Planning Reliable UMTS Access Networks

ATTILA SZLOVENCSÁK, JÁNOS SZIGETI
Traffic Analysis and Network Performance Laboratory,
Ericsson Research,
POB 107, H-1300 Budapest,
HUNGARY
{Attila.Szlovencsak,Janos.Szigeti}@eth.ericsson.se

Abstact: - The access network of the Universal Mobile Telecommunication System (UMTS) forms a tree–topology network, however, the great amount of traffic carried in it requires a more reliable network structure. That way, new methods are needed to plan a minimum cost extension of the classical topology. In order to satify the reliability expectation, expressed as a limitation for the traffic loss in the network, we insert additional links, while taking into account several topological constaints. In the current paper, we introduce a genetic algorithm to solve the specific problem, and we compare the results of it with the ones produced by a greedy heuristic.

Key-Words: - Network Optimization, Reliability Analysis, Access Networks, 3G Mobile, Genetic Algorithms

1 Introduction

Today, increasing amount of the voice communication use wireless networks. Current tendencies show that beside voice transmission new types of services will appear, demanding high rate of data transmission capability from the next generation of wireless networks.

The Universal Mobile Telecommunication System (UMTS) [1], is the European representative of third generation networks. In UMTS, the network is built from two main type of components: the radio base station (RBS) and radio network controller (RNC), as shown on Fig. 1. The RBSs have the task to handle their own radio channels, while the task of the RNC is to manage the radio channels of the connected RBS's, and to concentrate/relay their traffic to the upper level core network. The access network forms tree–topology, where every RBS aggreagates and forwards the traffic of the corresponding lower level RBSs. Besides, in earlier releases of UMTS, the RBS does not have routing capability, therefore even the traffic between neighbouring RBSs need to pass their dedicated RNC.

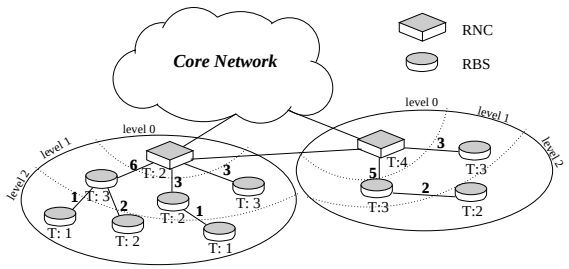

Figure 1: The structure of UMTS

The optimal construction of the access part of these networks is an important task in the current UMTS-related activities. If the correct number and location of RBSs is given, the planning process consists of two basic parts, namely:

a) *Clustering*: Find the optimal number and location of RNCs, and define sets of RBSs to be connected to them.

b) *Tree construction*: Create a cost-optimal RNC-rooted tree topology subnetwork from a given set of RBSs.

In the literature there are some papers dealing with the planning of multi–constrained spanning tree topologies. Paper [3] deals both with the clustering and the tree construction, while [4] focuses only on the construction of the trees. In the present paper we also focus only on the planning of an RNC-subnetwork.

In a high capacity data network, the reliability is also an important thing to deal with. In case of the access network of UMTS, the basic tree topology and the usage of microwave links lead to a high sensitivity to any kind of failure. Since a single failure may result a significant loss of data, a fault-tolerant topology is required. As the number of stations to be interconnected is large, both the traditional mesh–like and ring–topologies can be very uneconomical, so a new method is required to plan a topology with acceptable cost that satisfies given reliability demands.

Several papers are dealing with reliability. Most papers [5, 6, 7] propose heuristic (e.g. genetic) algorithms to minimize the cost of mesh networks while

taking into account the *all terminal reliability* constraint, which defines the probability that every pair of network nodes can reach each other. These methods provide good solutions of the problem above, but they are infeasible in case of such large networks, as the access part of UMTS. In order to handle networks up to some hundred devices, in [2] we presented a method for planning tree-topology networks and an algorithm for extending these networks with extra links to gain higher reliability. The latter algorithm occured to have some weaknesses, so in the present study we construct a more effective one, which solves the network extension task using a genetic approach.

The remainder for the paper is organized as follows. First we present the applied network and reliability models, and give the mathematical formulation of the planning problem. Second, we describe the specific operators, heuristics used in the genetic algorithm, then give a detailed numerical study of the algorithm by comparing it to the solution given in [2]. Finally, we conclude the paper with a brief summary.

