CHAPTER 4

COST OPTIMIZATION IN HIERARCHICAL MOBILE ACCESS NETWORK

4.1 INTRODUCTION

Previous chapter discussed shortcomings in cost optimization of non-hierarchical mobile access network. Non-hierarchical mobile access network is a fading architecture with several issues like (i) no concrete levels at access layer (ii) less number of network components are involved, (iii) cost optimization based on limited number of network components, and (iv) cost optimization is NP hard and needs better heuristic algorithm. This chapter discusses solutions to overcome the above issues by using hierarchical mobile access network. So that, performance in modern advanced mobile networks can be improved by incorporating these proposed techniques.

The primary goal of this chapter is to study the performance evaluation of cost optimization in hierarchical mobile access network. To achieve it, first section addresses mathematical modelling of cost optimization problem in hierarchical mobile access network along with constraints like connectivity and capacity of switches. Mixed integer programming is used to formulate cost optimization problem in hierarchical mobile access network. The mathematical modelling of cost optimization problem of hierarchical mobile access network is a complex process, as there exists hierarchy of layers in access network.

Second section presents Extended Evolutionary Heuristic Algorithm (EEHA) for cost optimization of hierarchical mobile access network. The proposed algorithm consists of three phases namely initialization phase, global search phase and local search phase. Third section presents evaluation of cost optimization using experiments conducted on proposed algorithm. The experimentation is conducted on various sets of network, ranging from smaller to larger size. Fourth, this chapter presents visualization of final output for optimized network using MATLAB software.
The second goal of this chapter is to identify gaps in the cost optimization of hierarchical mobile access network. Some of the issues are strictly due to hierarchical nature of access network and its vulnerability to failures, various types of failures, single homed network components, fault tolerance along with cost optimization, and redesigning solution approach to accommodate cost optimization along with fault tolerance in mobile access network. The second section of this chapter elaborates in detail about these issues. To begin with, chapter elaborates performance evaluation of cost optimization in hierarchical mobile access network.

4.2 PERFORMANCE EVALUATION OF COST OPTIMIZATION IN HIERARCHICAL MOBILE ACCESS NETWORK

This section elaborates the performance evaluation of cost optimization in hierarchical mobile access network in three steps. First step is mathematical modelling of cost optimization in hierarchical mobile access network. Second step is using EEHA algorithm to solve the cost optimization problem. Third step is result analysis based on objective cost and, computation time. Further, the later section discusses visualization of optimized network using MATLAB.

4.2.1 MATHEMATICAL MODELING OF HIERARCHICAL MOBILE ACCESS NETWORK

Chapter 2 discussed in detail about architecture of hierarchical mobile access network. In short, hierarchical mobile access network consists of three major components namely NodeB, RNC and MSC. These three components interconnect in hierarchical manner using T1/E1 transmission cables. In addition, hierarchical mobile access network comprises three levels of access network namely NodeB level, RNC level and MSC level. For example, in practical scenario of 3G wireless access network, RNC controls all NodeBs and MSC controls all RNCs.

The cost optimization problem of hierarchical mobile access network is a complex process. For this purpose, efficient mixed integer programming is used to formulate this problem. However, the mixed integer programming has to incorporate following parameters while mathematical modelling hierarchical access subsystem.
• **Single homed:** The NodeBs are single homed. Each NodeB has only one network interface. Therefore, NodeB connects to only one RNC in next level of hierarchy. Hence, the mathematical modelling has to ensure that all Nodes are single homed.

• **Multilayered:** The access network has three levels of hierarchy namely NodeB level, RNC level and MSC level. Hence, the mathematical modelling has to exhibit layered optimization.

• **Technical constraints like connectivity, capacity of switches:** The optimization problem has to satisfy two main constraints namely connectivity and capacity. The connectivity constraint ensures that each node of access network has to be connected to at least one switch. The capacity constraint ensures that each request from node is within the limit of capacity of serving switch. Hence, the mathematical modelling has to exhibit these technical constraints.

