Enhanced Survivable Topology Redesign of Optical Broadband Networks with Biconnectivity

Rong Zhao
Detecon International GmbH
53227 Bonn, Germany
rong.zhao@detecon.com

Christian Minge
Dimension Data Germany AG&Co.KG
61440 Oberursel, Germany
christian.minge@eu.didata.com

Mathias Schweigel
Detecon International GmbH
01187 Dresden, Germany
mathias.schweigel@detecon.com

Abstract

This paper presents an efficient strategy for the optimal network redesign with a biconnectivity-oriented topology (both edge and vertex biconnected). It helps re-designing existing networks or generating new networks considering practically relevant constraints (such as leased lines with long running contracts to remain, maximum number of ports per device) typically found in these phases. The proposed strategy is composed of Reduction, Augmentation, and Fine-Tuning Reduction. Empirical tests using several IP network topologies showed the robustness and applicability of the method. Its application to other optical backbone (or access) network redesign problems is possible.

Keywords - Optical Networks, Redesign, Broadband Networks, Survivability, Biconnectivity

1 Introduction

When designing/redesigning telecommunications networks, we have to face up to two opposing objectives: high survivability and low costs. The former leads to a fully meshed topology (expensive and highly redundant), whereas the latter results in a minimum spanning tree topology (cheap with no redundancy). The main idea of this approach is to redesign a network by retaining essential network links (reduce network migration costs), improving their utilization (but not to overload), and ensuring network survivability. If necessary, only a few links are allowed to be added into networks subject to minimum costs. Simultaneously, some constraints are considered, such as routing, maximal hop number, and node degree. A part of this work was presented at AICT2008 conference [1].

Some prior approaches addressed the topological design for backbone networks by [2][3][4][5][6] and access networks by [7][8][9][10][11]. Generally, network structures are illustrated by hierarchical star-star, tree-star, or mesh-star topology. Most of these optimization problems are NP-hard [12]. Due to the complexity of the tasks different methodologies were investigated, such as the linear programming, Simulated Annealing(SA), Tabu search(TS), Genetic Algorithms(GA). A detailed formulation for general network design problems with connectivity requirements was introduced in [13][14]. Furthermore, the network redesign is discussed in [15][16].

IP networks reportedly suffer from node failures as frequently as from link failures [17]. To avoid service degradation (and related penalties) the design of reliable communication networks is a significant problem for network providers. The general planning problem is finding the best positions of components and their links subject to minimal costs and a high reliability [18]. An essential summary of approaches to different reliability problems, such as constrained reliability measures and reliability optimization, is provided in [19]. A general definition of *reliability* of network components is the probability that the network is functioning [20]. In comparison with reliability, survivability is to describe the resilient ability of networks when one or more network components fails. More precisely, the survivability analysis is to make a conservative assumption of failures and study how to prevent them. Typical network survivability techniques are based on well designed networks and network restoration [21][22], e.g. link restoration or link/node protection.

A possible solution for the above mentioned design problem is a two-connected topology that can be described by means of graph theory. Graph connectivity properties are meaningful for transport network designs. In order to survive all single edge failures, a graph must be at least *two-edge connected*. Furthermore, it has been shown that every *two-vertex connected* graph is also two-edge connected, while the reverse is not valid [23]. In this work a redesigned network has to be at least *two-vertex connected*. In the following we characterize the *two-vertex connected* graph as *biconnected* graph.

The proposed strategy for network topology redesign to improve network reliability and to find out a cost-effective structure is composed of three parts: *Reduction*, *Augmentation* and *Fine-Tuning Reduction*. An approximation algorithm for the *Augmentation* problem was introduced in 1981 in [24]. In 1993 another heuristic for the same problem with better time complexity was proposed in [25]. Later, this approach has been improved [26] and it has been solved by applying a Genetic Algorithm [27]. In [28] Hackbarth et al. introduced a heuristic that covered the whole problem of telecommunications network design from a totally meshed network to a *two-edge connected* topology solving the *Reduction* and *Augmentation* problem. This contribution extended their approach:

- The problem is extended to a *two-vertex connected* topology;
- An additional step is proposed to remove redundant edges through (*Fine-Tuning Reduction*) for the final solution set;
- Time complexity and efficiency of the *Augmentation* are improved by applying a *modified Depth First Search* (DFS)[29] and classifying the candidate edges before *Augmentation*;
- Links are classified for efficient manipulation of the optimization considering practically relevant network optimization constraints.

The paper is organized as follows: firstly, it provides a mathematical formulation of the considered problem and a discussion of the related work. In terms of biconnectivity, several relevant concepts of graph theory are introduced and analyzed. Next, a detailed description of the algorithm is given. Furthermore, the efficiency of algorithm by means of the presentation of optimization results of *AT&T* backbone network and *G-WiN* network is shown. Finally, some parameters of the redesign algorithm are analyzed.

2 Problem Statement

2.1 Objective Function

The existing telecommunications network can be described as an undirected Graph $G(V, E, W)$ with node $v \in V$, edge $e \in E$ and weight $w \in W$. The weight *W* is a problem-specific parameter. In this approach the weight of edges is not the same for all calculation steps. For calculating network costs, it is based on bandwidth and length. But for other redesign steps, different functions are defined to represent the weight of an edge. This will be discussed in detail by the introduction of the redesign procedure.

The objective is to minimize total network costs by re-designing the network structure. Some links can be removed or added to the network considering the given constraints at all the time. All network nodes are fixed without any change and their costs are ignored in this work. The objective function is defined as:

$$C_{net} = \sum_{e \in E} x_e c_e \quad (1)$$

where e is an edge in *E*; x_e is a binary variable to check if edge *e* is accepted or not. c_e is the leasing cost of edge e, see Eq.2.

$$c_e = f(l_e, \mu_e) + C_{e,k} \quad (2)$$

where $f(l_e, \mu_e)$ is the leasing cost function of edge (optical cable) e depending on edge length l_e and bandwidth (i.e. capacity) μ_e. $C_{e,k}$ is the fixed cost for different edge (link) types *k* in terms of bandwidth μ_e. A practical example will be introduced in the section IV.