2 Problem Statement

As mentioned, our aim is to enhance the reliability of an RNC-rooted subnetwork, in a cost optimal way. During the extension of the network with extra links, we have to consider topology constraints and traffic demands, besides the reliability expectations.

2.1 Network model

Let the communication network be modeled as a directed graph $G(\mathbb{N}, \mathbb{E})$, where $\mathbb{N}$ denotes the set of network nodes including all the RBSs and the RNC, and where $\mathbb{E}$ ($\mathbb{E} \subseteq \mathbb{N} \times \mathbb{N}$) stands for the set of communication links. Our input is a single–connected tree topology network, including parameters T_i, the traffic demand between RBS $i \in \mathbb{N}$ and the RNC, and $C_{i,j}$, the capacity of the link $(i,j) \in \mathbb{E}$. Initially, $C_{i,j}$ equals to T_{aggr_i}, where T_{aggr_i} is the traffic aggregated by RBS i. During the optimization the initial set of links, and the traffic demands are considered to be fixed.

The input also includes black-box cost functions for the RNC, RBSs and links, denoted by $\mathbb{C}_{RNC}$, $\mathbb{C}_{RBS}(k)$ and $\mathbb{C}_{link}(i,j)$, respectively. These functions can reflect several features of the given components (including installation cost, upkeep, an so on).

The link cost $\mathbb{C}_{link}(i,j)$ between sites i and j is typically represented by an increasing step–wise or piece–wise function, and capable of modeling both wired (leased-line, fiber, coax) and wireless (microwave) interconnections.

The node cost functions of $\mathbb{C}_{RNC}$, $\mathbb{C}_{RBS}(k)$ are composed of several sub-costs regarding the capacity and the number of ports, the number of processors, and so on.

In addition, from the viewpoint of the planning process, we have two important technological constraints to be considered:

- *Cascading constraint (L)*: The parameter l_i (level), defines the length of the shortest path between the RNC and RBS i, and constraint L is the upper bound for this parameter. This constraint limits the maximum delay of communication in the network, which is essential for real time applications.

- *Degree constraint (D)*: The degree of an RBS (d_i) is the number of links per RBS, including both incoming and outgoing ones, and D is an upper bound of this value. We note, that this is a relatively weak limit factor, however, it can have a significant impact on reliability, if a strong reliability expectation could be fulfilled only with a dense network topology.

2.2 Reliability model

If we take the reliability into account during the network planning, the complexity of the planning problem can highly depend on the reliability model we choose. If we assume multiple failures and mesh topologies, the exact calculation of network reliability can be NP-hard [10]. A good estimation of the *all-terminal reliability* metric can be given using Monte Carlo simulation, but even this one can be an expensive procedure if we use iterative methods, and this calculation must be performed in each iteration. Fortunately, in our case an easier calculation can be made, because of the centralized structure of the network.

Originally, there is exactly one path between each RBS and the RNC, containing several RBSs and links. The availability function of these components is also given as input, so the availability of the whole path can be computed as given in (1),

$$A(P) = \prod_{e \in P} A(e) \tag{1}$$

where $A(e)$ is the availability of the e element in the path.

If we extend the network with additional links, then new paths are assigned for the RBSs beside the original one. In that case, the availability of RBS i is calculated by

$$\begin{aligned} A_i = \; & \textstyle\sum_{P_a \in \mathbb{P}_i} A(P_a) - \\ & \textstyle\sum_{P_a, P_b \in \mathbb{P}_i} A(P_a \cup P_b) + \\ & \textstyle\sum_{P_a, P_b, P_c \in \mathbb{P}_i} A(P_a \cup P_b \cup P_c) - \\ & \ldots, \end{aligned} \tag{2}$$

where $\mathbb{P}_i = \{P_a, P_b ... P_z\}$ is the set of paths assigned to RBS i, and $P_a \cup P_b \cup ... \cup P_z$ denotes the union of elements in these paths.