The mathematical modelling of hierarchical mobile access network is carried out in three steps viz (i) minimization of cost at NodeB level of access network (ii) minimization of cost at RNC level of access network and (iii) summative cost minimization of access network. Table 4.1 depicts various symbols and meaning of the symbols used in mathematical modelling.
Table 4.1: Symbols and Meanings

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Meanings</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Total Number of NodeBs</td>
</tr>
<tr>
<td>J</td>
<td>Total Number of RNCs</td>
</tr>
<tr>
<td>K</td>
<td>Total Number of MSCs</td>
</tr>
<tr>
<td>s</td>
<td>1 for primary link and 2 for backup link of NodeB</td>
</tr>
<tr>
<td>$\lambda_i$</td>
<td>Call servicing capacity of single homed NodeB$_i$</td>
</tr>
<tr>
<td>$\lambda_{is}$</td>
<td>Call servicing capacity of dual homed NodeB$_i$</td>
</tr>
<tr>
<td>$\omega_j$</td>
<td>Call handling capacity of RNC$_j$</td>
</tr>
<tr>
<td>$\mu_k$</td>
<td>Call handling capacity of MSC$_k$</td>
</tr>
<tr>
<td>$\alpha_i$</td>
<td>Cable cost for existing link between cells NodeB$_i$ and switch RNC$_j$</td>
</tr>
<tr>
<td>$\beta_i$</td>
<td>1 if NodeB$_i$ and RNC$_j$ are connected; zero otherwise</td>
</tr>
<tr>
<td>$\beta_{is}$</td>
<td>1 if dual homed NodeB$_i$ and RNC$_j$ are connected; zero otherwise</td>
</tr>
<tr>
<td>$v_{is}$</td>
<td>0 for single homing and 1 for dual homing of NodeBs</td>
</tr>
</tbody>
</table>
Firstly, the objective function at NodeB level is defined as minimization of objective cost by optimal assignment of NodeBs to RNCS while satisfying the connectivity and capacity constraint. The mathematical representation of the same is given below.

**Objective function for connectivity at NodeB level:**

Minimize \( \delta = \sum_{i=1}^{I} \sum_{j=1}^{J} \alpha_{i}^{j} \beta_{i}^{j} \) \hspace{1cm} (4.1)

Subject to:

\[ \sum_{i=1}^{I} \beta_{i}^{j} = 1 \quad \text{for } j = 1,2,\ldots,J \] \hspace{1cm} (4.2)

\[ \sum_{i=1}^{I} \lambda_{i}^{j} / \beta_{i}^{j} \leq \omega_{j} \quad \text{for } j = 1,2,\ldots,J \] \hspace{1cm} (4.3)

Secondly, the objective function at RNC level is defined as minimization of objective cost by optimal assignment of RNCs to MSCs while satisfying the connectivity and capacity constraint. The mathematical representation of the same is given below.

**Objective Function for connectivity at RNC Level:**

Minimize \( \theta = \sum_{j=1}^{J} \sum_{k=1}^{K} \alpha_{j}^{k} \beta_{j}^{k} \) \hspace{1cm} (4.4)

Subject to:

\[ \sum_{j=1}^{J} \beta_{j}^{k} = 1 \quad \text{for } k = 1,2,\ldots,K \] \hspace{1cm} (4.5)

\[ \sum_{j=1}^{J} \omega_{j}^{k} \beta_{j}^{k} \leq u_{k} \quad \text{for } k = 1,2,\ldots,K \] \hspace{1cm} (4.6)
Capacity Constraint for single home-multilayered hierarchical access network:

The number of connectivity requested through NodeB of each cells for a particular RNC, must be less than or equal to the remaining call handling capacity \( \omega_j \) currently available at RNC \( j \).

- **NodeB Level:**

\[ \sum_{i=1}^{J} \lambda_i^j \beta_i^j \leq \omega_j \quad \text{for } j = 1, 2, ..., J \]  \hspace{1cm} (4.7)

The number of connectivity requested through RNC of each cells for a particular MSC, must be less than or equal to the remaining call handling capacity \( \mu_k \) currently available at MSC\( k \).

- **RNC Level:**

\[ \sum_{j=1}^{J} \omega_j^k \beta_j^k \leq \mu_k \quad \text{for } k = 1, 2, ..., K \]  \hspace{1cm} (4.8)

By considering equation (4.1) and (4.4), final objective function for the hierarchical mobile access network,

\[ \xi = \delta + \theta, \text{ for } k = 1, 2, ..., K \]  \hspace{1cm} (4.9)

where \( \delta = \min \left( \sum_{i=1}^{J} \sum_{j=1}^{I} \alpha_i^j \beta_i^j \right) \) and \( \theta = \min \left( \sum_{j=1}^{J} \sum_{k=1}^{K} \alpha_j^k \beta_j^k \right) \)  \hspace{1cm} (4.10)

