Dynamic Channel Assignment Problem in Mobile Networks using Particle Swarm Optimization

Sanchita Ghosh[1], Amit Konar[2], Atulya Nagar[3]
BIT Mesra, India[1], Dept. of ETCE Jadavpur University, India[2], Liverpool Hope University[3]
Email: bij_arn@yahoo.com [1], konaramit@yahoo.co.in [2], nagar@hope.ac.uk[3]

Abstract—The problem of optimal channel assignment has become increasingly important because of available frequency spectrum and increasing demand for cellular communication services. This has been shown to be an Np complete optimization problem. Many heuristic approaches including neural network, simulated annealing and genetic algorithm have been used to solve it. In this paper we propose a novel and efficient channel assignment approach, particle swarm optimization, to seek a conflict free channel assignment such that demand is achieved. Simulations on six well-known benchmark problems showed that the PSO could effectively generate the low-band results.

Key Words – Mobile cellular network, Channel assignment problem, Hard and soft constraints, Allocation matrix, Fitness function, Particle swarm optimization, Call rejection probability.

I. INTRODUCTION

In recent years, there is a continuously growing demand for mobile telecommunication. However, because of limited number of usable frequencies that are necessary for the communication between mobile users and the base stations of cellular radio networks, the efficiency in spectrum utilization becomes an important issue in frequency planning. This issue is commonly referred to as channel assignment. The objective of a channel assignment algorithm is to determine a spectrum efficient allocation of channels to the cells while satisfying both the traffic demand and the electromagnetic compatibility constraints. In general, there are three types of constraints [5]:

Co-channel Constraint (CCC): The same channel cannot be simultaneously allocated to a pair of cells unless there is a minimum geographical separation between them.
Adjacent Channel Constraint (ACC): Adjacent channels cannot be assigned to a pair of cells unless there is a minimum distance between them.

Co-site Constraint (CSC): A pair of channels can be employed in the same cell only if there is a minimum separation in frequency between them. .The radio frequency propagation and the spatial density of the expected traffic determine these constraints.

The commonly used schemes employed for solving the CAP are:

Fixed Channel Assignment (FCA): In this simple but non-adaptive scheme [6], a set of channels is permanently allocated to each cell in the network.

Dynamic Channel Assignment (DCA): In this scheme [5], all the channels are kept in a *central pool* so that every cell has access to every channel; cells are assigned to calls as and when they arrive, depending on certain conditions. DCA may be *centralized*, when a single central controller decides the fate of all the calls in the entire network, or *distributed*, when the concerned base station on the basis of available information on the neighborhood takes the relevant decision.

Hybrid Channel Assignment (HCA): In this scheme [8], the set of available channels is partitioned into two subsets, one of which is allocated to the network according to the FCA scheme and the other as per DCA scheme.

In this paper, we propose a DCA scheme that employs Particle Swarm Optimization Algorithm. In the DCA scheme, whenever a fresh call arrives, the entire network should, ideally, be subjected to rearrangement of channel assignment to improve the quality of service and reduce the call blocking probability. However, as this is highly time-consuming and computationally complex, the rearrangement process is limited to the cell involved in new call arrival [1].

The subsections of this paper are organized as follows. In Sec. II, a mathematical model is proposed to represent the channel assignment problem. In Sec. III we describe the general method of particle swarm optimization and how to modify it to operate on discrete binary variables. Afterwards, in Sec. IV some

978-0-7695-3325-4/08 $25.00 © 2008 IEEE
DOI 10.1109/EMS.2008.63

results are gained by simulation experiment. Finally, Sec. V concludes the paper and gives the direction of future work.

II. A MATHEMATICAL MODEL

2.1. Cellular model assumptions

1. The geographical model is a set of contiguous, non-overlapping cells assumed to be of hexagonal shape and collectively forming a parallelogram (fig. 1).

