greatly depends on the setting and balance of these parameters, which again problem dependent. Though they reported good results for small and selected problems with less interferences between neighboring cells, these approaches have strong restrictions as they have to be customized for the problem.

III. The Algorithm

The proposal here is a heuristic algorithm to create, not a single solution but a pool of valid solutions of the given channel assignment problem. While creating these solutions the optimization criterion of minimizing the bandwidth i.e. M, is not taken care of. All these solutions satisfy the interference constraint (set by $\mathbf{C}$ matrix), and the traffic demand of different cells (set by $\mathbf{D}$ vector). The algorithm could find optimum or near optimum solution, if the best one from the pool is selected. The algorithm is simple and fast, $O(n^2)$ where n is the number of network nodes. It is much faster than most of the existing heuristic algorithms which are typically $O(m^2)$, where m is the total number of channel requirements. Though it depends on $\mathbf{D}$, typically $m \gg n$. With the proposed algorithm, if a moderately large population (usually about 100) of valid solutions are created and the best one is selected, the probability of getting the optimum solution is very high. In section III.A, we will describe the details of the algorithm, and explain it with an example. In section IV, we will discuss several experiments, their results, and do statistical analysis of them.

A. Description of the Algorithm

Here, we give the informal description of the algorithm with an example. The pseudocode and analysis of the complexity of the algorithm is given in the appendix.

For a N-node network, we start with a solution of the frequency allotment matrix $F_{M\times N}$ with M rows. M is sufficiently large so that even after adequate channels are allocated to the cells satisfying their demands without violating the constraints, the solution $F_{K\times N}$ would always be found where $K < M$.

Let us represent the elements of the allotment matrix F as f_{ij}. The algorithm creates a population, P number of such solution matrices. Let us name them $F^1, F^2, \ldots F^p, \ldots F^P$. All the elements of the allotment matrix are initialised with 0s. After the algorithm is executed, the elements of the allotment matrix $F_{M\times N}$ could either be a "0", a "1", a "-1" or a "9". We use a quadnary representation as follows:

$$[0,\ +1,\ -1,\ +9] = [\ \textit{assignable},\ \textit{used},\ \textit{unassignable},\ \textit{unused}]$$

	cell 1	cell 2	cell 3	cell 4
f_1	0	0	0	0
f_2	0	0	0	0
f_3	0	0	0	0
f_4	0	0	0	0
f_5	0	0	0	0
f_6	0	0	0	0
f_7	0	0	0	0
f_8	0	0	0	0
f_9	0	0	0	0
f_{10}	0	0	0	0
f_{11}	0	0	0	0
f_{12}	0	0	0	0
f_{13}	0	0	0	0
f_{14}	0	0	0	0
f_{15}	0	0	0	0

TABLE II

Initial configuration of a solution matrix F^1

A "0" at f_{ij} indicates that the j^{th} node (represented by the column number) is not using the i^{th} frequency (represented by the row number). In addition, it also means that if that frequency is *assigned* to that node, there will be no conflict with other existing frequency allocations. A "1" at f_{ij} indicates that the j^{th} node is using the frequency f_i. A "-1" at f_{ij} indicates that the j^{th} node is not assigned the frequency f_i, and it is *unassignable* because then it would cause interference with some other existing allocations, according to the compatibility matrix. Indicator '9' at f_{i1}, i.e. at the first column, indicates that the corresponding frequency f_i is not allocated to any cell. They appear at the end, marking the tail part of $F_{M\times N}$ i.e., for those frequencies that remain unallocated.

To explain the algorithm using a small example, we choose the simple case of 4-node network *Example 1*, introduced in section II.A. For convenience, once again the compatibility matrix $\mathbf{C}_1$, and the demand vector $\mathbf{D}_1$ are shown in Fig. 3. After initialization, all the entries in F matrices will be 0s as in Table. II. Here we have started with 15 rows of F matrix i.e., $M = 15$.

$$\mathbf{C_1} = \begin{bmatrix} 5 & 4 & 0 & 0 \\ 4 & 5 & 0 & 1 \\ 0 & 0 & 5 & 2 \\ 0 & 1 & 2 & 5 \end{bmatrix} \qquad \mathbf{D_1} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 3 \end{bmatrix}$$

Fig. 3. A 4-cell Channel allocation problem *Example 1*

Now let us convert one of the initial matrix, say F^1, to a valid solution of the *Example 1* problem. First an arbitrary permutation of the digits $1,\ 2,\ 3, \ldots N$ is done, and we get an arrangement whose elements in order are $\sigma_1, \sigma_2, \ldots \sigma_N$, denoting the nodes in a particular sequence. Our example is a 4-node network. Let the result of random permutation be 3, 1, 2, 4. So, $\sigma_1 = 3$, $\sigma_2 = 1$, $\sigma_3 = 2$, $\sigma_4 = 4$.

Allotment of channels will start with node σ_1, then σ_2 and so on until σ_N. In this case, allotment of channels will start with node 3 i.e. column 3 of the F^1 matrix. The first free Channel f_1 is assigned to node 3, and f_{13} is changed $0 \rightarrow 1$. Now the algorithm checks which are the elements i.e. *(freq, node)* locations in the F^1 matrix, where if a transmission is allowed would result in interference according to the compatibility matrix $\mathbf{C}_1$. As node 3 in this case does not conflict at all with node 1 and node 2, all entries in the column of node 1 and node 2 remain unchanged. As the *cosite constraint* for node 3 is 5, f_{23} to f_{53} are all packed with "-1", prohibiting any possible allotment at those positions, which would otherwise result in *cosite constraint* violation. Between node 3 and node 4 it is an *adjacent channel constraint*. To prevent any conflict between node 3 and node 4 frequency assignment, f_{14} and f_{24} are changed $0 \rightarrow -1$. Now the task of putting "-1" is over, the algorithm checks if the demand for channels for node σ_1 (i.e. node 3 here) is met or not. If the demand is not met, we go to the next row in column σ_1 to find the next "0" and assign the corresponding channel to σ_1 (as it is still free). In *Example 1*, as the demand for node 3 is only 1, the assignment for node 3 is complete just after the first assignment.

In the permutation, the next σ_2 is node 1. We start along column 1 to find a "0" i.e. a free channel which could be allocated to node 1. As f_{11} is a "0", channel 1 is assigned to node 1 without any interference constraint violation. Following same rule as in the case of node σ_1 (i.e. node 3), "-1"s are packed at relevant locations of F^1. This will put "-1"s at $f_{21}, f_{31}, f_{41}, f_{51}$ (*cosite constraint*) and at $f_{12}, f_{22}, f_{32}, f_{42}$ (to avoid interference with transmissions from node 2). Node σ_2 (i.e. node 1) has no interference constraint with node 3 and node 4. So column 3 and 4 remain unchanged. Here too the demand for node σ_2 is only 1, and therefore its channel assignment is complete. We next start channel assignment for node σ_3 i.e. node 2. On column 2 the first "0" is found at row 5 and f_5 is assigned to node 2. "0"s are converted to "-1" as explained to avoid interference from this assignment. This is repeated until the last node i.e. node σ_N (here node 4) gets all the channels it requires.