In mesh networks usually two or more node-disjoint paths are assigned for each traffic demand. In the current case, this protection scheme is not applicable due to the large number of sites, so we protect the most failure sensitive parts of the network only, as presented in Fig. 2

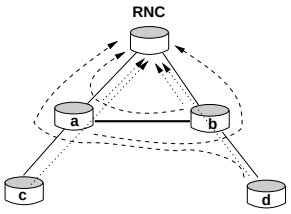

Figure 2: Protection scheme

As we can see, only RBSs a and b has a full protection path, c is not protected, and d has a backup path containing one edge common with the original path. The additional link (a, b) is shared between two paths, as there is no error case, when both backup paths need to be used.

Using this protection scheme, it is important to find the proper place for a backup link. The availability of RBSs is not a perfect metric for this, since two RBSs with the same availability can have different importance, due to the different amount of traffic aggregated in them. That way, we define parameters $\mathbb{L}_i = (1 - A_i) \cdot T_i$, and $\mathbb{L}_{aggr_i} = (1 - A_i) \cdot T_{aggr}$ for each RBS. The first parameter determines the loss of the originated traffic of RBS i, and we use the average of this variable ($\mathbb{L}_{avg}$) as a metric for the overall network reliability. The second parameter defines the loss for the aggregated traffic in RBS i, an we use this value to select the appropriate RBS to be protected.

2.3 Optimization task

Our goal is to extend the input network with additional links until we satisfy the constraint to the average traffic loss, denoted by $\mathbb{L}_{lim}$. The optimization problem is to minimize the total cost of the network $\mathbb{C}$, where

$$\begin{aligned} \mathbb{C} &= \mathbb{C}_{RNC} + \textstyle\sum_{k\in\mathbb{N}, k\neq RNC} \mathbb{C}_{RBS_k} \\ &+ \textstyle\sum_{(i,j)\in\mathbb{E}} \mathbb{C}_{link}(i,j) \end{aligned} \quad (3)$$

subject to the constraint $\mathbb{L}_{avg} \leq \mathbb{L}_{lim}$.

Besides, the provided solution also has to satify the following constraints:

- $l_i \leq L, d_i \leq D$ where $i \in \mathbb{N}$
- $l_j - l_i \in \{0, 1\}$, where $(i, j) \in \mathbb{E}$ is a backup link. It means we can create backup links between RBSs on the same or neighbouring levels.
- the maximum number of extra links per RBS is 1
- every protection path must contain exactly 1 backup link. In other words, protection paths can be at maximum one longer, than the appropriate working ones.

3 Genetic Algorithm

The genetic algorithms are inspired by the well-known genetic process, where populations of species slowly evolve and adapt to the environment. The slow evolution of populations is advanced by three main procedures, namely:

- *Mutation*, which provides the diversity of the population by making random changes in the genes of individuals.
- *Crossover*, which supplies the spread over of the advantageous features as the parents may mix their successful genes in their offspring.
- *Selection*, which lets only the best individuals survive the race for the resources of the environment.

Genetic algorithms are often used in case of optimizations where analytical methods cannot be used. In our case it seems to be a feasible, however, we also know that a classical GA, using classical binary or real operators can hardly give a solution that comparable to the one provided in [2]. So we had to adapt our GA operators to the specific problem.

3.1 Encoding

The first decision to make, is how to represent the networks as a genetic instances. As we mentioned, some input parameters (fixed links, traffic demands, etc.) are considered to be constant, and these are the same in all instances. In that way, two type of vector describe an extended network:

- The vector of extra (backup) links $\overline{x}$, where $x_i = j$, if there is an existing backup link from RBS i to RBS j.
- The vector of protection paths $\overline{y}$, storing a list of assigned protection paths for each RBS. As the maximum length and number of protection paths per RBS is limited by constraints given in 2.3, $\overline{y}$ can be also a fixed size vector.

During the genetic optimization, each instance is represented by a $(\overline{x}, \overline{y})$ pair. The standard operators manipulate directly $\overline{x}$, but also maintain $\overline{y}$ according to the changes.

3.2 Objective function

Our task is a *multiobjective optimization with constraint handling* [8]. The genetic operators we use, do not produce individuals, which violate the constraints, except for $\mathbb{L}_{avg} \leq \mathbb{L}_{lim}$. The traffic loss is regarded as the part of our objective, so we have a two-dimensional objective function $\overline{f}(\overline{x}, \overline{y})$, where $f_1(\overline{x}, \overline{y}) = \mathbb{C}$ and $f_2(\overline{x}, \overline{y}) = \mathbb{L}_{avg}$.