2.2 Constraints

A packet transmitted over an asynchronous, time–multiplexed, packet switched network (like IP networks) undergoes a queuing and serialization delay at the end/beginning of a link (router entrance/exit respectively) which depends on the link utilization [30][31]. Mean packet delay depends on the traffic rate λ_e and capacities of links μ_e, i.e. utilization $\rho_e = \lambda_e/\mu_e$. In this work the maximum link utilization is taken as the first constraint. Therefore, we firstly study the relationship between link utilization and delay. A number of systems models have been proposed to describe different characterized IP networks in the last years. Here each edge is modeled as an independent M/M/1 queuing system [32]. To apply this queueing model in this work, the following assumptions are used: 1) each queue has an exponentially distributed mean service time; 2) an average arrival rate of new packets, which follows a Poisson distribution; 3) the packet length is also exponentially distributed; 4) the network structure should have a fixed routing, where the channels are error-free, etc. If mean packet length is $E[l_P]$, mean packet delay is derived as Eq.3.

$$t_{M/M/1} = \frac{E[l_P]}{\mu_e} \cdot \frac{1}{1 - \rho_e} \quad (3)$$

If ρ_e increases, mean packet delay will be incremented, too. The maximum link utilization allows us to keep the expected delay low.

The second constraint in this approach is the *maximum hop number* between source and destination, which influences delay, reliability and survivability of networks. The hop number is given as number of links one packet has to pass on its way from the source to the destination node. A network solution is only valid, if the maximum hop number is not exceeded by any relevant source-destination combination. Thus, by limiting the maximum number of used edges per routed path an additional indirect constraint for a low end-to-end delay is given. Generally, every link has a certain reliability. By limiting the maximum hop number also the end-to-end reliability is manipulated, because along one path the single link reliabilities affect the total end-to-end reliability. Moreover, too high hop number makes it difficult to apply some routing protocols, such as Distance-vector Routing Protocol. The actual propagation delay on the edges is not directly incorporated in the optimization algorithm, which can be studied as further work.

In addition, the proposed algorithm also takes into account the *maximum node degree* (maximum number of links which are connected to a node). For some network planning tasks the *maximum node degree* was a limiting requirement in [8][33][34].

Furthermore, network survivability should be fulfilled during the redesign. The link restoration assumes single, total failures of individual links and restores the entire (or partial) capacity of the failed link on one or several paths between two end nodes of the link [6]. In this work the edge/node-biconnectivity is applied to protect single link or node failures during the network redesign.

3 Graph Analysis

3.1 Node/Edge-Connectivity

A graph is connected, if there exists at least one path from any point to any other point in the graph; otherwise the graph is called disconnected. The edge (or node) connectivity of a graph *G* is the minimum number of edge (or node) deletions sufficient to disconnect *G*, which is characterized by Menger's theorem [35]. In view of the disjoint paths, an undirected graph $G = (V, E)$ is defined as *k*-edge-connected, if there are *k* paths between two nodes $v, v' \in V$ and these paths do not share any edge. If these paths do not have node between v and v' in common, graph G is defined as *k*-node-connected. Two connected graphs are shown in Fig.1, where the left tree-topology is a *1*-edge/node-connected graph (or edge/node-connected) and the right mesh-topology is 3-edge/node-connected. The term mesh does not imply that the network topology is a full mesh, but rather that the network is at least two (edge) connected [36][23]. Therefore, the mesh structure has a higher survivability than the tree structure in telecommunications networks, but is more expensive due to additional edges. There is a close relationship between edge- and node-connectivity. The node connectivity is never smaller than the edge connectivity, since deleting one node incident on each edge in a cut set succeeds in disconnecting the graph [37][38].

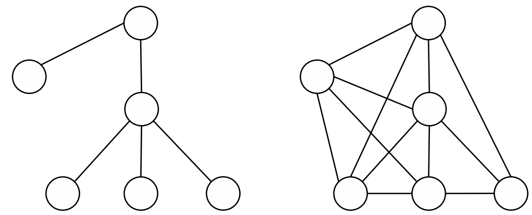

Figure 1. Connected graph: tree and mesh

3.2 Biconnectivity

A graph is called two-edge connected (edge-biconnected), if there are at least two edge-disjoint paths between every pair of nodes. Similarly, a graph is called two-node connected (node-biconnected), if there are at least two node-disjoint paths between each pair of nodes. An example is shown in Fig.2. Every node-biconnectivity graph is also edge-biconnectivity, while the reverse is not valid [23]. A biconnected topology can effectively ensure the network survivability. If an efficient method is applied to change a graph from one-connectivity to biconnectivity, the network cost will not be significantly incremented. For instance, adding only a few necessary links can make a tree topology biconnected. Hence, this work addresses the node-biconnectivity to improve the network survivability, where two node-disjoint paths can be available for the flows between source and destination.

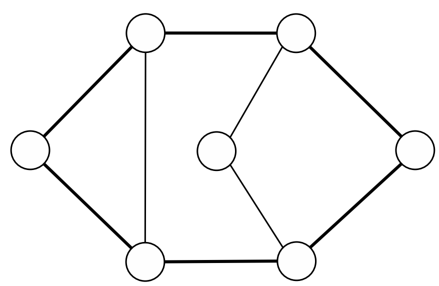

Figure 2. Two-node/edge-connected topology

3.3 Block Structure and Articulation

The *articulation* of a connected graph is a node whose removal will disconnect the graph [39]. Fig.3 shows two graphs with articulation nodes, which are depicted with gray color. The right graph represents a graph with 1-node-connectivity, but 3-edge-connectivity for more than two edge-disjoint paths for any pair of nodes.