Thus we see that SOLUTION 1 in Table. III is generated by the algorithm when the random permutation produces the sequence of nodes as

$$3,\ 1,\ 2,\ 4$$

Some other permutations would also lead to the same solutions. Those permutations are:

$$3,\ 1,\ 4,\ 2; \quad 3,\ 4,\ 1,\ 2; \quad 1,\ 2,\ 3,\ 4; \quad 1,\ 3,\ 2,\ 4; \quad 1,\ 3,\ 4,\ 2;$$

In fact, for *Example 1*, all permutations where 3 comes before 4 and 1 comes before 2, would lead to the same solution. Permutations in which 2 precedes 1, or 4 precedes 3, would produce different solutions.

SOLUTION 1					SOLUTION 2			
+1	-1	+1	-1	f_1	+1	-1	-1	+1
-1	-1	-1	-1	f_2	-1	-1	-1	-1
-1	-1	-1	+1	f_3	-1	-1	+1	-1
-1	-1	-1	-1	f_4	-1	-1	-1	-1
-1	+1	-1	-1	f_5	-1	+1	-1	-1
-1	-1	+0	-1	f_6	-1	-1	-1	+1
-1	-1	-1	-1	f_7	-1	-1	-1	-1
-1	-1	-1	+1	f_8	-1	-1	+0	-1
+0	-1	-1	-1	f_9	+0	-1	+0	-1
+0	+0	+0	-1	f_{10}	+0	+0	-1	-1
+0	+0	+0	-1	f_{11}	+0	-1	-1	+1
+0	+0	-1	-1	f_{12}	+9	+0	-1	-1
+0	-1	-1	+1	f_{13}	+9	+0	+0	-1
+9	+0	-1	-1	f_{14}	+9	+0	+0	-1
+9	+0	+0	-1	f_{15}	+9	+0	+0	-1

TABLE III

EXAMPLES OF VALID SOLUTIONS FOR PROBLEM 1

Similarly, the sequence that generated SOLUTION 2 in Table. III is:

$$1,\ 4,\ 2,\ 3$$

And other sequences that can generate the same solution are

$$4,\ 1,\ 2,\ 3; \quad 1,\ 4,\ 3,\ 2; \quad 4,\ 1,\ 3,\ 2; \quad 4,\ 3,\ 1,\ 2$$

Permutations in which node 4 precedes 2 and 3, and node 1 precedes node 2, will all generate solutions like SOLUTION 2.

In fact, SOLUTION 2 is an optimal solution. For other permutations for which node 4 precedes node 3, would lead to optimal solutions of same channel bandwidth of 11 as in SOLUTION 2, though the assignments of "1"s, "-1"s and "0"s will be different. Thus many possible permutations of the nodes will be mapped to solution frame of same bandwidth, and many permutations would create optimal solutions. The pseudocode of the algorithm is available in the appendix.

IV. Experiments: Set up, Results and Analysis

To test the performance of the algorithm we did exhaustive simulations for various problems with varied degrees of complexity. Problems for which the frequency bandwidth for the optimal solution is known, our algorithm could always find that. For other problems too, it could find the known best solution. In section IV.A, we describe how the different problems were created, following which we give the simulation results for the different problems. At the end, in section IV.C, statistical analysis of the results is done and we calculated the probability of finding optimum solution for different cases.

Before going into the details of different experiments, let us mention here about the trivial lower bound of bandwidth requirement. We also explain in brief how the level of difficulty of the channel assignment problem depends on the problem specifications. If $(c^*_{ii} \times d^*_i)$ is the maximum of $(c_{ii} \times d_i)$ for all i, then the trivial lower bound of the required channel bandwidth is $(c^*_{ii} \times (d^*_i - 1) + 1)$. Here, as is obvious, we assume that c_{ii} values are highest among all c_{ij}s. As c_{ii} s are usually the same for all i, this bound can simply be written as $(c_{ii} \times (d^*_i - 1) + 1)$, where $d_i{}^*$ is the maximum element in **D** vector. When c_{ii} is high and d^*_i is large compared to other cells' demands, this trivial lower bound is typically the optimum value of bandwidth requirement. Many of the reported works, where the required frequency length has to be decided a-priori, start their experiments with this frequency bandwidth. But for problems with high c_{ij}, low c_{ii}, and channel demand for different cells nearly equal, the optimum bandwidth is usually much higher than this trivial lower bound.

As is obvious (*Lemma 3:* in [7]), when elements of the *demand vector* **D** and the *compatibility matrix* **C** become larger the problem becomes more difficult. One way is to assign the *degree* of a cell $\mathbf{x_i}$ as [8],

$$\delta_i = \left(\sum_{j=1}^{N} d_i \, c_{ij} \right) - c_{ii}, \qquad 1 \le i \le N,$$

which is a heuristic measure of the difficulty of assigning frequency in that cell. Typically, if all the elements of **D** are nearly equal, and high, and c_{ij} elements are high and comparable to c_{ii}, the problem becomes very difficult. We constructed 21, 25 and 55 node networks with different degrees of interference. We then ran our algorithm with different heterogeneous demands, every problem for several million times. Finally we analyzed the results statistically, and reported the probability of success of getting optimum solution and corresponding required computation times.

A. Problem Set up

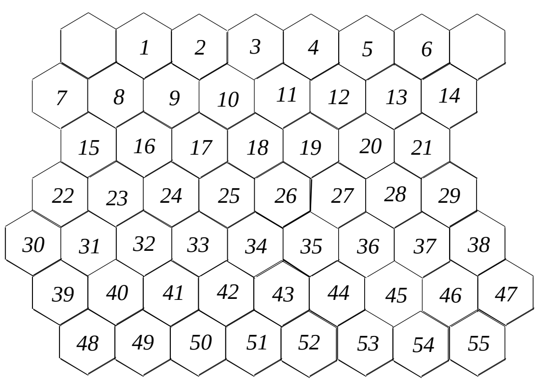

Fig. 4. A cellular network with 55 cells

We implemented different problems with a 21-node network as well as 25-node network. These two networks were used in many previous works [7] [8] [9] [11]. The 21-node network is already shown in Fig. 1. A 55-node network, as shown in Fig. 4, is also constructed for our experiments.

We introduce two parameters here, namely ρ and α. Parameter ρ actually represents the range of interference and is related to the *minimum reuse distance* as $\rho =$ *(minimum reuse distance - 1)*. In Fig. 1 $\rho = 2$, as the *minimum reuse distance* is 3.