Subject to:

\[ \sum_{i=1}^{I} \beta_i^j = 1 \quad \text{for } j = 1, 2, ..., J \]  \hspace{1cm} (4.11)

\[ \sum_{j=1}^{J} \beta_j^k = 1 \quad \text{for } k = 1, 2, ..., K \]  \hspace{1cm} (4.12)

\[ \sum_{i=1}^{I} \lambda_i^j \beta_i^j \leq \omega_j \quad \text{for } j = 1, 2, ..., J \]  \hspace{1cm} (4.13)

\[ \sum_{j=1}^{J} \omega_j^k \beta_j^k \leq \mu_k \quad \text{for } k = 1, 2, ..., K \]  \hspace{1cm} (4.14)
However, the algorithmic complexity of finding the optimal configuration is high. Further, this problem of minimizing the optimization cost without violating any of the capacity and connectivity constraint is NP hard [14, 15]. The conventional polynomial algorithms have the limitation on number of nodes and constraints that can be used. Hence, the next section proposes extended evolutionary heuristic algorithm to find the feasible solution for the above problem of NP hard.

4.2.2 EXTENDED EVOLUTIONARY HEURISTIC ALGORITHM (EEHA)

The major demerit of conventional methods like simplex and search algorithms is that, they work efficiently only for a deterministic number of network nodes. Further, they need a prior knowledge about entire network. Also, these deterministic algorithms fail to exploit the local information in large-scale networks.

This work, proposes evolutionary heuristic algorithm, to overcome disadvantages of conventional methods. Evolutionary algorithms [14] with their powerful local search techniques are used widely for solving the cost optimization in various real life problems. In addition, in this work, powerful local search techniques like redistribution, reassignment and shuffle tolerance limit are used.

During global search, it uses evolutionary techniques like multiple generations, where each generation has multiple populations. Advantage of this proposed technique is to overcome the problem of local search being struck in the local optimum. Convergence of the algorithm is comparatively faster than naïve algorithms. The experimental results show good feasible solution even under restrictions like capacity constraint and connectivity constraint.

This study targets a multilevel optimization of access networks. In order to perform optimizations in multilevel, first we execute our proposed extended evolutionary heuristic algorithm at NodeB level. The obtained optimal assignment of NodeB to RNC at this NodeB level becomes input to the RNC level assignment. The algorithm is repeated for RNC level assignment where RNCs are optimally assigned to MNCs. The proposed algorithm consists of three phases namely initialization phase, global search phase and local search phase.
Initialization Phase

In initialization phase, various inputs like number of NodeBs, RNCs and matrices like distance matrix $\beta_i^j$ and flow matrix $\alpha_i^j$ are initialized. The various capacity vectors like $\lambda_i$ of NodeBs, $\omega_j$ of RNCs, $\mu_k$ of MSCs are also initialized. Consider, $G$ represents generation and $m$ represents number of generation. Each generation consists of $n$ populations. Each population has $I$ NodeBs. Now, each generations $G_m$ is initialized with set of populations $\{P_1^0, P_1^1, P_1^2, \ldots, P_1^n\}$. For $n^{th}$ populations, $I$ number of NodeBs are initialized with $j$ RNCs. Then the objective function $(P_i^n)_{obj}$ of each population is calculated using Euclidean distance between NodeB and its assigned RNCs. Next, the minimum value ($S_{new}$) among these sorted objective values is identified. Finally, global best solution ($S_{best}$) is assigned with initial best solution ($S_{new}$) obtained from initial random populations.

Global Search Phase

During global search phase, the residual capacity ($R_j$) of each RNC is updated. Here, global search is performed until the number of generations does not exceed the maximum limit of generations. Next, based on initial best solution ($S_{best}$), local search is performed. Then, the objective function ($S_{local}$) of local search procedure is estimated using redistribution and reassignment mechanisms. The obtained objective values of populations are sorted in order to find the minimum value.

The current minimum ($S_{new}$) is compared with local minimum value ($S_{local}$). If the objective cost is lesser than current minimum, then local minimum objective value ($S_{local}$) is updated with the new current minimum ($S_{new}$). Further, if the local minimum value is less than global minimum value ($S_{best}$), best global value is also updated with current local minimum. Feasible solution is the assignment corresponding to best global value. In this regard, Figure 4.1 depicts the pseudo code for the global search procedure of the extended evolutionary heuristic algorithm.
$G_0 \leftarrow \{p_i^0, p_i^1, p_i^2, \ldots, p_i^n\}$

$p_i^n \leftarrow \text{rand}(j), \forall i \in I, \forall j \in J$

$(p_i^n)_{obj} \leftarrow \sum_{i=1}^{l} \sum_{j=1}^{j} \alpha_i^j \beta_i^j$

$s_{new} \leftarrow \min\left(\text{sort}\left((p_i^n)_{obj}\right)\right)$

$s_{best} \leftarrow s_{new}$

$r_j \leftarrow \omega_j - \sum_{i=1}^{l} \lambda_i, \forall j \in J$

$\omega_j \leftarrow r_j$

until ($Gm! = \text{MaxGeneration}$){
    \text{localsearch}(s_{new})
    