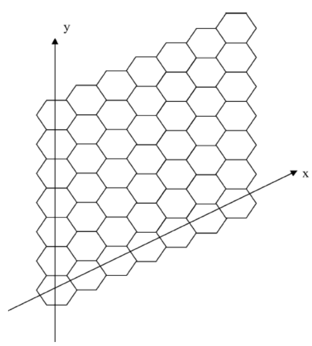

Fig. 1. Cellular network model (Vidyarthi *et al.*)

2. We have not considered the start-up situation but the situation at a certain intermediate time-instant *t* when the network is already serving a certain number of calls.
3. At time *t*, only *one* new call arrives at the cell, called the *host cell*. All other conditions in the entire network remain unaltered.
4. We set a minimum "reuse distance" *RD*, which represents the minimum allowable normalized distance between two cells which may use the same channel at the same time. This defines an "interference region" extending up to (*RD*–1) cells in all directions from the host cell.

2.2 Problem representation

2.2.1. The Allocation Matrix and the solution vector

Let N_{ce} denote the number of cells in a system and N_{ch} be the total number of channels. Then, for this system, an $N_{ce}\times N_{ch}$ matrix ***A***, called the *allocation matrix* [1], may be defined whose (*i*, *j*)-th element is given by

$$A_{ij}=\begin{cases}1 & \text{if channel } j \text{ is currently being used in cell } i,\\ 0 & \text{otherwise}\end{cases}$$

$\forall i=1,2,\dots,N_{ce},\ j=1,2,\dots,N_{ch}$; e.g. Suppose N_{ce} = 4, N_{ch} = 10 and channels 3,4 are currently in use in cell 1, channel 6 in cell 2, channel 1 in cell 3 and channels 3,7 in cell 4. Then the allocation matrix is given by:

Table 2.1

		Channels									
		1	2	3	4	5	6	7	8	9	10
Cells	1	0	0	1	1	0	0	0	0	0	0
	2	0	0	0	0	0	1	0	0	0	0
	3	1	0	0	0	0	0	0	0	0	0
	4	0	0	1	0	0	0	1	0	0	0

We assume that the new call demand is placed at cell *k* which is already serving [traf(*k*) – 1] calls where traf(*k*) denotes the total traffic load (ongoing + incoming) in cell *k* at time *t*. There are no pending calls and no ongoing call is terminated in the entire network. Our problem is to assign an available channel to the incoming call where possible reassignment of channels to the calls in progress in cell *k* may also be possible.

A candidate solution is a vector of channel numbers, denoted by *X*, of length traf(k), which represents a set of channels assigned to the traf(k) number of calls at cell *k* at the time-instant concerned [1].

For each candidate solution, the information about the new channel allocation in cell *k* is stored in is a {0,1}-valued vector of length N_{ch}, denoted by *V;* e.g. if *k* = 4, N_{ch} = 10 and a candidate solution is [1 2 7], then *V* = [1 1 0 0 0 0 1 0 0 0]. Clearly, *V* is a one-to-one function of *X*.

2.2.2. Formulation of the fitness function

We use the *fitness function*, suggested by BBB [2], whose minimum value is likely to correspond to a good solution (X_{best}) for the new channel allocation in cell *k*. This function is a linear combination of the following terms:

1. $$f_1(X)=\sum_{j=1}^{N_{ch}}\sum_{\substack{i=1\\ i\neq k}}^{N_{ce}} V_j A_{ij}\,\text{interf}(i,k)$$

   where V_j denotes the *j*th element of *V* and interf(*i*,*k*) is a function which returns 1 if the cells *i* and *k* interfere, otherwise it returns 0. This term contributes 1 for each cell, interfering with *k*, which uses a channel employed in *k*. It thus ensures that solutions with no interference give *better* (smaller) fitness values.