The other parameter α decides the degree of interference between cells. When the cell distance is 1, the required frequency distance (to avoid interference) is α. This required frequency distance is decreased by 1, as the cell distance increases by 1, until it reaches the value 1 and then remain same within cell distance of ρ. Thus, for parameters $\rho = 2$ and $\alpha = 1$, we have only *co-channel* interference between two neighboring cells at *cell distances* 1 as well as 2. Beyond *cell distance* 2 there is no interference. Parameters $\rho = 2$ and $\alpha = 2$ mean, we have *adjacent channel* interference between cells with *cell distance* 1, and *co-channel* interference between cells with *cell distance* 2. This is summarized in Table. IV.

Value of ρ	Value of α	*cell distance* between two cells i and j	c_{ij} entry in the compatibility matrix **C**
2	1	1	1
		2	1
		> 2	0
2	2	1	2
		2	1
		> 2	0

TABLE IV

VALUE OF ρ AND α AND CORRESPONDING INTERFERENCE BETWEEN CELLS

Obviously, the problem becomes more difficult as ρ and α are increased. For all problems created, we set $\rho = 2$ i.e. *minimum reuse distance* of 3. α is set to values 1 or 2. As we will soon see, with $\alpha = 2$ the problems are more difficult.

The other parameter for constructing different problems is the value of c_{ii}. When c_{ii} is high compared to c_{ij} values, the problem is easy. *Compatibility matrices* with 21-node network and c_{ii}s set to different values of 7, 6, 5 and 4 were created, making the problem more and more difficult. *Compatibility matrix* for 25-node and 55-node networks with $\rho = 2$, $\alpha = 1$ are also constructed and used in our simulations. For convenience of reference, we assign short names to the different *compatibility matrices* as shown in the following Table. V.

We here show explicitly one such *Compatibility matrix* $\mathbf{C}^5_{21}$ in Table. VI.

For the 21-node network we used two demand vectors $\mathbf{D}^1_{21}$ and $\mathbf{D}^2_{21}$ as follows. Many other researchers used the same demand vectors in their works [7] [8] [9] [11].

$\mathbf{D}^1_{21} = \langle$ 8 25 8 8 8 15 18 52 77 28 13 15 31 15 36 57 28 8 10 13 8 $\rangle$

$\mathbf{D}^2_{21} = \langle$ 5 5 5 8 12 25 30 25 30 40 40 45 20 30 25 15 15 30 20 20 25 $\rangle$

As is evident, the demand vector $\mathbf{D}^2_{21}$ poses harder problem because the demand for different cells are more uniform compared to that of $\mathbf{D}^1_{21}$, where cell 9 has a sudden high demand of 77 channels and there are many cells with low

(01)	5	1	1	0	0	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0
(02)	1	5	1	1	0	0	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0
(03)	1	1	5	1	1	0	0	1	1	1	1	0	0	0	0	1	1	1	0	0	0
(04)	0	1	1	5	1	0	0	0	1	1	1	1	0	0	0	0	1	1	0	0	0
(05)	0	0	1	1	5	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0
(06)	1	0	0	0	0	5	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0
(07)	1	1	0	0	0	1	5	1	1	0	0	0	1	1	1	1	0	0	1	0	0
(08)	1	1	1	0	0	1	1	5	1	1	0	0	0	1	1	1	1	0	1	1	0
(09)	1	1	1	1	0	0	1	1	5	1	1	0	0	0	1	1	1	1	1	1	1
(10)	0	1	1	1	1	0	0	1	1	5	1	1	0	0	0	1	1	1	0	1	1
(11)	0	0	1	1	1	0	0	0	1	1	5	1	0	0	0	0	1	1	0	0	1
(12)	0	0	0	1	1	0	0	0	0	1	1	5	0	0	0	0	0	1	0	0	0
(13)	0	0	0	0	0	1	1	0	0	0	0	0	5	1	1	0	0	0	0	0	0
(14)	1	0	0	0	0	1	1	1	0	0	0	0	1	5	1	1	0	0	1	0	0
(15)	1	1	0	0	0	1	1	1	1	0	0	0	1	1	5	1	1	0	1	1	0
(16)	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	5	1	1	1	1	1
(17)	0	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	5	1	1	1	1
(18)	0	0	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	5	0	1	1
(19)	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1	0	5	1	1
(20)	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1	1	5	1
(21)	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1	1	5

TABLE VI

21-NODE *Compatibility matrix* $\mathbf{C}_{21}^{5}$

Serial no.	No. of Cells (N)	ρ	α	c_{ii}	*Compatibility matrix* name
01	21	2	1	7	$\mathbf{C}_{21}^{1}$
02	21	2	2	7	$\mathbf{C}_{21}^{2}$
03	21	2	1	6	$\mathbf{C}_{21}^{3}$
04	21	2	2	6	$\mathbf{C}_{21}^{4}$
05	21	2	1	5	$\mathbf{C}_{21}^{5}$
06	21	2	2	5	$\mathbf{C}_{21}^{6}$
07	21	2	1	4	$\mathbf{C}_{21}^{7}$
08	21	2	2	4	$\mathbf{C}_{21}^{8}$
09	25	2	1	2	$\mathbf{C}_{25}^{1}$
10	55	2	1	7	$\mathbf{C}_{55}^{1}$

TABLE V

NOMENCLATURE OF *Compatibility matrices*

channel demand of less than 20.

For the 25-node network we used the following two demand vectors:

$\mathbf{D}_{25}^{3} = \langle$ 10 11 9 5 9 4 5 7 4 8 8 9 10 7 7 6 4 5 5 7 6 4 5 7 5 $\rangle$

$\mathbf{D}_{25}^{4} = \langle$ 5 5 5 8 12 25 30 25 30 40 40 45 20 30 25 15 15 30 20 20 25 8 5 5 5 $\rangle$

Here, $\mathbf{D}_{25}^{4}$ poses harder problem. For the 55-node network we used the following two demand vectors:

$\mathbf{D}_{55}^{5} = \langle$ 5 5 5 8 12 25 30 25 30 40 40 45 20 30 25 15 15 30 20 20 25 8 5 5 5 5 5 5 8 12 25 30 25 30 40 40 45 20 30 25 15 15 30 20 20 25 8 5 5 5 25 8 5 5 5 $\rangle$

$\mathbf{D}_{55}^{6} = \langle$ 10 11 9 5 9 4 5 7 4 8 8 9 10 7 7 6 4 5 5 7 6 4 5 7 5 10 11 9 5 9 4 5 7 4 8 8 9 10 7 7 6 4 5 5 7 6 4 5 7 5 6 4 5 7 5 $\rangle$

Experiments were done with different combinations of *compatibility matrices* and *demand vectors* and are reported in section IV.B.