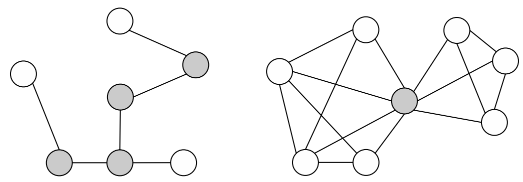

Figure 3. Examples with only 1-node-connectivity

A *block* is defined as a maximal biconnected subgraph for an undirected graph $G = (V, E)$. If this graph is biconnected, G itself is called a block. If graph G has N blocks and $i, j \in [1, N]$, $G_i = (V_i, E_i)$ is defined as block *i* with

(a) $|V_i \cap V_j| \leq 1$ for $i \neq j$;

(b) articulation node $a \in V$, if $|V_i \cap V_j| = \{a\}$ for $i \neq j$.

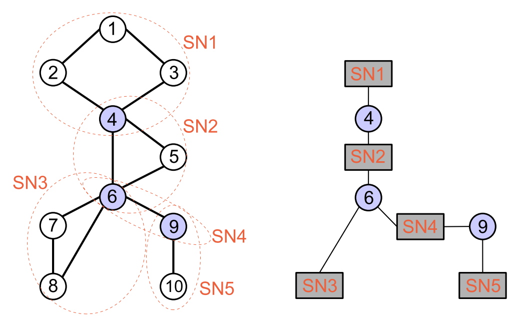

Figure 4. Graph G and its block graph *B(G)*

The block structure is defined as a block graph *B(G) = (V',E')* of graph *G*. *B(G)* is made up of blocks and articulation nodes, which can be found by a modified *Depth First Search* (DFS) [29] based on the method of Tarjan [40]. In Fig.4, the left graph presents a 10-nodes-topology with 5 subnets (SNs), the right graph presents a block graph with 5 blocks (square) and articulation nodes *4, 6, 9*.

4 Description of the Algorithm

4.1 Notation

In the following an overview of the used notations is given:

E_0 Set of edges representing the existing links (active);

E_R Set of edges representing the links (active) after the *Reduction*;

E_A Set of edges representing the links (active) after the *Augmentation*;

E_F Set of edges representing the links (active) after the *Fine-Tuning Reduction* $\longrightarrow$ Solution set;

E_{FIX} Set of edges representing the fixed links (active);

E_{POT} Set of edges representing the potential links (inactive);

E_{RED} Set of edges representing the reduced links (inactive) during the *Reduction*;

E_{AUG} Set of edges representing the augmented links (active) during the *Augmentation*;

E_{FTR} Set of edges representing the Fine-Tuning reduced links (inactive) during the *Fine-Tuning Reduction*;

w_{cost} Weight of cost for calculating cost-metric;

$w_{capacity}$ Weight of capacity for calculating cost-metric during the *Reduction*;

w_{flow} Weight of flow for calculating cost-metric during the *Augmentation*;

$w_{utilization}$ Weight of utilization for calculating cost-metric during the *Fine-Tuning Reduction*.

Note: The *active edges* are a part of the current network and thus they are used during the network calculation. The *inactive edges* are not a part of the current network.

4.2 Redesign Strategy

The complete redesign procedure consists of *Reduction*, *Augmentation*, *Fine-Tuning-Reduction*, as shown in Fig. 5.

Graph theory helps us to formulate the problem as follows (*nodes* are represented by *vertices* and *links* by *edges*): Let $E_0 \subset E$ be a fixed set of operational edges (representing the existing links), such that $G(V, E_0)$ is connected. And Let $E_{POT} \subset E$ be a fixed set of given edges (representing the potential links), such that $E_0 \cup E_{POT} = E$. The biconnectivity problem can be subdivided as follows:

1. The first step (*Reduction*) is to find a set $E_{RED} \subset E_0$ with $E_R = E_0 - E_{RED}$ which reduces the number of edges (the network costs) to a minimum without violating the constraints.

2. The second step (*Augmentation*) is to find a set $E_{AUG} \subset (E_{POT} \cup E_{RED})$ of augmenting edges with minimal costs, such as the biconnected graph $G(V, E_A, W)$ with $E_A = E_{AUG} \cup E_R$.

3. On augmenting E_R to E_A it is possible that edges from E_R become redundant for the graph biconnectivity. Hence, the third step (*Fine-Tuning-Reduction*) is to further reduce the network cost by finding a set $E_{FTR} \subset E_R$. Then the number of edges are decremented to $E_F = E_A - E_{FTR}$ subject to constraints and biconnectivity.

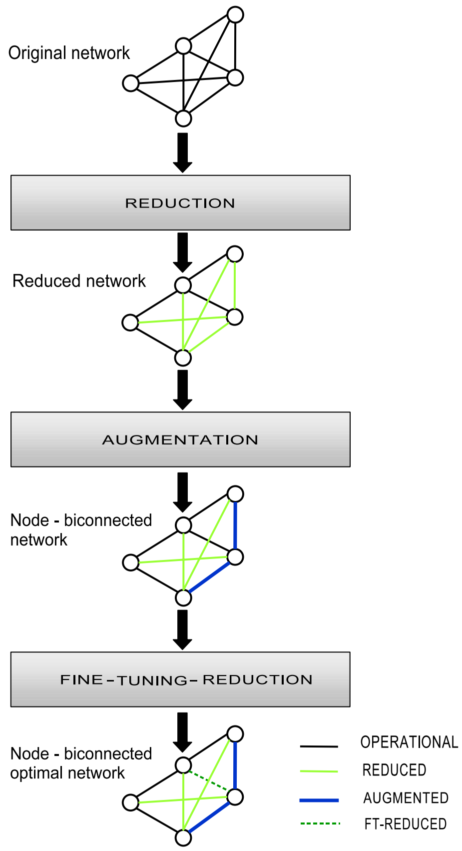

Figure 5. Flow chart of the redesign strategy

It is assumed that the network consisting of initial links (E_0) and potential links (E_{POT}) must fulfill the constraints and guarantee the biconnectivity.