    $s_{local} \leftarrow \min\left(\text{sort}\left((p_i^n)_{obj}\right)\right)$

    \text{If} ($s_{local} < s_{new}$) \text{ then}
    
    \hspace{1cm} $s_{local} \leftarrow s_{new}$

    \text{elseif (shuffletolerance > ShuffleToleranceLimit) {}
    \hspace{1cm} \text{shuffletolerance} ++
    \hspace{1cm} \text{exchg}(p^n_x, p^n_y)
    \}

    \text{If} ($s_{new} < s_{best}$) \text{ then}
    
    \hspace{1cm} $s_{best} \leftarrow s_{new}$

}

Declare $s_{best}$

\textbf{Figure 4.1:} Global Search in EEHA
The parameter shuffle tolerance tests the limit of tolerance when the local search struck at local optima. In generic heuristic algorithm, while performing search operation for feasible solution, often they are struck to the local optimum solution due to randomization for certain number of iterations. In order to push the algorithm to perform search operation beyond local optima, we need to perturb the current local assignment. This perturbation is initiated by verifying a global constant variable called shuffle tolerance limit.

For certain number of iterations, whenever the local search algorithm does not produce any output that differs from known local best solution, then the local parameter shuffle tolerance is incremented by one. In exceeding the limits of shuffle tolerance with shuffle tolerance limit, an unguided random shuffling of population takes place.

**Local Search Phase**

Figure 4.2 depicts pseudo code for local search algorithm. The local search phase performs multipoint exchange by choosing randomly x and y members of populations. Next, the prospective neighborhood solutions is obtained by permuting the members of population from the current local solution randomly. Base on this, for each RNC, the residual capacity (R_j) is updated. Here, if any of the NodeBs capacity exceeds the residual RNCs capacity, then redistribution move is performed. Next, the objective function of the neighborhood solutions is calculated. Reassignment function is to estimate the neighborhood solution.
localsearch(S_{new}){

exchg(p^n_x, p^n_y)

if(λ_i > ω_j) then redistrib(S_{new})

(P^n_i)_{obj} ← Σ_{i=1}^l Σ_{j=1}^l α_i^j β_i^j

R_j ← ω_j - Σ_{i=1}^l λ_i, ∀ j ∈ J

ω_j ← R_j

S_{local} ← min(sort((P^n_i)_{obj}))

If(S_{local} < S_{new}) then

S_{local} ← S_{new}

}

Figure 4.2: Local search in EEHA
Reassignment procedure

Reassignment procedure is performed by permuting cells of the current local solution, so that it leads to neighborhood solutions. Here, the search space is constrained by single homing of cells and the call handling capacity of switch. Consequently, intermediate solutions obtained are either feasible or infeasible. The main aim of this procedure is to explore various neighborhood solutions of the current local optimal solution, as there are chances, that these neighborhood solutions may have global best solution within it.

Sometimes, the local search is stuck in the local optimum solution. Even though best solution exists for the problem, there is no further improvement in the local optimum solution obtained. Reassignment is a method to overcome possibility of being stuck in the local optimum and continue search for the global best solution.

In Figure 4.3, NodeB \(x_i^{(m')}\) is currently connected to RNC (m’). Now, based on reassignment procedure, NodeB \(x_i^{(m')}\) is reassigned to RNC (m). The parameter delta \(\Delta_i^{(m' \rightarrow m)}\) represents differences between the cost of assigning NodeB i to switch m’ and the cost of reassigning NodeB i to switch m. As a result, if the current local solution exhibits lesser cost than global solution then update the reassignment. Otherwise, if no improvement is observed, then repeat the entire local search.
\( \phi_i^m \leftarrow \text{Current Best Solution} \)

\( C_i^m \leftarrow \text{Distance between } i^{th} \text{ NodeB and } m^{th} \text{ RNC} \)

\( X_i^m \leftarrow \text{Current Assignment of } i^{th} \text{ NodeB and } m^{th} \text{ RNC} \)