B. Simulation and Analysis of Results

Results obtained from different experiments are summarized in Table. VII. In all the cases, the results are either equally good or better than those reported using different soft computing approaches or heuristic algorithms. Problems for which we are not aware of any reported result, we put the symbol $\emptyset$ in the corresponding column of Table. VII. While evaluating the performance of our algorithm, we ran it to generate 10 million or more valid solutions (F^is) for each problem. Out of them, solutions having minimum channel bandwidth requirement are the best solutions and are denoted as F^*. For all cases, F^*s are better or equally good as solutions reported by earlier works, when there is any. When a trial solution F^i is such a best solution F^*, we call it a *hit*.

Though computation time is not important for static allocation problems, in Table. VII we included the time required for creating 1000 solutions. All computations were performed in DEC ALPHA station. As the coding was not optimized, and other reporting and analysis codes were embedded in the same program, the actual execution time would be comparable but much less.

The number of *hit*s, when a population of 1000 solutions are created, is reported in Table. VIII. For simple problems with *compatibility matrices* like $\mathbf{C}_{21}^{1}$, $\mathbf{C}_{21}^{2}$ etc., solutions with the obvious lower bounds i.e., $7 \times (77 - 1) + 1 = 533$ for $\mathbf{C}_{21}^{1}$ and $\mathbf{D}_{21}^{1}$, or $7 \times (45 - 1) + 1 = 309$ for $\mathbf{C}_{21}^{1}$ and $\mathbf{D}_{21}^{2}$, were many. Please see Fig. 5(a)(b). The probability of *hit* in 1000 trials is almost 1. When compatibility matrix is changed to $\mathbf{C}_{21}^{2}$ (i.e., $\alpha = 2$), the number of *hit*s are much less.

As the value of c_{ii} is decreased and for higher interference with $\alpha = 2$, the problem becomes hard. Not only the quality of solutions is worse, i.e., the required bandwidth is more compared to the trivial lower bound, but also the number of *hit*s are less in comparison to simpler problems.

Serial number	Problem Specification		Trivial lower bound	Existing reported best	Best solution F^*	Computation time for 1000 F^is
	Compatibility matrix	*Demand vector*				
1	$\mathbf{C}_{21}^{1}$	$\mathbf{D}_{21}^{1}$	533	533	533	8.2 Sec.
2	$\mathbf{C}_{21}^{1}$	$\mathbf{D}_{21}^{2}$	309	309	309	6.0 Sec.
3	$\mathbf{C}_{21}^{2}$	$\mathbf{D}_{21}^{1}$	533	533	533	11.1 Sec.
4	$\mathbf{C}_{21}^{2}$	$\mathbf{D}_{21}^{2}$	309	309	309	10.2 Sec.
5	$\mathbf{C}_{21}^{3}$	$\mathbf{D}_{21}^{1}$	457	$\emptyset$	457	8.9 Sec.
6	$\mathbf{C}_{21}^{3}$	$\mathbf{D}_{21}^{2}$	265	$\emptyset$	265	8.1 Sec.
7	$\mathbf{C}_{21}^{4}$	$\mathbf{D}_{21}^{1}$	457	$\emptyset$	457	9.8 Sec.
8	$\mathbf{C}_{21}^{4}$	$\mathbf{D}_{21}^{2}$	265	$\emptyset$	280	7.9 Sec.
9	$\mathbf{C}_{21}^{5}$	$\mathbf{D}_{21}^{1}$	381	381	381	7.5 Sec.
10	$\mathbf{C}_{21}^{5}$	$\mathbf{D}_{21}^{2}$	221	221	221	6.9 Sec.
11	$\mathbf{C}_{21}^{6}$	$\mathbf{D}_{21}^{1}$	381	$\emptyset$	463	9.5 Sec.
12	$\mathbf{C}_{21}^{6}$	$\mathbf{D}_{21}^{2}$	221	$\emptyset$	273	7.7 Sec.
13	$\mathbf{C}_{21}^{7}$	$\mathbf{D}_{21}^{1}$	305	$\emptyset$	305	7.3 Sec.
14	$\mathbf{C}_{21}^{7}$	$\mathbf{D}_{21}^{2}$	177	$\emptyset$	197	6.8 Sec.
15	$\mathbf{C}_{21}^{8}$	$\mathbf{D}_{21}^{1}$	305	$\emptyset$	465	8.4 Sec.
16	$\mathbf{C}_{21}^{8}$	$\mathbf{D}_{21}^{2}$	177	$\emptyset$	278	7.5 Sec.
17	$\mathbf{C}_{25}^{1}$	$\mathbf{D}_{25}^{3}$	21	73	73	1.9 Sec.
18	$\mathbf{C}_{25}^{1}$	$\mathbf{D}_{25}^{4}$	89	$\emptyset$	121	6.3 Sec.
19	$\mathbf{C}_{55}^{1}$	$\mathbf{D}_{55}^{5}$	309	$\emptyset$	309	24.5 Sec.
20	$\mathbf{C}_{55}^{1}$	$\mathbf{D}_{55}^{6}$	71	$\emptyset$	79	16.7 Sec.

TABLE VII
SIMULATION RESULTS WITH COMPUTATION TIMES

Serial number	Problem Specification		Freq. of *hit*s in 1000 F^is	Prob. of *hit* in 1000 F^is	No. of F^is required for 99% *hit* probability
	Compatibility matrix	*Demand vector*			
1	$\mathbf{C}_{21}^{1}$	$\mathbf{D}_{21}^{1}$	68.475	≈100%	65
2	$\mathbf{C}_{21}^{1}$	$\mathbf{D}_{21}^{2}$	17.860	≈100%	255
3	$\mathbf{C}_{21}^{2}$	$\mathbf{D}_{21}^{1}$	02.337	90.4%	1970
4	$\mathbf{C}_{21}^{2}$	$\mathbf{D}_{21}^{2}$	00.109	10.4%	42,250
5	$\mathbf{C}_{21}^{3}$	$\mathbf{D}_{21}^{1}$	52.403	≈100%	85
6	$\mathbf{C}_{21}^{3}$	$\mathbf{D}_{21}^{2}$	48.450	≈100%	92
7	$\mathbf{C}_{21}^{5}$	$\mathbf{D}_{21}^{1}$	31.168	≈100%	145
8	$\mathbf{C}_{21}^{5}$	$\mathbf{D}_{21}^{2}$	00.618	46.1%	7450
9	$\mathbf{C}_{25}^{1}$	$\mathbf{D}_{25}^{3}$	53.333	≈100%	84
10	$\mathbf{C}_{55}^{1}$	$\mathbf{D}_{55}^{5}$	00.731	51.8%	6270

TABLE VIII
SIMULATION RESULTS WITH *hit* PROBABILITIES

We can see in Table.VII that, the solutions are further from the trivial lower bound when $c_{ii} = 4$ and $\alpha = 2$ (compatibility matrix $\mathbf{C}_{21}^{8}$, row 15 and 16) compared to when $c_{ii} = 5$ and $\alpha = 2$ (compatibility matrix $\mathbf{C}_{21}^{6}$, row 11 and 12), with both demand vectors $\mathbf{D}_{21}^{1}$ as well as $\mathbf{D}_{21}^{2}$. To our knowledge, there are no previous works to compare our results for these cases. For harder problems, though there are many near optimum solutions (please see Fig. 5 to Fig. 8), the number of actual *hit*s are low. Thus we can get a near optimum result quickly. But to get a *hit*, we need to generate a large pool of solutions. For example, for serial entry 10 of Table. VIII, out of 1 million trials there are 731 *hit*s at the optimum bandwidth solution of 309. But there are as many as 2751 near optimum solutions with bandwidth requirement of 310, just 1 carrier worse than the optimum.