The advantage of our edge redesign is that the history of the network and the experience of the network planner is taken into consideration because *Reduction* bases on the set of edges E_0 which represents the real existing links. Moreover, we introduce another possibility to further influence the direction of optimization by implementing the set $E_{FIX} \subset E_0$. Edges of the set E_{FIX} are not allowed to be reduced by the algorithm and hence they constitute a definitive part of the solution set E_F. E_{FIX} makes it possible to consider practical relevant situation, e.g. a long term of a leased line link. (Note that real existing links as well as potential links can be assigned to the set E_{FIX} at the beginning of the optimization.)

4.3 Reduction

Suppose E_0 is the set of edges in a graph *G*. The objective of the *Reduction* is to find a set E_{RED} that decreases the set of edges E_0 to a minimum E_R ($E_R, E_{RED} \subset E_0$ and $E_R \cup E_{RED} = E_0$) such that E_R is a graph with minimal number of edges (costs) still fulfilling the constraints (maxHop, maxUtil, maxNodeDegree). Ideally, this graph is a minimal spanning tree. Any edge $e \in E_0$ is evaluated by a cost metric $Metric_e^{RED}$ with normalized cost c_e and bandwidth μ_e:

$$Metric_e^{RED} = \frac{w_{cost} \cdot min\{c_{e,e\in E_0}\}}{c_e} + \frac{w_{capacity} \cdot \mu_e}{max\{\mu_{e,e\in E_0}\}} \quad (4)$$

with constraints:

$$w_{capacity} + w_{cost} = 1$$

$$0 \leq w_{capacity} \leq 1$$
$$0 \leq w_{cost} \leq 1$$

$$0 < Metric_e^{RED} \leq 1$$

The normalized values are used to efficiently represent the cost metric. The cost and capacity weights influence the evaluation of the cost metric. We assumed a cost function for the links, which is derived and estimated from basis network rate for leased lines of the *Deutsche Telekom* (2004) [41]. Depending on the link capacity and length, the costs consist of a base rate and a piecewise linear increasing cost function (CU: *Cost Unit*), as shown in Fig. 6. In a sense, the leasing cost of *10* Gbit/s is less than the leasing cost of *2.5* Gbit/s multiplied by *4*. Therefore, the edges with high cost (long optical cable) and low capacity will be preferably removed. The smaller $Metric_e^{RED}$ is, the earlier the edge *e* is reduced. During the *Reduction* the constraints have to be fulfilled, such as capacity, utilization, etc.

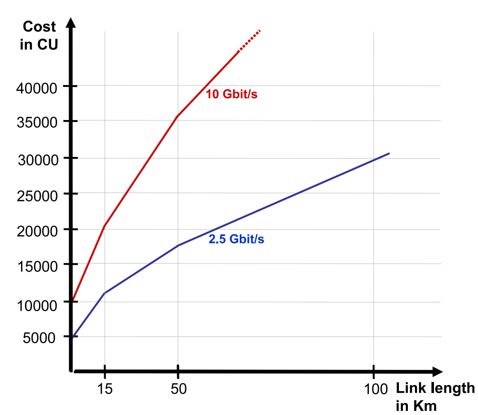

Figure 6. Cost function for calculation of leased line link cost per year in terms of link capacity and link length

We suppose that all demands are routed, and the edge loads, edge utilizations are calculated. The *Reduction* algorithm can then be described as follows:

1. The edge weights composed of a weighted standardized sum of real edge cost and load, are calculated for all edges of E_0.

2. On basis of the edge weights one edge is selected and temporarily added to the set E_{RED}: Thus, the selected edge is deactivated.

3. The demands are rerouted and then edge loads and edge utilizations are calculated considering all edges belonging to the sets E_0 and E_R.

4. If all constraints are fulfilled, the selected edge is reduced, which means it is finally added to the set E_{RED}. If a constraint is not fulfilled, the selected edge is marked as required, which means it is added to the set E_R.

5. The algorithm recalculates the edge weights and selects the next edge.

This process is repeated until all edges from E_0 are transferred to the sets E_{RED} or E_R. The program flow chart (PFC) of the *Reduction* is depicted in Fig. 7.

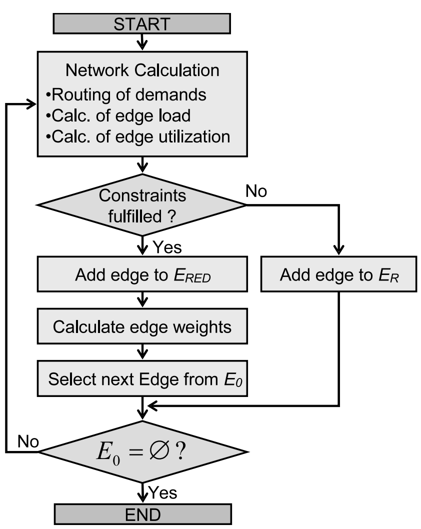

Figure 7. Program flow chart of the *Reduction*

Note that the *Reduction* algorithm solution depends on the sequence of selected edges. Hence, a unique branch of the complete combinatorial solution tree is calculated.

4.4 Augmentation

The objective of the *Augmentation* is to find out a set E_{AUG} from sets E_{RED} and E_{POT} so that the graph *G* with the edges $E_A = E_R \cup E_{AUG}$ between vertices is biconnected. In Fig. 8 the PFC of the *Augmentation* is depicted. At first the graph is analyzed by means of the modified *Depth First Search* (DFS) mentioned in the last section. The DFS finds all articulation points of *G*, marks them, and splits the graph into its biconnected components (subnets). If the graph is biconnected without any articulation point), the *Augmentation* will be stopped, otherwise the next *Augmentation* step is done.