*Choose randomly and permute the contents of } i^{th} \text{ NodeB and } j^{th} \text{ NodeB}*

\[ \text{if} \left( x_i^m = 0 \text{ and } x_j^m = 0 \right) \text{then} \]

\[ \begin{cases} 
  x_i^{m'} & \leftarrow x_j^m \\
  x_j^{m'} & \leftarrow x_i^m \\
  \phi_i^{m'} & \leftarrow \sum_{m=1}^{M} \sum_{i=1}^{I} X_i^{m'} C_i^{m'} 
\end{cases} \]

\[ \text{if} \left( \Delta_i^{m' \leftarrow m} = \phi_i^{m'} - \phi_i^m \leq 0 \right) \text{then} \]

\[ \begin{cases} 
  \phi_i^m & \leftarrow \phi_i^{m'} \\
\end{cases} \]

*else*

\[ \phi_i^m \leftarrow \text{feasible solution} \]

*Figure 4.3:* Pseudo code for Reassignment procedure
**Neighborhood Redistribution Move**

The local search procedure performs neighborhood redistribution moves when there exist any infeasible solution. Basically, the infeasible solution occurs during global search because of constraint problem. However, by effective handling of overlapped infeasible neighborhood solution, will lead to better solutions. For this purpose, initially, RNC \( j \) that has minimal residual capacity is identified by using equation 4.15 given below

\[
R_j \leftarrow \omega_j - \sum_{i=1}^{\lambda_j} \lambda_i, \forall \ j \in J
\]

(4.15)

A local move is performed by reassigning NodeB \( i \) to new RNC \( j \), which has higher call handling capacity \( \lambda_i^{(j)} \). This reassign of NodeB \( i \) to switch RNC \( j \) ensures adequate residual capacity is existing at RNC\( j \). The residual capacity of each switch is updated after redistribution. The purpose of neighborhood redistribution method is to balance call-handling capacity of switches and to maximize the number of cells getting connected to switches.

Primary move in redistribution procedure is that, the call handling capacity of switch with minimum residual capacity is retained, by reassigning larger call capacity cell connected to it. This technique allows many possible moves that were infeasible earlier due to switch capacity constraint. The new assignment with lowest impact on the cost function is chosen for further iterations.

**4.2.3 RESULTS AND COMPARATIVE ANALYSIS OF EEHA**

The performance analysis of EEHA algorithm is conducted for both small-scale and large scale networks. The parameters like number of users on the network, the number of cells and the number of switches are used to create different network instances. The distance matrix and flow matrix are given as input to the algorithm. The cable cost of link between cells and switches are considered as proportional to the geometric distance between cells and switches. In binary flow matrix, the value one indicates when there exists a link between cell and switch. The value zero indicates non-existence of link between cell and switch. The capacity of the nodes is considered
randomly distributed. The experiment has been performed on Genuine Intel® CPU T2250 @ 1.73GHz, 1.00 GB of RAM machine. C++ language is used to implement the extended evolutionary heuristic algorithm.

Table 4.2: Network instances and its search space

<table>
<thead>
<tr>
<th>Network Instances Id</th>
<th>Network Instances represents as (NodeB, RNC, MSC)</th>
<th>NodeB to RNC Search Space</th>
<th>RNC to MSC Search Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(10,2,1)</td>
<td>$2^{10}$</td>
<td>$1^2$</td>
</tr>
<tr>
<td>2</td>
<td>(10,3,1)</td>
<td>$3^{10}$</td>
<td>$1^3$</td>
</tr>
<tr>
<td>3</td>
<td>(15,5,2)</td>
<td>$5^{15}$</td>
<td>$2^5$</td>
</tr>
<tr>
<td>4</td>
<td>(15,6,2)</td>
<td>$6^{15}$</td>
<td>$2^6$</td>
</tr>
<tr>
<td>5</td>
<td>(30,9,3)</td>
<td>$9^{30}$</td>
<td>$3^9$</td>
</tr>
<tr>
<td>6</td>
<td>(30,10,3)</td>
<td>$10^{30}$</td>
<td>$3^{10}$</td>
</tr>
<tr>
<td>7</td>
<td>(60,10,2)</td>
<td>$10^{60}$</td>
<td>$2^{10}$</td>
</tr>
<tr>
<td>8</td>
<td>(70,12,2)</td>
<td>$12^{70}$</td>
<td>$2^{12}$</td>
</tr>
<tr>
<td>9</td>
<td>(80,14,2)</td>
<td>$14^{80}$</td>
<td>$2^{14}$</td>
</tr>
<tr>
<td>10</td>
<td>(90,16,3)</td>
<td>$16^{90}$</td>
<td>$3^{16}$</td>
</tr>
<tr>
<td>11</td>
<td>(100,18,3)</td>
<td>$18^{100}$</td>
<td>$3^{18}$</td>
</tr>
<tr>
<td>12</td>
<td>(120,20,3)</td>
<td>$20^{120}$</td>
<td>$3^{20}$</td>
</tr>
</tbody>
</table>
Table 4.2 depicts various test cases used for analysis. Here, the number of NodeBs is varied between 10 and 120, the number of RNCs is considered between 2 and 20, and the number of MNCs is varied between 1 and 3. The search space size for mapping NodeBs to RNCs is varied between $10^{24}$ and $20^{120}$. Each instance is executed for 1000 iterations. The search space is varied between 1 and $3^{20}$ in order for mapping RNCs with MNCs. The following observations are made out this experiments.