C. Solution Distribution and Calculation of Hit *Probability*

For each of the 20 different channel assignment problems, listed in Table. VII, we created 10 million or more solutions. Number of solutions with particular bandwidth requirements were collected. Here we report the frequency distribution of the solutions. The results with 21-node network are shown in Fig. 5 to Fig. 8. We can see that the number of times the optimum solution (the trivial lower bound) and solutions very near to it is reached, is very high for simple problems (e.g. serial number 1 to 6 of Table.VII). For $\mathbf{C}_{21}^{7}$ with $\mathbf{D}_{21}^{1}$ too we get a large number of *hits* or near *hits*. For $\alpha = 2$, i.e., compatibility matrices $\mathbf{C}_{21}^{4}$, $\mathbf{C}_{21}^{6}$, and $\mathbf{C}_{21}^{8}$, though the obtained best solutions are quite near to the trivial lower bounds, the number of solutions in the vicinity of *hit* is very low, and is not noticeable in the corresponding figures, Fig. 6(c)(d), Fig. 7(c)(d) and Fig. 8(c)(d).

The results, illustrated in Fig. 5 to Fig. 8, are summarized in Table. VIII. The probability of at least one *hit* in a population of 1000 solutions is also included. The method of calculation of this probability is discussed in the next paragraph. In Table. VIII, we mentioned results only for those problems for which the number of *hits* in a pool of 1000 solutions is at least nearly one. Also included is the number of solutions needed for having at least one *hit* with a probability of 99%.

Suppose out of τ number of trial solutions, the total number of *hits* is ν. When sufficient number of independent samples are available, i.e. when the number of trial solutions τ is sufficiently large, we can say that in a single trial the probability of *hit* is ν/τ. The probability of *no-hit* in a trial (i.e. solution with longer than optimum frequency bandwidth) is $(1 - \nu/\tau)$. Thus the probability of *no-hit* in μ trials is $(1-\nu/\tau)^{\mu}$, whence the probability of at least one hit in μ trials is $(1-(1-\nu/\tau)^{\mu})$. Therefore the probability of at least one hit in 1000 trials is $(1-(1-\nu/\tau)^{1000})$. Now we calculate the required number of trials to have at least one *hit* with 99% probability. The probability of at least one *hit* in μ trials i.e. $(1 - (1 - \nu/\tau)^{\mu})$ is set to 0.99, and the corresponding μ is calculated and included in the last column of Table. VIII.

As mentioned in section IV.A, when $\rho = 2$ and $\alpha = 2$ i.e. when interference is more, the frequency allocation is more difficult. This is evident from row 3 and 4 of Table. VIII, where the compatibility matrix is $\mathbf{C}_{21}^{2}$. When C_{ii} is low and/or channel demands from different cells are more or less the same, the channel allocation problem becomes difficult. This is evident from row 8 of Table. VIII.

All the experiments, serial number 1 to 20 mentioned in Table. VII, were repeated 10 times or more, each time generating a pool of one million solutions. The solutions were always found to be confined within a well defined range of bandwidth. Also, the percentage of solutions with a particular bandwidth requirement remained almost constant over different trials. This indicates that the set of all permutations of N is partitioned into a very small number of subsets, and starting with any member of a particular subset, we get solution of same bandwidth. As we can see from Fig. 5 to Fig. 8, for all the problems, the number of such partitions is always limited within a few hundreds. To estimate the confidence interval of the outcome of our experiments, we performed experiment 2 of Table. VII fifty times, generating 1 million solutions at each trial. The average value of the number of *hits* was 178080 per million, with standard deviation only 340. Assuming normal distribution, the 99% *confidence interval* for the number of *hits* is only $\pm 2.576 \times 340/\sqrt{50} = \pm 123.8$ [24] (chap. 17) i.e. 0.00069 fraction of the average value. With this and from the consistency of the results over different trials for all the experiments, we can assure a high degree of confidence to the results stated in Table. VIII.

V. Conclusion and Discussion

In this paper we have proposed a very fast algorithm to solve channel assignment problem in mobile network. The idea is to generate a number of valid solutions of the problem, during which the optimization criterion of minimizing the bandwidth is not given any attention. Finally if the best of all solutions is selected, there is a high probability that the optimum or very near to optimum solution is obtained. From Table. VIII it is clear that for all the problems reported earlier, we could find optimum solutions in small number of trials with very high probability, and therefore it is very fast. The probability is much more when near optimal results are acceptable too, as is evident from Fig. 5 to Fig. 8.

One main question regarding the usability of the algorithm is, what is the probability of getting the optimal solution. We have shown that for all known problems it could achieve the optimum solution in a very short time with a high probability. The required number of random solutions, to get a *hit*, may be quite high for very hard problems. But depending on the amount of available computation time, even for a difficult problem a near optimal solution could always be found fast.

For very hard problems, instead of generating a large pool of solutions, it is also possible to improve a smaller population of solutions using genetic search. The difficult part is to define genetic operators, such that the solutions remain valid after crossover and mutation.

With the quadnary representation [0, +1, -1, +9] of frequency assignments, the information contained in the channel allocation matrix is in a format so that, it is simple to extend its use for dynamic allocation. In the solution, Table. III, a 0 at i^{th}-row and j^{th}-column indicates that frequency f_i could be allocated to the j^{th} node without causing any conflict to other existing assignments. Thus, it is possible to assign f_i to node node-j when needed, and turn "-1"s at proper locations by running PUT_MINUS1 routine (please refer to line 14 to 23 of the pseudocode in the appendix). This is to ensure that the next such frequency allotment will not create any conflict. The original solution matrix could be restored, once node-j's demand is returned to its original level. In this way *Call blocking* could be avoided in situations where demand varies within a small range over static demand. With this extension in mind, we can characterize the quality of channel allocation matrix primarily by the bandwidth requirement, and between two allocations of same bandwidth, by the number of "0" entries. In future we would like to simulate this dynamic allocation strategy and evaluate its *performance.*