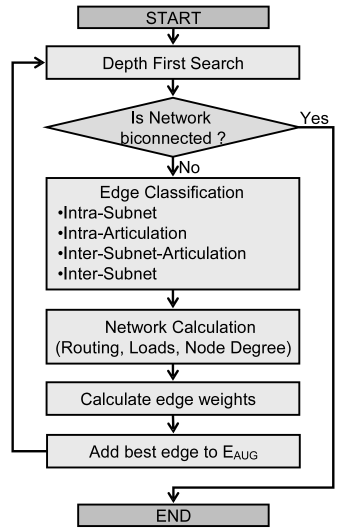

Figure 8. Program flow chart of the *Augmentation*

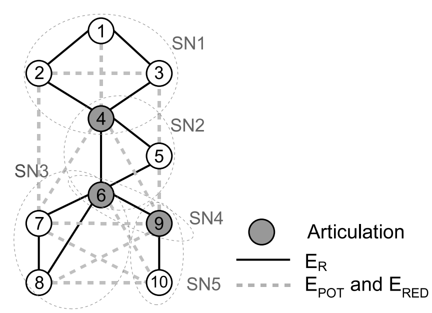

Figure 9. *Augmentation* network example: DFS marks articulation points and divides network into subnets: In black are the edges of the set E_R and in gray (dashed lines) are the edges depicted of the sets E_{RED} and E_{POT} (potential candidates for the *Augmentation*). The three articulation points are marked with a dark color and the different subnets are surrounded by dashed lines.

Figure 9 shows an example of a not biconnected graph after the DFS. Next, we classify the potential candidate edges in a way that only edges between vertices of different

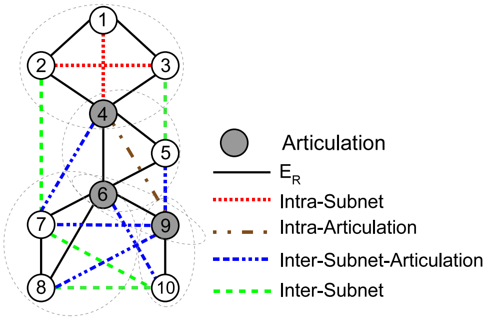

Figure 10. *Augmentation* network example: Classification of the potential candidate edges (E_{POT} and E_{RED}) depending on source and destination vertex

subnets are accepted as candidates for the *Augmentation*, while no vertex is an articulation point (Inter-Subnet). In Figure 10, the edge classification is shown for our network example. Due to the classification Intra-Subnet edges like {2,3} or {1,4} are not considered. Furthermore, we disregard Intra-Articulation edges like {4,9}, since their introduction would not improve the network situation regarding to the *Augmentation* problem. As a rule, we do not consider edges between different subnets, if one vertex is an articulation point (Inter-Subnet-Articulation), e.g. {7,9} or {10,6}, because in comparison with the Inter-Subnet edges they are only the second best option. Only in case that the graph is not biconnected, and no Inter-Subnet edge is left, these edges are accepted.

After that, a special cost metric $Metric_e^{AUG}$ is used to evaluate edge e subject to edge cost, subnet flow and node degree:

$$Metric_e^{AUG} = \frac{w_{cost} \cdot min\{c_{e,e\in(E_{POT}\cup E_{RED})}\}}{(d_v + d_{v'}) \cdot c_e} + \frac{w_{flow} \cdot f_e^{sn-sn}}{(d_v + d_{v'}) \cdot max\{f^{sn-sn}_{e,e\in(E_{POT}\cup E_{RED})}\}} \qquad (5)$$

with constraints:

$$w_{flow} + w_{cost} = 1$$

$$0 \le w_{flow} \le 1$$

$$0 \le w_{cost} \le 1$$

$$0 < Metric_e^{AUG} \le 0.5$$

where v and v' are end nodes of edge *e* between two blocks (unbiconnected subnets). d_v and $d_{v'}$ are node degrees of v and v', respectively. f_e^{sn-sn} is the flow over edge *e* between two subnets. $min\{c_{e,e\in(E_{POT}\cup E_{RED})}\}$ and $max\{f^{sn-sn}_{e,e\in(E_{POT}\cup E_{RED})}\}$ present the minimal cost metric and maximal flow for $e \in (E_{POT} \cup E_{RED})\}$. The existing network topology is connected so that d_v and $d_{v'}$ are no less than *1*. Hence, $Metric_e^{AUG}$ is between *0* and *0.5*. Edges are preferably introduced when they are not costly, highly loaded (due to a high subnet-subnet-flow) and where the average degree of source and destination vertex is small. More precisely, the edges with a high cost metric preferably added.

Regularly for each edge a single routing has to be done where all other candidates are deactivated. Due to the time complexity of calculating the edge loads, an estimator is used. Therefore, all candidates are temporarily added to E_{AUG}. Then a routing of the demands and a load calculation is done considering the edges of E_{AUG} and E_R. On basis of the weights, one edge is selected and then definitively added to the set E_{AUG}, the remaining temporarily added edges are retransferred to their former sets. The described procedure is repeated until the DFS algorithm confirms biconnectivity of the graph.

4.5 Fine-Tuning Reduction

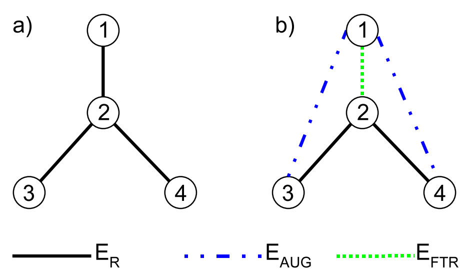

Figure 11. *Fine-Tuning Reduction* example: a) shows graph after *Reduction* b) shows that due to the *Augmentation* one edge from E_R becomes mathematically redundant for the biconnectivity of the graph

On augmenting E_R to E_A it is possible that edges from E_R become redundant for the graph biconnectivity of E_A. The example in Fig. 11 shows a graph, where E_R has the shape of a tree structure (Fig. 11a). The *Augmentation* added the minimal number of two edges {1,3}, {1,4}. Hence, E_A is biconnected. From the viewpoint of graph theory, the graph would still be biconnected, if we discard the edge {1,2} of the former tree structure (Fig. 11b). Thus, objective of the *Fine-Tuning Reduction* is to find a set $E_{FTR} \subset E_R$ that reduces the number of edges to $E_F = E_A - E_{FTR}$ with a minimum total cost, considering constraints and graph biconnectivity at any time. However, we only consider edges of the set E_R as candidates for the *Fine-Tuning Reduction*, because if the *Augmentation* performs well hardly edges from E_{AUG} will become redundant.