In order to compare two instances in the two dimensional product space of $\overline{f}(\overline{x}, \overline{y})$, we use the *preference relation*. In case of comparing instances U and V having objective functions $\overline{f}^U = \overline{f}(\overline{x}^U, \overline{y}^U)$ and $\overline{f}^V = \overline{f}(\overline{x}^V, \overline{y}^V)$, we prefer U to V,

- If $f_2^U, f_2^V \leq \mathbb{L}_{lim}$ and $f_1^U < f_1^V$, or
- If $\mathbb{L}_{lim} < f_2^V$ and $f_2^U < f_2^V$

3.3 Mutation

We have the following demands on the mutation operator:

- give diversity to the population
- create new backup links
- operate directly at the most failure sensitive parts of the network

For realization of mutation we assembled two procedures:

M_1 Insert an outgoing backup link to an unprotected RBS .

M_2 Modify the endpoint of an existing backup link. This operation creates a new link in the same way as M_1, but also deletes an existing backup link.

The procedure of the mutation is as follows:

1. Select an individual S of the population in a random way.
2. Create instance U as a replica of S.
3. Select RBS i. In case of M_1, i is an RBSs with high $\mathbb{L}_{aggr}$ value, and with $x_i^U = 0$ representing that RBS i does not have any outgoing backup link. In case of M_2, i is randomly selected from RBSs with $x_i^U \neq 0$.
4. Let $\overline{n}_i^U$ be a list of the neighboring RBSs of RBS i, representing a possible set of endpoints for backup links.
5. Choose an RBS j from list $\overline{n}_i^U$ by probabilistic method, ensuring that a distant RBS has the smallest chance to be selected, and change x_i^U to j.

3.4 Crossover

The operator of crossover aims to fuse the high reliability parts of two parent networks. It constructs two offspring, one with a higher and one with a lower number of backup edges, and higher/lower reliability, respectively. Therefore, using crossover we can purify an over-secured network. As another effect we also expect that the genes responsible for good properties should spread over the population by crossing an average and a better instance.

The realization of the operator is as follows:

1. Select two parents (R, S) from the population randomly.
2. Create two offsprings: U and V.
3. For genes x_i^U,x_i^V we decide whether to inherited them from parent R or S:
   - Let k and l denote the starting endpoints of the links represented by x_i^R and x_i^S, respectively.
   - IF $A_k * rnd() > A_l$,
     THEN $x_i^U := x_i^R$ and $x_i^V := x_i^S$,
     ELSE $x_i^U := x_i^S$ and $x_i^V := x_i^R$,
     where $rnd()$ returns a value from uniformly distributed range (0.75;1.25).

The use of the $rnd()$ multiplier needs an explanation. If we make the above comparison without this randomization, then the inheritance would be deterministic and we would get to two extreme offsprings, which could hardly compete with other instances. In order to eliminate this behavior, we use this randomization, where range of function $rnd()$ is defined empirically.

3.5 Selection

If we want to keep the number of instances under a limit, generation by generation, we have to drop some individuals of the population. At this point we must consider two things:

- The best instances should not die, and mainly the better instances should be kept.
- On the other hand, even a worse instance should have the possibility to survive.

Therefore we use the *k-Tournament*[9] (more exactly *2-Tournament*) method, where the 2 opponents are selected randomly and competed. The loser of the competition dies out, and after a number of tournaments, we reach the required size of population.

In this procedure, we use the previously introduced *preference relation*, so we do not have to rank all the items.

3.6 Enhancements

3.6.1 Disaster

While performing an evolution, after a couple of generations it can be realized that the population does not evolve any more or the evolution slows down. The population in that case can have very similar instances, and this low diversity is the reason to stuck. At this point we need a drastic intervention, called *disaster*, which is as follows : we decrease the size of the population down to 2 - by making 2-tournaments - and then apply several mutations on the remained networks and on their mutated instances, to reach the original size of the population. To create significantly different instances, we use the M_1 and M_2 procedures several times.