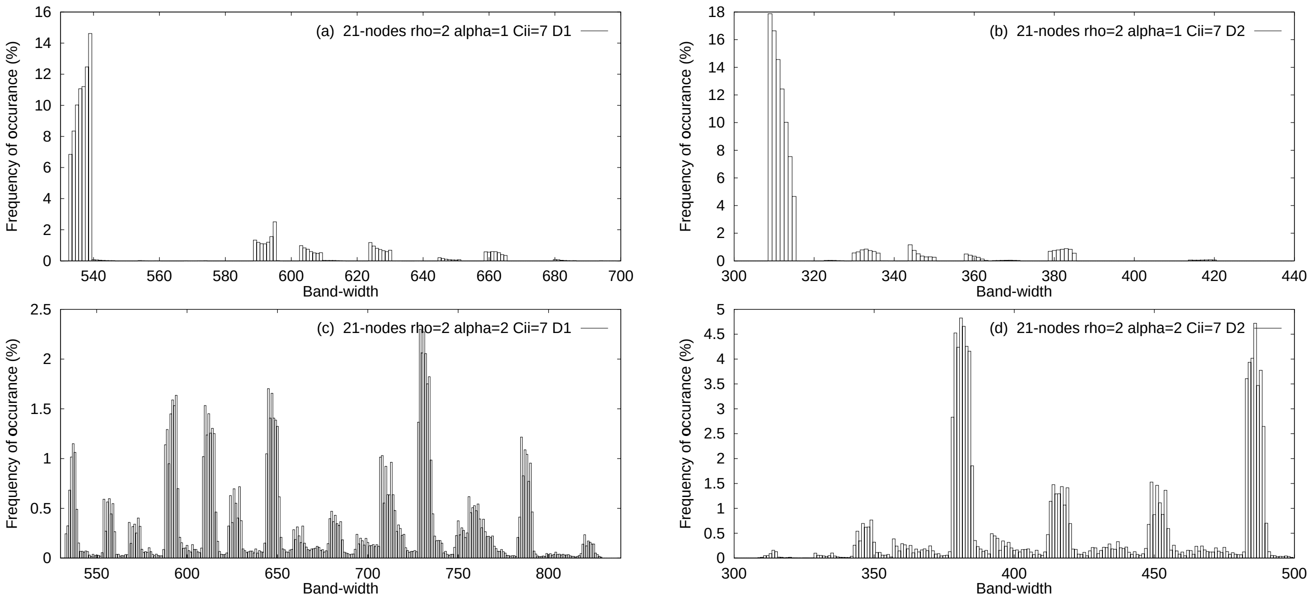

Fig. 5. Frequency Distribution of Solutions vs. corresponding bandwidth requirements (a) for $\mathbf{C}_{21}^{1}$ and $\mathbf{D}_{21}^{1}$, (b) for $\mathbf{C}_{21}^{1}$ and $\mathbf{D}_{21}^{2}$, (c) for $\mathbf{C}_{21}^{2}$ and $\mathbf{D}_{21}^{1}$, (d) for $\mathbf{C}_{21}^{2}$ and $\mathbf{D}_{21}^{2}$

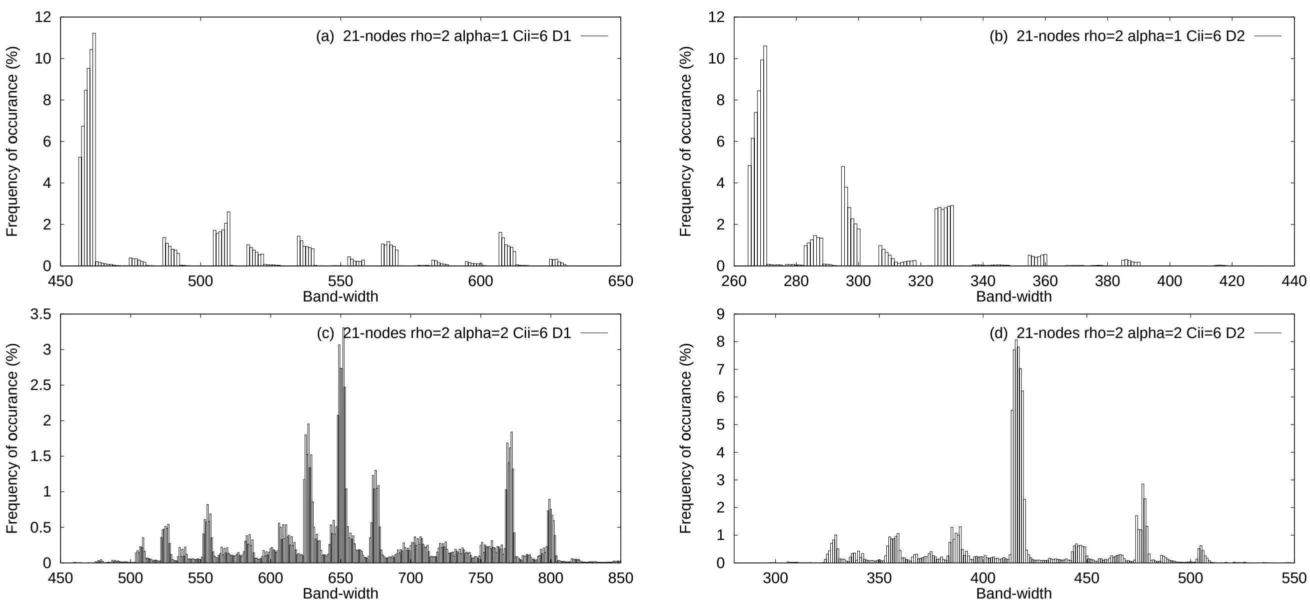

Fig. 6. Frequency Distribution of Solutions vs. corresponding bandwidth requirements (a) for $\mathbf{C}_{21}^{3}$ and $\mathbf{D}_{21}^{1}$, (b) for $\mathbf{C}_{21}^{3}$ and $\mathbf{D}_{21}^{2}$, (c) for $\mathbf{C}_{21}^{4}$ and $\mathbf{D}_{21}^{1}$, (d) for $\mathbf{C}_{21}^{4}$ and $\mathbf{D}_{21}^{2}$

References

[1] W. C. Y. Lee., "Mobile Cellular Telecommunications: Analog and Digital Systems," McGraw-Hill, 1995.

[2] A. Baiocchi, F. D. Priscoli, F. Grilli, and F. Sestini, "The Geometric Dynamic Channel Allocation as a Practical Strategy in Mobile Networks with Bursty User Mobility," *IEEE Trans. on Vehicular Technology*, Vol. VT-44, No. 1, pp. 14-23, February 1995.

[3] S. Tekinay and B. Jabbari, "Handover and Channel Assignment in Mobile Cellular Networks," *IEEE Communication Magazine*, pp. 42-46, November 1991.

[4] A. Gamst and W. Rave, "On frequency assignment in mobile automatic telephone systems," in *Proc. GLOBECOM '82*, pp. 309-315, 1982.

[5] D. C. Cox and D. O. Reudnik, "Increasing Channel Occupancy in Large-Scale Mobile Radio Systems: Dynamic Channel Assignments," *IEEE Trans. on Vehicular Technology*, Vol. VT-22, pp. 14-23, 1973.