The PFC of the *Fine-Tuning Reduction* is in general the same as the PFC of the *Reduction* (Fig.7). At the beginning we have a set of edges E_A. At each loop one edge out of the former set E_R is selected and temporarily added to the set E_{FTR}. Then this edge is deactivated. Then the network is calculated, such as routing of the demands, calculation of edge loads and edge utilizations. After that, the selected edge is either definitely assigned to the set E_F, if a constraint is not fulfilled or the remaining graph is not biconnected. Or it is assigned to the set E_{FTR}, if the graph is still valid. This procedure is repeated until all edges of the former set E_R have been tested. The sequence of tested edges is defined by the edge weights, which are recalculated in every cycle. The cost metric $Metric_e^{FTR}$ is composed of a weighted, standardized sum of edge leasing cost c_e and edge utilization ρ_e in Eq.6. The edges with a low cost metric will be firstly considered to be reduced.

$$Metric_e^{FTR} = \frac{w_{cost} \cdot min\{c_{e,e\in E_R}\}}{c_e} + \frac{w_{utilization} \cdot \rho_e}{max\{\rho_{e,e\in E_R}\}} \tag{6}$$

with constraints:

$$w_{cost} + w_{utilization} = 1$$

$$0 \leq w_{cost} \leq 1$$

$$0 \leq w_{utilization} \leq 1$$

$$0 < Metric_e^{FTR} \leq 1$$

4.6 Random Selection

To avoid trapping into local optima, a random process is applied to determine which edges are removed or added. In addition, we assume $pThres_{RED}$, $pThres_{AUG}$, $pThres_{FTR}$ as potential thresholds for *Reduction*, *Augmentation* and *Fine-Tuning-Reduction* to efficiently limit the search space. The random selection is described as following:

1. Evaluate cost metrics for all candidate edges and sort metrics from the minimal metric (*Min_Metric*) to the maximal metric (*Max_Metric*);

2. Find all edges with metric $\geq Min\_Metric(1 + pThres_{RED})$ for *Reduction*, or all edges with metric $\leq Min\_Metric(1 - pThres_{AUG})$ for *Augmentation*, or all edges with metric $\geq Min\_Metric(1 + pThres_{FTR})$ for *Fine-Tuning-Reduction*;

3. Select one edge by means of a uniformly distributed random number.

5 Results and Analysis

5.1 Redesign Environment

The algorithm is implemented in C++ and applied using a commercial network planning tool - NetWorks [42], designed for optimizing large scale telecommunications and IP networks. We studied the practical behavior of the algorithm on a set of real network examples. Two test networks will be introduced in this section. The optimization environment is based on a standard PC (Pentium III with *800* MHz and *384* Mbyte RAM). Each scenario with different parameters has been tested for at least 10 times.

We estimated demands between the nodes by applying the gravity coefficient method [43]. Therefore, for each pair of nodes the demands were calculated in direct proportion to the population of the corresponding area, and in indirect proportion to the distance between the individual nodes. Furthermore, the edge cost is derived from the leasing lines, see Fig. 6 and Eq. 2.

The information on the following networks has been collected from public sources. Since the traffic matrices haven't been available, the previously mentioned gravity model was used to generate one. This means that the following results are not absolutely applicable to the studied networks. But this is not the intention of the study. The idea is to show the application of the algorithm using real life topologies.

5.2 Test Network I: G-WiN

G–WiN was a part of Germany Research Education Networks by DFN-Association [44], which connects *42* core nodes by *75* links at two levels. The bandwidth of links ranges from 622 Mbit/s to 10 Gbit/s. In this approach we focus on the 10 core nodes (level 1) and their connecting 21 links (see Fig. 12). The potential edges are assumed to have a capacity of 2.5 Gbit/s. Furthermore, the maximal edge utilization is 99.9999% without limit of maximal hop number and node degree.

Fig. 12 shows the original links used for *G–WiN*. As result of the redesign redundant and high capacity (expensive) edges are successfully removed in Fig. 13. During the redesign all constraints have to be fulfilled.

More details are presented in the Tab. 1. The cost and number of links are significantly reduced. Mean edge utilization is increased, which should be taken into account. However, the original average link utilization was low. After the redesign the maximum utilization value is decreased. Hence, this increment of the edge utilization has no remarkable influence on the redesigned *G–WiN*. The reduced node degree can save the ports of core nodes.

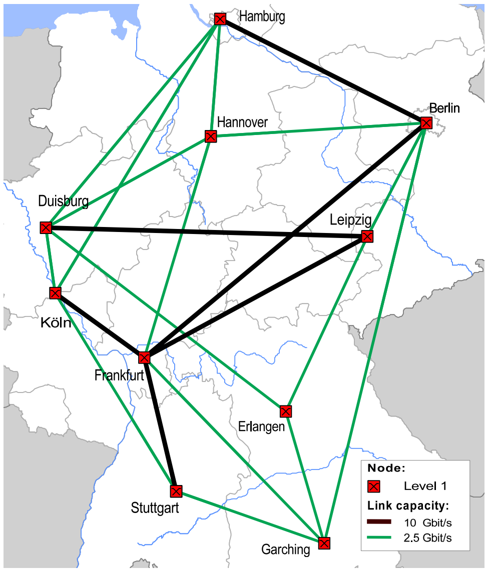

Figure 12. Original *G–WiN* (level 1)