- **Larger the network, the objective value is larger. Smaller the network the objective value is smaller:**
  
  In Figure 4.4, the network instance of 10 NodeBs, 2 RNCs and 1 MSCs have objective value of less than 10, where as the network instance of 120 NodeBs, 2 RNCs and 3 MSCs have objective value above 8000. It is observed that, in case of larger network, where the number of NodeB, RNC and MSC are more, the cost of these individual components add to the overall cost. This impacts the objective value to grow higher.

![Figure 4.4: Obtained objective value for various network instances](image-url)
• **The objective value decreases with increase in number of access network components:**

In Figure 4.5, the objective value of the network instance of \((15, 5, 2)\) has obtained value as the 254 where as the obtained value of the network \((15, 6, 2)\) is decreased to 221. The decrease in the obtained value is because of increase of RNCs.

It is to be noted that, the number of NodeBs is kept same, but number of RNCs is varied by increasing it from 5 to 6. This change impacts the obtained objective value to decrease. As, when the number of RNCs is increased, indicates more NodeBs need to be served by each of RNCs. Thus, the overall objective cost is decreased with increase in access network components.

![Obtained objective value after increasing network components](image)

**Figure 4.5:** Obtained objective value after increasing network components
The initial objective value is the highest of all the objective values obtained:

For example, the initial objective value of network instance 70x12x1 is 4124 where as the final obtained objective value is 16084. It is observed that initial objective value is high because of the random assignment during initialization phase. Now that, the proposed EEHA algorithm tries to improve the initial random assignment by using local search, reassignment and redistribution techniques.

Table 4.3, depicts the GAP estimation as percentage difference of initial value and obtained value of objective function. Approximately 63% of average improvement is achieved from initial objective value to feasible solution.
Table 4.3: Estimation of GAP between initial value and obtained value for various network instances

<table>
<thead>
<tr>
<th>Network Instances</th>
<th>Initial Value</th>
<th>Obtained Value</th>
<th>GAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10,2,1)</td>
<td>13</td>
<td>9</td>
<td>0.31</td>
</tr>
<tr>
<td>(10,3,1)</td>
<td>13</td>
<td>9</td>
<td>0.31</td>
</tr>
<tr>
<td>(15,5,2)</td>
<td>669</td>
<td>282</td>
<td>0.58</td>
</tr>
<tr>
<td>(15,6,2)</td>
<td>700</td>
<td>245</td>
<td>0.58</td>
</tr>
<tr>
<td>(30,9,3)</td>
<td>923</td>
<td>475</td>
<td>0.49</td>
</tr>
<tr>
<td>(30,10,3)</td>
<td>1000</td>
<td>450</td>
<td>0.47</td>
</tr>
<tr>
<td>(60,10,2)</td>
<td>13300</td>
<td>3683</td>
<td>0.72</td>
</tr>
<tr>
<td>(70,12,2)</td>
<td>16084</td>
<td>4121</td>
<td>0.74</td>
</tr>
<tr>
<td>(80,14,2)</td>
<td>19077</td>
<td>6065</td>
<td>0.68</td>
</tr>
<tr>
<td>(90,16,3)</td>
<td>21889</td>
<td>6657</td>
<td>0.70</td>
</tr>
<tr>
<td>(100,18,3)</td>
<td>24862</td>
<td>6994.57</td>
<td>0.72</td>
</tr>
<tr>
<td>(120,20,3)</td>
<td>29781</td>
<td>8012</td>
<td>0.73</td>
</tr>
</tbody>
</table>

Average 0.63
In larger networks, more is required to obtain the feasible solution. The smaller network consumes lesser time.

As given in Figure 4.6, for 500 iterations, the smaller network instance #1 (10, 2, 1) has computation time of 0.062 sec, whereas the computation time for network instance of (120, 20, 3) is 84.83 sec. The reason being that the search space is smaller in such network, for example in network instance (10, 2, 1) the search for heuristic algorithm is $10^2$, whereas the search space for the larger network (120, 20, 3), the search space is $(20^{120})^3$.