[6] J. Tajima and K. Imamura, "A Strategy for Flexible Channel Assignment in Mobile Communication Systems," *IEEE Trans. on Vehicular Technology*, Vol. VT-37, 1988.

[7] Andreas Gamst,"Some Lower Bounds for a Class of Frequency Assignment Problems," *IEEE Trans. on Vehicular Technology*, Vol. VT-35, No. 1, pp. 8-14, February 1986.

[8] K. N. Sivarajan, R. J. McEliece, and J. W. Ketchum, "Channel assignment in cellular radio," in *Proc. 39th IEEE Veh. Technology Soc. Conference*, pp. 846-850, May 1989.

[9] D. Kunz, "Channel assignment for Cellular radio using neural networks," *IEEE Transaction on Vehicular Technology*, vol. 40, no. 1, pp. 188-193, February 1991.

[10] Hiroshi Tamura, M. Sengoku, S. Shinoda, and T. Abe, "Channel

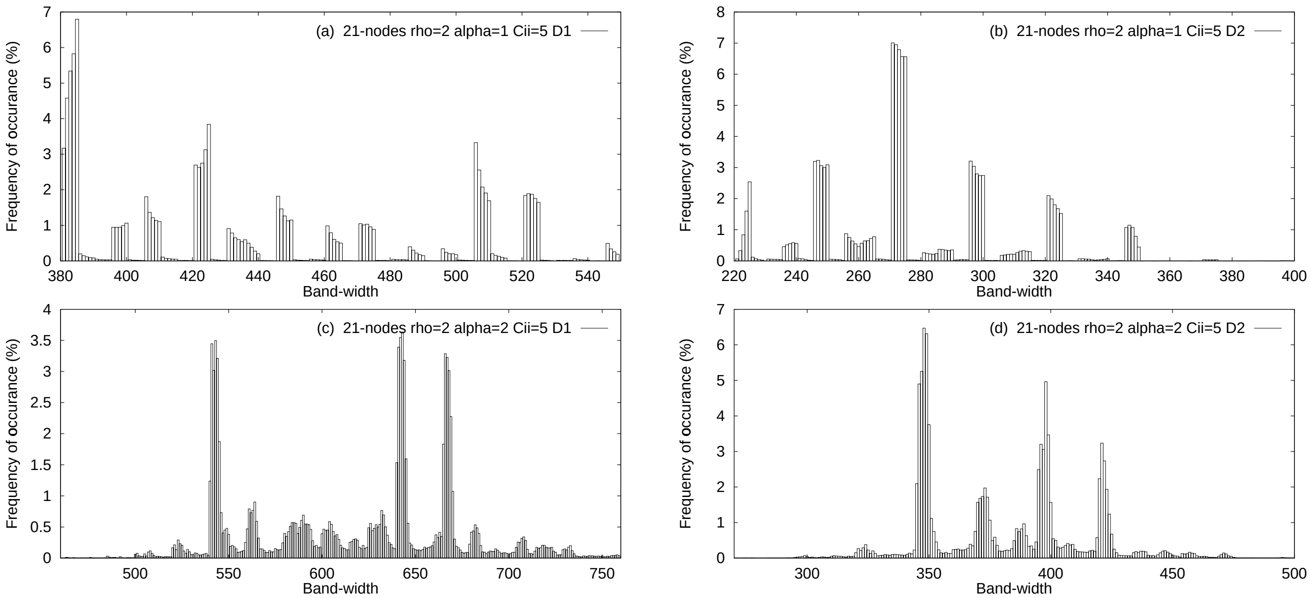

Fig. 7. Frequency Distribution of Solutions vs. corresponding bandwidth requirements (a) for $\mathbf{C}_{21}^{5}$ and $\mathbf{D}_{21}^{1}$, (b) for $\mathbf{C}_{21}^{5}$ and $\mathbf{D}_{21}^{2}$, (c) for $\mathbf{C}_{21}^{6}$ and $\mathbf{D}_{21}^{1}$, (d) for $\mathbf{C}_{21}^{6}$ and $\mathbf{D}_{21}^{2}$

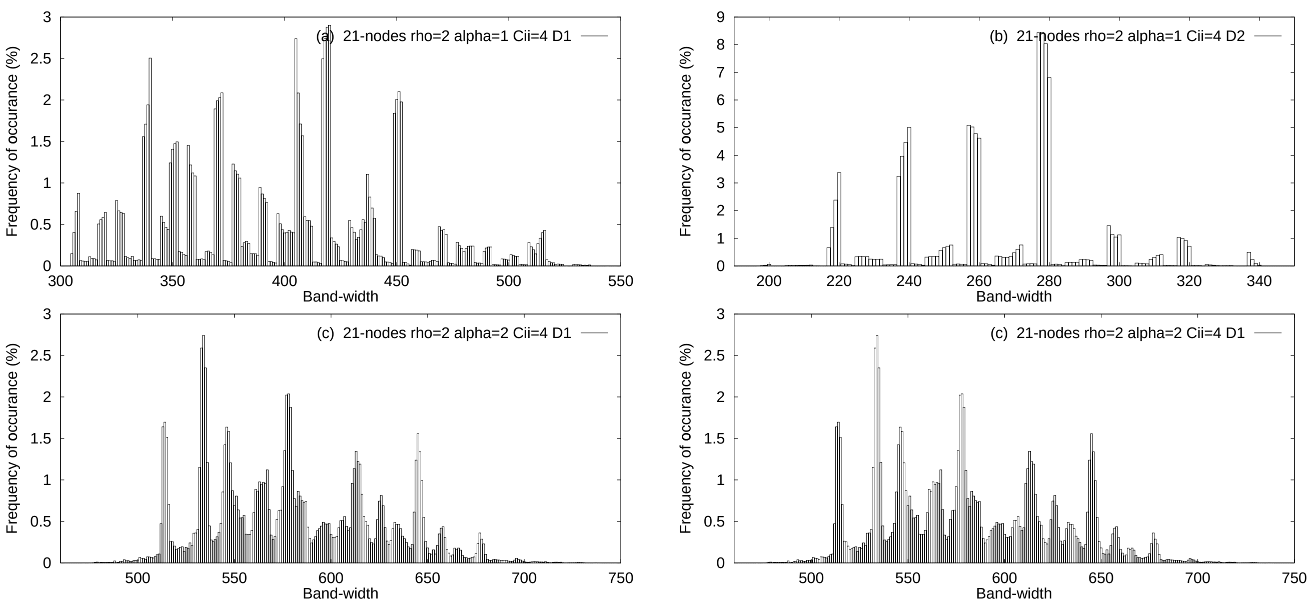

Fig. 8. Frequency Distribution of Solutions vs. corresponding bandwidth requirements (a) for $\mathbf{C}_{21}^{7}$ and $\mathbf{D}_{21}^{1}$, (b) for $\mathbf{C}_{21}^{7}$ and $\mathbf{D}_{21}^{2}$, (c) for $\mathbf{C}_{21}^{8}$ and $\mathbf{D}_{21}^{1}$, (d) for $\mathbf{C}_{21}^{8}$ and $\mathbf{D}_{21}^{2}$

Assignment Problem in a Cellular Mobile System and a New Coloring Problem of Networks," *IEICE Transactions*, Vol. E 74, No. 10, pp. 2983-2989, October 1991.