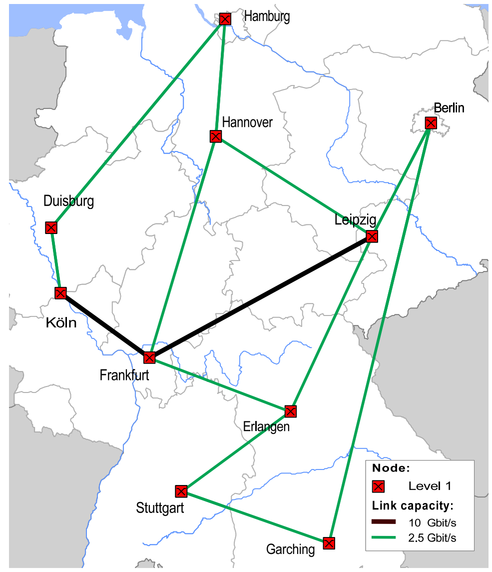

Figure 13. *G–WiN* after the redesign

Table 1. Optimization results of *G–WiN*

G–WiN	Original	Average
Cost in cost unit (CU)	1 402 247.49	671 915.49
Number of edges	21	13
Mean edge utilization	17.5%	41%
Max. edge utilization	97%	93%
Mean hop number	1.64	2.29
Max. hop number	3	5
Mean node degree	4.2	2.6
Max. node degree	6	4

5.3 Test Network II: AT&T

The level one and two of the *AT&T* backbone network (shown in Fig. 14a) is applied to redesign, which is comprised of *37* nodes in major US cities and *58* links (*10* Gbit/s or *2.5* Gbit/s link capacities) interconnecting them [42].

Figure 14 shows the procedure of the algorithm from the initial network with the existing links (set E_0) and the potential links (set E_{POT}) and to the final biconnected solution with the following settings and constraints:

- all links of the set E_{POT} have a capacity of *2.5* Gbit/s
- maximum link utilization = *99%*
- maximum hop number = *16*
- maximum node degree = *4*

Figure 14b) depicts the network after the *Reduction* (E_R = *37* links). The algorithm almost achieves a pure tree structure, only the *10* Gbit/s link $\{$*Chicago*, *New York*$\}$ can not be reduced due to the effect of a too high utilization resulting from its removal. During the Augmentation *18* links (with dashed lines marked links in Fig. 14c) are added to the network in order to make it biconnected, whereupon the *Fine-Tuning Reduction* discards *15* links again (set E_{FTR} in Fig. 14d) taking into account the constraints and biconnectivity. Figure 15 compares the network topology before and after the optimization. The corresponding optimization results are summarized in Table 2.

Table 2. Optimization results of *AT&T* backbone network

AT&T	Original	Best	Average
Cost in CU	8 901 599	3 712 977	4 259 874
Number of edges	58	40	42.1
Mean edge util.	16%	51%	41.8%
Max. edge util.	54%	96%	92.4%
Mean hop number	3.47	6.59	6.1
Max. hop number	9	16	15.5
Mean node degree	3.13	2.14	2.25
Max. node degree	9	3	3.5

As mentioned, the solution depends on the sequence of edges (selected during *Reduction*, *Augmentation* and *Fine-Tuning Reduction*). If the edge with the best weight is always selected, the same unique branch of the complete combinatorial solution tree will be generated again and

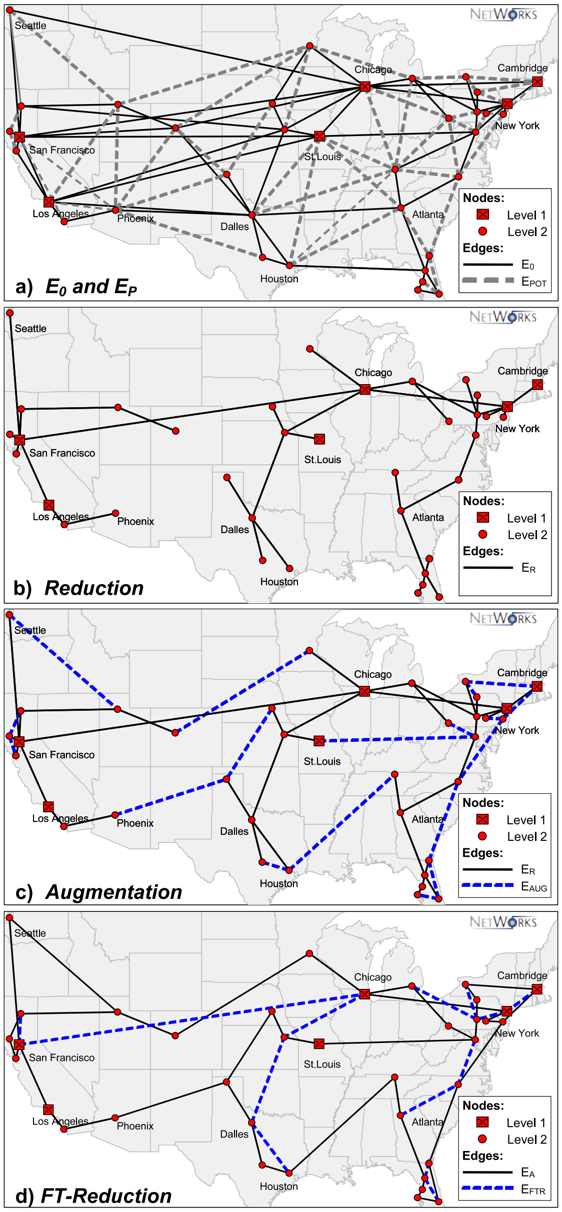

Figure 14. *AT&T* backbone network redesign: Successive depiction of the network after every optimization step. a) initial network, b) after *Reduction*, c) after *Augmentation*, d) after *Fine-Tuning Reduction*