![Figure 4.6: Computation time for various network instances](image-url)
Increase in number of iterations, increases the time for computation of objective value. The lesser the number of iterations, the convergence time decreases. For more number of iterations, the convergence time is larger. This is depicted in Figure 4.7.

![Figure 4.7: Computation time for smaller networks](image)

In Figure 4.7, the computation time is recorded for various iterations viz 500, 2000, 2500, 5000, 10000 and 15000. For the network instance of (10,2,1) the convergence time is less than 2 second of computation time, where as in the larger network instances like (60,10,2) the computation time ranges below 140 sec.
Table 4.4 depicts the comparison report of the proposed EEHA algorithm and the greedy algorithm for various network instances. It is observed that, for the proposed evolutionary heuristic algorithm, the maximum gap between best and mean of obtained objective values is achieved as 0.004. Whereas, in case of greedy algorithm, the maximum gap between best value and the mean of obtained objective value is 0.028.

Further, the average time taken by the proposed EEHA algorithm is 36.6 sec whereas the greedy algorithm consumes 75.5 sec. The average time gap between EEHA algorithm and greedy algorithm is 3.045. It is clear that, the proposed algorithm outperforms even in case of larger network.
Table 4.4: Comparison result between proposed heuristic algorithm and greedy algorithm

<table>
<thead>
<tr>
<th>Network instance ID</th>
<th>Proposed Heuristic Algorithm</th>
<th>Greedy Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best</td>
<td>Average</td>
</tr>
<tr>
<td>(10,2,1)</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>(10,3,1)</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>(15,5,2)</td>
<td>282</td>
<td>283</td>
</tr>
<tr>
<td>(15,6,2)</td>
<td>245</td>
<td>243</td>
</tr>
<tr>
<td>(30,9,3)</td>
<td>475</td>
<td>476.5</td>
</tr>
<tr>
<td>(30,10,3)</td>
<td>450</td>
<td>451.5</td>
</tr>
<tr>
<td>(60,10,2)</td>
<td>3683</td>
<td>3684</td>
</tr>
<tr>
<td>(70,12,2)</td>
<td>4121</td>
<td>4121</td>
</tr>
<tr>
<td>(80,14,2)</td>
<td>6065</td>
<td>6068</td>
</tr>
<tr>
<td>(90,16,3)</td>
<td>6657</td>
<td>6660</td>
</tr>
<tr>
<td>(100,18,3)</td>
<td>6994.57</td>
<td>6997</td>
</tr>
<tr>
<td>(120,20,3)</td>
<td>8012</td>
<td>8015</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.2.4 OUTPUT VISUALIZATION OF NETWORK

Figure 4.9, represents the sample output for a network instance with 30 NodeB, 8 RNC and 3 MSC, for the proposed evolutionary heuristic algorithm. The first round of algorithm is applied on the NodeB and RNC search space of size $10^{30}$ for the objective function of cost optimization problem given in equations (4.1 - 4.3). Then the same algorithm is repeated for the RNC and MSC level based on the objective function specified in equation (4.4 – 4.7).

(a) Network instance (30, 8, 3) - Initial node distribution
(b) Network instance (30, 8, 3) - NodeB – RNC Assignment
Figure 4.9: MATLAB Visualization of Optimized Mobile Access network
Figure 4.10 represents the sample output for a network instance with 90 NodeB, 10 RNC and 1 MSC. Then, the proposed evolutionary heuristic algorithm is executed for this instance. The first round of algorithm is applied on RNC level with NodeBs and RNCs. Here, the search space of $10^{90}$ is analyzed for the objective function expression given in equations (15–20). Then the EEHA algorithm is repeated for MSC level with RNCs and MSCs for the objective function.

(a) Network instance (90, 10, 1) - Initial node distribution
(b) Network instance (90, 10, 1) - NodeB – RNC Assignment

(c) Network instance (90, 10, 1) - RNC – MNC Assignment

* represents NodeB; • represents RNC; ♦ represents MNC

Figure 4.10: MATLAB Visualization of Optimized Mobile Access network
4.3 ISSUES IN COST OPTIMIZATION OF HIERARCHICAL MOBILE ACCESS NETWORK

This chapter discussed about the necessity of and efficient techniques for performance enhancement, to be incorporated in cost optimization problem in hierarchical mobile access network. However, there are certain issues and shortcomings with hierarchical mobile access network. This section presents some of these issues. First, this section discusses, various issues in hierarchical mobile access network, and then later discusses various issues in cost optimization problem of hierarchical mobile access network.