[11] N. Funabiki and Y. Takefuji, "A Neural Network parallel algorithm for channel assignment problems in cellular radio networks," *IEEE Transactions on Vehicular Technology*, vol. 41, no. 4, pp. 430-437, 1992.

[12] Peter T. H. Chan, M. Palaniswami, and David Everitt, "Neural Network-Based Dynamic Channel Assignment for Cellular Mobile Communication Systems," *IEEE Transactions on Vehicular Technology*, Vol. 43, No. 2, pp. 279-287, 1994.

[13] N. Funabiki and S. Nishikawa, "An Improved Neural Network for Channel Assignment Problems in Cellular Mobile Communication Systems," *IEICE Transactions*, Vol. E78-B, No. 8, pp. 1187-1195, August 1995.

[14] Kate Smith, and M. Palaniswami, "Static and Dynamic Channel Assignment Using Neural Network," *IEEE Journal on Selected Areas in Communication*, Vol. 15, No. 2, pp. 238-249, February 1997.

[15] Frank Box, "A Heuristic Technique for Assigning Frequencies to Mobile Radio Nets," *IEEE Trans. on Vehicular Technology*, Vol. VT-27, No. 2, pp. 57-64, May 1978.

[16] W. K. Hale, "Frequency assignment: Theory and applications," *Proc. IEEE*, vol. 68, pp. 1497-1514, December 1980.

[17] M. Duque-Anton, D. Kunz and B. Ruber, "Static and dynamic Channel Assignment Using simulated annealing," *Neural Networks in Telecommunications*, B. Yuhas and N. Ansari (eds.), Kluwer Academic Publishers, Boston, chap 10, pp. 191-210, 1994.

[18] R. Dorne and J. K. Hao, "An Evolutionary approach for frequency assignment in cellular radio networks," *Proceedings IEEE International Conference on Evolutionary Computation*,

vol. 2, pp. 539-544, 1995.

[19] W. K. Lai, and George G. Coghill, "Channel Assignment Through Evolutionary Optimization," *IEEE Transactions on Vehicular Technology*, Vol. 45, No. 1, pp. 91-95, February 1996.

[20] H. G. Sandalidis, P. P. Stavroulakis, J. Rodriguez-Tellez, "An Efficient Evolutionary Algorithm for Channel Resource Management in Cellular Mobile Systems," *IEEE Transactions on Evolutionary Computation*, Vol. 2, No. 4, pp. 125-137, November 1998.

[21] M. Sengoku, K. Itoh, and T. Matsumoto, "A Dynamic Frequency Assignment Algorithm in Mobile Radio Communication Systems," *IEICE Transactions*, Vol. E 61, No. 7, pp. 527-533, July 1978.

[22] R. W. Nettleton and G. R. Schloemer, "A high capacity assignment method for cellular mobile telephone systems," in *Proc. 39th IEEE Veh. Technology Soc. Conference*, pp. 359-367, May 1989.

[23] Kate A. Smith, "A Genetic Algorithm for the channel assignment problem," Technical report, Monash University. 1998.

[24] G. M. Clarke and D. Cooke, *A Basic Course in Statistics*. Edward Arnold. 3rd. edition. 1992.

APPENDIX

The pseudocode of the algorithm to generate random valid solutions:

Input: The channel allocation problem $(\mathbf{X},\ \mathbf{C},\ \mathbf{D})$, where $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\}$ is the cellular network.
N: Number of cells.
$\mathbf{C}[N][N]$: Compatibility matrix.
$\mathbf{D}[N]$: Demand vector.
P: Population size, the number of solutions to be created.

Output: A set of valid solutions of the channel assignment problem $(\mathbf{X},\ \mathbf{C},\ \mathbf{D})$. $F^1, F^2, \ldots, F^P$ are the P, population size, number of solutions of the input channel assignment problem. Each F^i is a $M \times N$ matrix, where N is the number of cells and M is the number of channels (frequencies).

```
CREATE_POP(X, C, D)
01  for p ← 1 to P
02      σ[N] ← SHUFFLE-LIST(N)
03      do for each node ν = σ[n], n = 1 to N
04          demand = D[ν]
05          for m = 1 to M
06              if F^p[m][ν] == 0
07                  F^p[m][ν] = 1
08                  demand ← (demand − 1)
09                  PUT_MINUS1(F^p[m][ν])
10                  if (demand == 0)
11                      break     /* for loop at line 05 */
12      MARK_UNUSED_FREQ(F^p)

SHUFFLE-LIST(N)
13  Return a random permutation of numbers 1 to N

PUT_MINUS1(F^p[m][ν])
14  for each node j = 1 to N
15      range ←C[j][ν]
16      if range == 0
17          continue
18      else
19          range_up = m − (range − 1)
20          range_down = m + (range − 1)
21          for i = range_up to range_down
22              if F^p[i][j] ≠ 1
23                  F^p[i][j] ← −1

MARK_UNUSED_FREQ(F^p)
24  Put '9' at column 1 of unused frequncies of F^p
```

It could be noted that the way *range_up* and *range_down* are calculated in lines 19 and 20 of the pseudocode, they may point beyond the top or below the bottom of the solution table respectively, resulting error during execution. These details are avoided in the pseudocode to keep it simple.

Complexity Analysis:
We here examine the complexity of the algorithm for creating a valid solution. In the above pseudocode, a loop on lines from 03 to 11 iterates for N number of times. Within this loop, for a particular node i channels have to be assigned d_i times, where d_i is the channel demand for node-i. Once a channel is assigned i.e. a "o" in F^p is turned to a "+1", PUT_MINUS1 function is called and executed. PUT_MINUS1 function has a loop on lines 14 to 23 with iterations for all nodes i.e. N times. Thus for assigning d_i number of channels to node-i the complexity for running PUT_MINUS1 is $O(N \times d_i)$. If the average of the demand for different channels is denoted by $\hat{d}$, i.e. $\hat{d} = (\sum_{i=1}^{N} d_i)/N$, then the complexity of the algorithm is $O(\hat{d} \times N^2) = O(N^2)$, where N is the number of nodes.

Goutam Chakraborty received his Ph.D. in 1993 from Tohoku University, Japan. Presently he is an Associate Professor in the Department of Software and Information Science, Iwate Prefectural University, Japan. His research interests are Soft Computing methods and their applications to solve different pattern recognition, classification, prediction, scheduling and optimization problems including applications to computer Networks, mobile networking and data mining.