2. $f_2(X) = -\sum_{j=1}^{N_{ch}} \sum_{\substack{i=1 \\ i \neq k}}^{N_{ce}} V_j A_{ij} \frac{1 - \text{interf}(i,k)}{d_{ik}}$

where d_{ik} is the normalized Euclidean distance between the centers of the cells i and k. This term takes care of the packing condition. Clearly interf(i,k) = 0 if $d_{ik} \geq RD$.

3. $f_3(X) = -\sum_{j=1}^{N_{ch}} V_j A_{kj}$

which subtracts 1 whenever a channel already being used by cell k, before the arrival of the new call, is considered in the candidate solution (i.e. in the new configuration) so that a mobile terminal being served changes its channel as seldom as possible.

4. $f_4(X) = \sum_{j=1}^{N_{ch}} \sum_{\substack{i=1 \\ i \neq k}}^{N_{ce}} V_j A_{ij} \{1 - \text{res}(i,k)\}$

where res(i,k) is a function which returns 1 if cells i and k belong to the same "reuse scheme" , else it returns 0.

N.B.: The term of the form $\left(\text{traf}(k) - \sum_{j=1}^{N_{ch}} V_j\right)^2$ used by BBB, has been discarded because in our scheme, this term is always equal to zero.

Finally, the fitness function F (X) is given by

$$F(X) = W_1.f_1(X) + W_2.f_2(X) + W_3.f_3(X) + W_4.f_4(X)$$

where W_1, W_2, W_3, W_4 are *weights* that determine the importance of various terms. Clearly, $f_1(X)$ accounts for the *hard constraint* that should be given primacy over the other terms that are associated with the *soft constraints*. Thus, we use the same set of coefficient values as BBB i.e. W_1=7000, W_2=1.2625, W_3=0.01, W_4= 4.17625. Our task now is to determine the minimum of this fitness function.

III. PARTICLE SWARM OPTIMIZATION (PSO) ALGORITHM

PSO [9], [10], [11] is a population-based stochastic optimization technique, developed by Dr. Eberhart & Dr. Kennedy. The problem is to determine the *global optimum* (maximum or minimum) of a function of n independent variables x_1, x_2, x_3……, x_n , mathematically represented as $f(\vec{X})$, where $\vec{X} = (x_1, x_2, x_3, ..., x_n)$ is called the *parameter vector*. If all components of $\vec{X}$ assume real values only, then the task reduces to locating a particular point $\vec{X}$ in the n-dimensional hyperspace for which the function value $f(\vec{X})$ is either a minimum or maximum in the search range.

PSO is, in principle, a multi-agent parallel search technique. The initial population consists of a convenient number of "particles"; particles are conceptual entities that "fly" through the multi-dimensional search space as the algorithm progresses.

Each particle P has two state variables:

1. Its current position $\vec{X}(t)$.
2. Its current velocity $\vec{V}(t)$.

The position vector of each particle with respect to the origin of the search space represents a *candidate solution* of the search problem. Each particle also a small memory comprising:

1. Its personal best position experienced so far, denoted by $\vec{p}(t)$.
2. The global best position found so far, denoted by $\vec{g}(t)$.

For each particle, each component of the initial position vector is selected at random from a predetermined search range while each velocity component is initialized by choosing at random from the interval $[-V_{\max}, V_{\max}]$ where $V_{\max}$ is the maximum possible velocity of any particle in any dimension; the initial settings for $\vec{p}(t)$ and $\vec{g}(t)$ are taken as $\vec{p}(0) = \vec{g}(0) = \vec{X}(0)$ for all particles. After the particles are initialized, the iterative optimization process begins, where the positions and velocities of all the particles are altered by the following recursive equations (1) & (2), who for the time being assumed to progress in discret (unit) steps. The equations are presented for the d-th dimension of the position and velocity of the i-th particle.

$$\boldsymbol{V_{id}(t+1) = \omega V_{id}(t) + C_1\varphi_1. (p_d(t) - X_{id}(t)) + C_2\varphi_2. (g_d(t) - X_{id}(t))} \quad \textbf{... ... ... (1)}$$

$$\boldsymbol{X_{id}(t+1) = X_{id}(t) + V_{id}(t+1)} \quad \textbf{... ... ... (2)}$$

where the algorithmic parameters are defined as :

ω: inertial weight factor that determines the effect of past velocity on present velocity.