3.6.2 Path reduction

Using more backup paths offers a higher reliability, but sometimes backup paths make only a negligible reliability-enhancement. By dropping some inefficient backup paths we could decrease the capacity of links, therefore the network becomes less expensive. This is the reason to revise the backup paths of the network (i.e.. the $\overline{y}$ vector of the instance) and create the so-called *path reduction* procedure, which examines all RBSs in the selected R instance:

- Using y_i^R, collect all backup paths of RBS i into the list $\mathbb{P}_i^R$,
- For each P_j item of $\mathbb{P}_i^R$ calculate A_{ij}^R, denoting the availability of RBS i when P_j is omitted from the paths of RBS i
- If $\max_j(A_{ij}^R) \geq H \cdot A_i$, remove P_j of the set of backup paths of RBS i and update y_i^R accordingly. In our case, $H = 0.99$ and chosen empirically.

3.7 Evolutionary process

Using the previously defined operators and extensions, the evolutionary process is as follows:

- First, we create the initial population by mutating the input individual as described in procedure *disaster*.
- In each evolutionary cycle (generation) we perform the followings:
  1. Increase the population size by mutations and crossovers.
  2. Drop some instances using selection.
  3. Evaluate the population: If there is no development since some generations, apply a *disaster* on the population and *path reduction* on all individuals. Then loop to the next cycle, except the case when the maximal number of generations is reached or when no evolution occurred since the last three *disasters*.
- As output we store the most preferable individual of the last population and this can be used as input for a next evolutionary algorithm.

4 Numerical Results

In this section we examine the behavior of the genetic algorithm, using several test cases. In [2], we defined two algorithms, the *Penalty Tree Algorithm (PTA)*, and the *Randomized Reliability Enhancement (RRE)*. The first algorithm is able to create a preoptimized tree topology RNC-subnetwork, while the second one performs an extension on these networks to gain a higher reliability output. During the tests, we use the PTA resulted networks as input. On the other hand, as our network, cost and reliability model is identical to the one presented in [2], we can use the RRE as a reference.

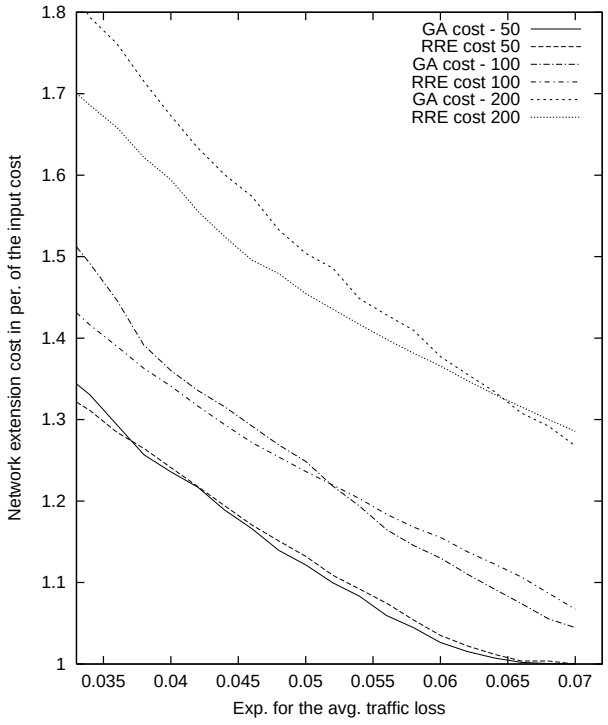

Figure 3: GA and RRE extension costs in case of different network sizes

In the first test, we compared the cost of the network extension resulted by the RRE and the evolutionary algorithm (GA). We used different network sizes, and different traffic loss expectations ($\mathbb{L}_{lim}$). Fig. 3 shows network extension costs in case of networks containing

50, 100 and 200 RBSs. The presented values are averages of ten runs. The costs are normalized to cost of the input networks, while the $\mathbb{L}_{lim}$ expectation values are expressed as a percentage of the average traffic of RBSs. (As Fig 3 shows, in the network containing 50 nodes, this initial average traffic loss is 0.067 %, so the additional cost is 0 until this value.)

As we can see, our GA gave better results in the cases, when a weaker traffic loss expectation ($\mathbb{L}_{lim}$) was used. However, increasing the size of input networks, the intersection of the curves represented by the two algorithms is place at higher $\mathbb{L}_{lim}$ values. The reason is the following: larger networks have larger initial traffic loss value, since the average length of paths between RBS and the RNC is bigger. Satisfying the same $\mathbb{L}_{lim}$ constraint, in case of larger networks, the GA requires more additional links, and have a larger search space, than in case of small ones.