Issues in hierarchical mobile access network

- **Strictly hierarchical network:** The first major issue in mobile access network is strictly hierarchical topology. For better understanding on this issue, the hierarchical topology is viewed as a graph [21, 41]. In this graph, the nodes are various network components like NodeB, RNC, and MSC. The edges of graph are the transmission links connecting these network components in hierarchical manner. Thus, tree like structure is exhibited in hierarchical connectivity among nodes.

Hierarchical topology is highly vulnerable to failures, due to weak network topology and weak connectivity as a single link/component failure renders network partly out of service. Further, from mobile service providers perspective, even single link or node failure in this hierarchical topology, could cost heavy damage to the network services. Hence, an effective fault tolerance mechanism has to be provided to handle intolerance to loss of communication. In this work, one such fault tolerance mechanism called dual homing has been proposed and has been discussed in next chapter.
• **Necessity to identify various types of failures:** Related works on fault tolerance show intensive studies that have been carried out, by considering mobile network as a complete network design. [25, 27] address various types of failure in entire mobile network. The impact of failure on the network operation and cost incurred due to failure have also been addressed.

However, as in [8, 10], it is very clear that, mobile access network plays vital role in functioning of entire mobile network. Further, not much literature is available specifically on failure in hierarchical mobile access network. Hence, various types of failures impacting to hierarchical mobile access network have to be worked out. To address this issue, this work identifies various types of failure specific to hierarchical mobile access network in the next chapter.

• **Single homed:** The next issue in hierarchical mobile access network is, single homing of nodes. In single homing, each node connects to exactly one node at next higher level [37, 38]. With this constraint, even single link failure could cost heavy damage to the network services. Especially loss of critical communication cannot be tolerated and need an effective restoration mechanism. Hence, this work proposes dual homing of nodes as an effective fault tolerance mechanism and the details are given in the next chapter.

**Issues in cost optimization of hierarchical mobile access network**

• **Needs further performance enhancement along with cost optimization:** As in [54-56], the cost optimization is the most important and complex problem in hierarchical mobile access network designing. However, the hierarchical nature of mobile access network suffers from issues like weak topology, single homing of network component and network failures. Hence, cost optimization alone cannot produce the best performance enhanced network design. As a result, along with cost optimization, an extended approach towards performance enhancement needs to be incorporated. In this regard, this work emphasizes fault tolerant mechanism as another major performance enhancement in hierarchical mobile access network. The next chapter addresses the fault tolerance along with cost optimization in hierarchical mobile access network.
• **Redesigning solution approach to accommodate cost optimization along with fault tolerance in mobile access network:** This chapter presents mathematical model and EEH algorithm as solution approach for cost optimization in mobile access network. However, from above discussion, identifies fault tolerance as another key factor to enhance performance in mobile access network is identified. Hence, it requires redesigning of solution approach in order to accommodate fault tolerance along with cost optimization in mobile access network. In this aspect, the redesigning is carried out, first in terms of mathematical model and then, second in terms of heuristic algorithm. Thus, the next chapter addresses cost optimization along with fault tolerance in mobile access network.

### 4.4 CONCLUDING REMARKS

This chapter discusses non-effectiveness in cost optimization approach of non-hierarchical access network. For this purpose, this chapter examines hierarchical nature of mobile access network and cost optimization techniques in hierarchical mobile access network. Further discusses mathematical model for hierarchical mobile access network at various levels viz NodeB level and RNC level was also discussed.

In order to find solution for cost optimization problem in hierarchical network, this chapter proposes Extended Evolutionary Heuristic algorithm (EEHA). EEHA utilizes efficient techniques like local search, redistribution, and shuffle tolerance. The analysis of EEHA algorithm outperforms in aspect of computation time and obtained feasible solution. Further, EEHA exhibits excellent gap estimation obtained by relative difference between initial value and obtained value of objective cost function.

Average percentage of improvement from initial objective value to that in final solution is approximately 63%. Further, the comparative analysis between EEHA and greedy algorithm shows good performance of the former. The maximum gap between best and mean value of the obtained objective function for the EEHA is 0.004, and is <1%. For greedy algorithm, the maximum gap between best value obtained and the mean is 0.028 (~3%).
In addition, EEHA computation time is approximately 48% less than conventional approach. The average time taken by the EEHA is 36.6 sec whereas the computation time taken by greedy algorithm is 75.5 sec. In case of larger network, the proposed EEHA algorithm clearly outperforms as the average time gap between proposed algorithm and greedy algorithm is 3.045. The experimental results have shown computation time for both smaller and larger network are approximately same.