C_1,C_2 : two constant multiplier terms known as "self confidence" and "swarm confidence" which respectively determine the influence of $\vec{p}(t)$ and $\vec{g}(t)$ on the velocity update formula.

φ_1, φ_2: two uniformly distributed random numbers.

Typically, this process is iterated for a certain predetermined number of time steps, or until some acceptable solution has been found by the algorithm or until an upper limit of CPU usage has been reached. Once the iterations are terminated, most of the particles are expected to converge to a small region surrounding the global optimum of the search space.

In our problem, the search range for position vector component in each dimension is the set of channel numbers $[1,2,\ldots,N_{ch}]$, $V_{\max} = (N_{ch} - 1)$. We take $\omega = 0.729$, $C_1 = C_2 = 1.494$, $0 < \varphi_1, \varphi_2 \leq 1$. Each run of the algorithm consists of 5000 iterations.

IV. DESCRIPTION OF BENCHMARK PROBLEMS

For simulation purposes, we have considered three different classes of problems available in the literature:

1. The first class consists of the data set, denoted as EX1, suggested by Sivarajan *et al* [4] as well as a slightly larger extension of EX1, denoted as EX2 [7]. This class of problems is used for experimental purposes only as its size is unrealistically small.
2. The next class comprises test problems denoted by HEX1-HEX4 [7], based on a 21-cell system. However, in this paper, we have discarded HEX2 and HEX4 because, in these problems, ACC has been taken into account while, in our model, we have assumed the absence of it.
3. The final set of problems viz. KUNZ1-KUNZ4 [7] was generated from the topographical data of an actual 24 km$\times$21 km area around Helsinki, Finland, as studied by Kunz. However, we have used only KUNZ1 and KUNZ2 in this paper and have neglected the occurrence of adjacent-channel interference due to our simplified model.

Each of these problems is specified in the literature in terms of the number of cells in the network (N_{ce}), the number of channels in the pool (N_{ch}) and a *demand vector* $\boldsymbol{D}$ which is a vector whose ith element denotes the traffic demand in cell i, $i = 1,2,\ldots, N_{ce}$. The descriptive details of each problem are tabulated in Table 4.1.

Table 4.1

Problem	N_{ce}	N_{ch}	*Demand vector* $\boldsymbol{D}$
EX1	4	11	1,1,1,3
EX2	5	17	2,2,2,4,3
HEX1	21	37	2,6,2,2,2,4,4,13,19,7,4,4,7,4,9, 14,7,2,2,4,2
HEX3	21	21	1,1,1,2,3,6,7,6,10,10,11,5,7,6,4, 4,7,5,5,5,6
KUNZ1	10	30	10,11,9,5,9,4,5,7,4,8
KUNZ2	15	44	10,11,9,5,9,4,5,7,4,8,8,9,10,7,7

However, in order to make each problem compatible with our model, we made the following modifications:

1. As we have assumed that all cells are arranged in the form of a parallelogram, we express the given N_{ce} of each problem in the form $r \times c$, where r, c are integers, and hence determine the configuration of the cellular network by setting the number of rows to r and the number of all columns to c. We maintain $r \leq c$ and try to choose r and c in such a way as to keep the rows and columns balanced as far as possible.
2. We arbitrarily select a cell k and assume that, just before a call demand arrives at this cell at time t, D_i calls were already in progress in the ith cell, $i = 1,2,\ldots,N_{ce}$, $i \neq k$, and $(D_k - 1)$ calls in the kth cell, where D_j is the jth entry of the *demand vector* $\boldsymbol{D}$, $j = 1,2,\ldots,N_{ce}$. Accordingly, we have constructed for each problem an $N_{ce} \times N_{ch}$ allocation matrix, avoiding co-channel interference, which describes the status of ongoing calls in each cell before the new call arrival and thus embodies the *initial condition*. e.g. in EX1, we assume that, when a fresh call arrives at cell 4 at time t, 1 call in each of cells 1 to 4 and 2 calls in cell 4 were going on with channels 1, 5, 9 assigned to cells 1, 2, 3 respectively and channels 2, 7 being used by cell 4. The relevant details are provided in Tables 4.2, 4.3, 4,4,4.5:

Problem	r	c	cell k	Number of calls in service in cells 1,2,3,…at time t.
EX1	2	2	4	1,1,1,2
EX2	1	5	4	2,2,2,3,3
HEX1	3	7	6	2,6,2,2,2,3,4,13,19,7,4,4,7,4, 9,14,7,2,2,4,2
HEX3	3	7	5	1,1,1,2,2,6,7,6,10,10,11,5,7, 6,4,4,7,5,5,5,6
KUNZ1	2	5	7	10,11,9,5,9,4,4,7,4,8
KUNZ2	3	5	7	10,11,9,5,9,4,4,7,4,8,8,9,10, 7,7

Table 4.2

N.B. r = Number of rows; c= Number of columns;

cell k is the host cell at time t.

Table 4.3

Prob.	*Allocation matrix*
EX1	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0
EX2	1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0

Table 4.4

Allocation matrix for HEX1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0
0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 1
1 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 1
0 0 0 0 1 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0 1 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1
0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Allocation matrix for HEX3
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
1 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 1 0
0 1 0 0 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0
0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 1
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0
0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0
1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 1
0 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0
0 1 0 1 0 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 1
0 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0
1 0 0 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0
0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0
0 1 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 1

Table 4.5

Allocation matrix for KUNZ1
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0
0 1 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 1 0 1 0 1 0 1
0 0 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0
1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0
0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0
1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0
Allocation matrix for KUNZ2
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 1
0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 1 0 0 1 0 1 0
0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0
1 0 0 0 0 0 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 1 0 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1
0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0

4.1 Simulation results

We wrote a program in C for minimising the fitness function $F(X)$ using PSO. For each problem, we performed at least 100 simulations under the same initial conditions (i.e. for all the simulations of a particular problem, we assumed that a call demand arrives in the same cell k when the network has the same allocation matrix). Each simulation is identical to taking the average of 10 runs of the PSO algorithm. If, on applying the update formulas, any component of position or velocity exceeds the highest permissible value or becomes smaller than the lowest allowed value, it is fixed at the corresponding highest or lowest value respectively. However, the final result is a vector of real numbers between 1 and N_{ch} but only integral values (channel numbers) are admissible. So, we replace (*round off*) each real-valued component of the solution vector by its "*floor*" or "*ceiling*" according as its fractional part is smaller or larger than 0.5. Each time that a simulation results in a solution, which violates CCC, the call is rejected. As a parameter for determining the effectiveness of our method, we define the *cell-based call rejection probability* (*CCRP*) as

$$CCRP = \frac{\text{No. of simulations in which the incoming call is rejected in the host cell considered}}{\text{Total no. of simulations}}$$

It is worthwhile to mention here that this parameter *CCRP* is based on but different from the *call blocking probability* used in the [1], [2]; the former characterizes *a particular cell* under a *given initial condition* while the latter characterizes the *cellular network as a whole*.

The computation of *CCRP* for each of the problems solved is presented in Table 4.6 and Fig. 3. Fig. 2. shows the convergence of CCRP for one of the problems.