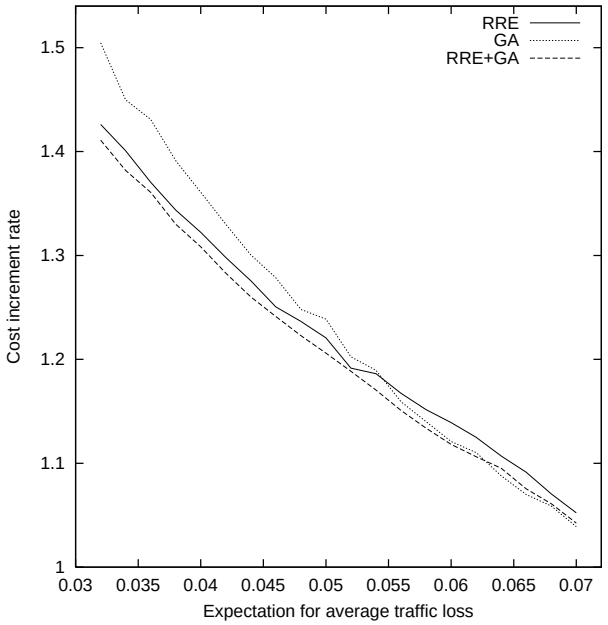

Figure 4: Joint use of GA and RRE

In the second test we examined the behavior of GA, in case of preoptimized input networks. On Fig. 4, we present extension costs of a network containing 100 RBS. Beside the cost of the GA and the RRE, a third curve shows the results, when we used the RRE resulted networks as an input for GA. As we can see, the our evolutionary algorithm can further optimize the outputs of RRE, giving a 1-2 % lower costs.

5 Conclusions

In this paper, we gave a solution for a network reliability enhancement problem, where the objective is to extend the a originally tree–topology network with extra links, to satisfy a constraint for the average traffic loss. A previously defined greedy heuristic called RRE can solve this problem, however, it is not effective in case of weaker traffic loss expectations. That way, we constructed an evolutionary algorithm with specialized operators and extrensions. During tests, we found that create algorithm met or expectations, as it gave better result than RRE in the cases mention above. In addition, we realized that our GA can make further optimization on RRE resulted networks, so the joint use of the two algorithms provides the most efficient solution.

References:

[1] Tero Ojanperä, Ramjee Prasad, *Wideband CDMA for Third Generation Mobile Communication*, 1998, Artech House Publishers

[2] Attila Szlovencsák, István Gódor, János Harmatos, Tibor Cinkler, *Planning Reliable UMTS Terrestrial Access Networks*, IEEE Communications Magazin, Vol 40, No 1, 2002, pp. 66-72

[3] Paul Kallenberg, *Optimization of the Fixed Part GSM Networks Using Simulated Annealing*, Networks 98, Sorrento, October 1998.

[4] János Harmatos, Áron Szentesi, István Gódor, *Planning of Tree-Topology UMTS Terrestrial Access Networks*, PIMRC 2000, London, September 2000.

[5] R. H. Jan, *Design of reliable networks*, Comput. Oper. Res., Vol 20, pp. 25-34, 1993

[6] M. M. Aliqullah, S. S. Rao, *Reliability optimization of communication networks using simulated annealing*, Microelectronics and Reliability, Vol 33, pp. 1303-1319, 1993

[7] Berna Dengiz, Fulya Altiparmak, Alice E. Smith, *Local Search Genetic Algorithm for Optimal Design of Reliable Networks*, IEEE Trans. on Evolutionary Comp., Vol 1, No. 3, 1997.

[8] C. M. da Fonseca, P. J. Flemming, *Multiobjective Optimalization and Multiple Constraint Handling with Evolutionary Algorithms I: A Unified Formulation*, IEEE Transaction on Systems, Man and Cybernetics - Part A: Systems and Humans, Vol. 28, pp.38-47, 1998

[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass., 1989

[10] M. Ball, R. M. Van Slyke., *Backtracking algorithms for network reliability analysis*, Ann. Discrete Math., Vol. 1, pp 49-66, 1977