Table 4.6

Problem	*Host cell* k	*Number of simulations*	*Number of times the call is rejected*
EX1	4	112	32
EX2	4	100	28
HEX1	6	104	21
HEX3	5	105	32
KUNZ1	7	110	39
KUNZ2	7	100	20

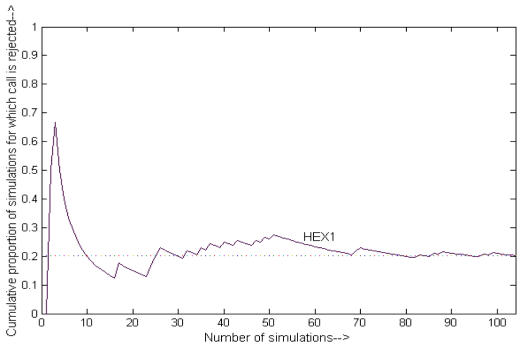

Fig.2. Plot of the cumulative proportion of simulations for which call is rejected in cell 6 vs. the number of simulations in HEX1

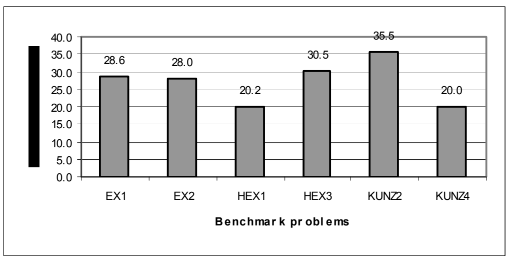

Fig.3. Bar chart showing call rejection probabilties in the respective host cells in the different problems considered

V. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

In this paper, we have applied PSO, which is known to perform better than many other optimization algorithms (such as Genetic Algorithm), to the CAP and have obtained reasonably good results in terms of the CCRP, which is a new parameter we have proposed.

However, a real cellular network is more complex and the calls arrive in different cells in a more random manner following a certain distribution. We are currently investigating how this algorithm will perform under such dynamically changing traffic load in a more realistic situation.

VI. REFERENCES

[1] Geetali Vidyarthi, Alioune Ngom, and Ivan Stojmenovic´,"A Hybrid Channel Assignment Approach Using an Efficient Evolutionary Strategy in WirelessMobile Networks", *IEEE Transactions 0n Vehicular Technology*, Vol. 54, NO. 5, September 2005, pp. 1887-1895.

[2] Roberto Battiti, Alan A. Bertossi, Mauro Brunato, "Cellular Channel Assignment : a New Localized and Distributed Strategy", *Mobile Networks and Applications 6(2001), —Kluwer*, December 17, 2001,pp. 493–500.

[3] Sa Li, Lipo Wang,"Channel Assignment for Mobile Communications Using Stochastic Chaotic Simulated Annealing", J. Mira and A. Prieto (Eds.): IWANN 2001, LNCS 2084, pp. 757-764, 2001.

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

[5] K. N. Sivarajan, R. J.McEliece, and Ketchum, "Dynamic channel assignment in cellular radio," in *Proc. 40th IEEE Vehicular Technology Conf.*, 1990, pp. 631–637 .

[6] C. Y. Ngo and V. O. K. Li, "Fixed channel assignment in cellular radio networks using a modified genetic algorithm", *IEEE Trans. Veh. Technol.* vol. 47, no. 1, pp. 163-172, Feb. 1998.

[7] K. Smith and M. Palaniswami, "Static and dynamic channel assignment using neural network", *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 2, Feb 1997.

[8] T. J. Kahwa and N. D. Georgans, "A hybrid channel assignment schemes in large-scale, cellular structured mobile communication systems," *IEEE Trans. Commun.*, vol. COM-26, pp. 432–438, 1978.

[9] J. Kennedy, R. Eberhart, "Particle Swarm Optimization", *Proceedings of IEEE International conference on Neural Networks*, 1995, 1942-1948.

[10] J. Kennedy, and R. C. Eberhart, *Swarm Intelligence*, ISBN 1-55860-595-9, Academic Press (2001).

[11] S. Das, A. Konar, U. K. Chakraborty, "Improving Particle Swarm Optimization with Differentially Perturbed Velocity", *ACM-SIGEVO Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2005)*, Washington DC, June, 2005.