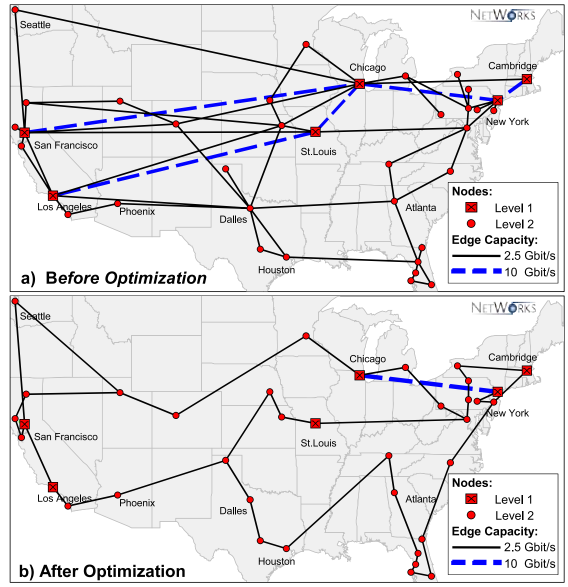

Figure 15. *AT&T* backbone network: Link capacities a) before and b) after optimization (maxHop = 16). The optimization process is illustrated in Fig. 14

again. But the deterministic solution is not always the best. Therefore, we implemented a random function that controls edge selection such that not always the best one is selected, but also the second or third best. For the *AT&T* network optimization we generated 50 solution sets (E_F) with an individual set of parameters. Afterward a statistical analysis has been performed and evaluated the efficiency of the chosen parameter. The algorithm took about *30* seconds calculating one *AT&T* network parameter set (one E_F). The result presented in Table 2 (column *Best*) and Fig. 15 is the best one out of fifty solution sets. The average values of the fifty solution sets are shown in column *Average* of Tab. 2.

Finally network cost are reduced to less than 50%. But the price is high: the mean link utilization, mean hop number and maximum hop number are increased. This results from a weakly restricted optimization. Hence, what we see is the potential saving of cost, if the network is operated at the limit and in consequence with a higher probability of having bad quality of service (QoS). This can be improved by tightening up the constraints. The degree of meshing can be decreased by adjusting the maximum hop number or the maximum link utilization. Our tests showed, that the hop number constraint is a better control because routing algorithm mostly finds the shortest path (with the least hop number), but does not consider link utilization. So a maximum utilization constraint can be easily exceeded and thus disturb the success of the optimization result (network could be not valid), although traffic engineering methods like load sharing or routing metric optimization [45] could balance network utilization situation.

In Fig. 16 an optimization result is shown with the same parameters like in the previous example, but with a maximum node degree of *5* and a maximum hop number of *9*. Thus, the maximum hop number is as big as in the original *AT&T* network. This optimized network has *51* active links and costs 4 284 965 CU. The average link utilization is *31%*.

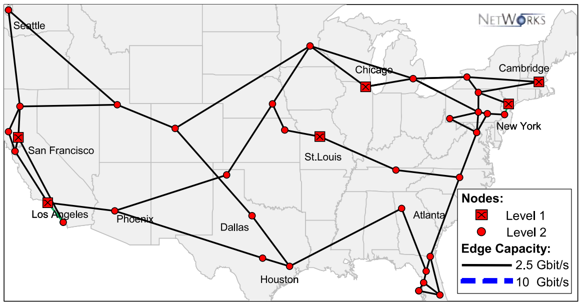

Figure 16. *AT&T* backbone network: Link capacities after optimization with constraints: max. Hop = *9*; max. link utilization = *99%*; max. node degree = *5*

Although maximum utilization is still high (*96%*), the network performs much better, since only one link is utilized with more than eighty percent and the average hop number is *4.59*. However, too high utilization has negative influence on the network operation. Therefore, the maximum utilization can be decreased from 99% to a lower threshold, or the mechanism of sharing load can be applied to reduce the link utilization. This paper provides only some results based on the predefined scenarios to show the characteristics of three-steps algorithms.

An interesting effect is the counter balancing behavior of the three optimization steps. Imagine an unfavorable *Augmentation*, adding a more than necessary number of edges to the graph. This does not necessarily lead to a bad overall result, because the supplementary edges also introduce additional optimization potential to the *Fine-Tuning Reduction*. In this case *Fine-Tuning Reduction* can compensate the bad *Augmentation* performance.

Besides the described directed optimization (network redesign) also the opposite is possible (designing new networks) by assigning all potential and real existing edges to the initial set E_0. Thus, the algorithm has all degrees of freedom to create a completely new solution set. Applying the new design method, our tests show e.g. for the *AT&T* network (Fig. 15, same constraints) a best solution set with minimal cost of over again 10% less compared to the best redesign method.

5.4 Parameter Studies

Different parameters influence the overall performance of the proposed algorithm. This paper presents some results of cost metrics, maximal hop number and potential threshold for network calculation and redesign. The points in the following figures are derived from 50 trials with confidence level 95%. Furthermore, a specified new-design is defined. Only one difference from the redesign is that this new-design takes all active and potential edges as operational edges, i.e. E_0 is extended. Referring to empirical results, both design processes are similar. Hence, the new-design of *AT&T* network is fulfilled to evalute the parameters mentioned above. The new-design has no limitation of node degree, but with maximal hop number and maximal edge utilization.

5.4.1 Reduction–Metric

By evaluating the cost metric for *Reduction*, w_{cost} and $w_{capacity}$ play an important role. As mentioned in *Reduction*, the metrics with low values will be removed with a higher probability. In addition, the weight of cost has more positive influence in the redesign than the weight of capacity. Namely, the expensive edges are preferably removed. In terms of w_{cost} and $w_{capacity}$, Fig. 17 presents optimization results after *Reduction*, *Augmentation* and *Fine-Tuning-Reduction*. A high weight of cost can lead to a better optimization solution, e.g. the right side of the figure w_{Cost} close to 1.0. Several unusual slight increment on the curves, such as Fine-Tuning-Reduction, stand the reason that the setting of w_{cost} and $w_{capacity}$ also depends on other parameters and could lead local optima.

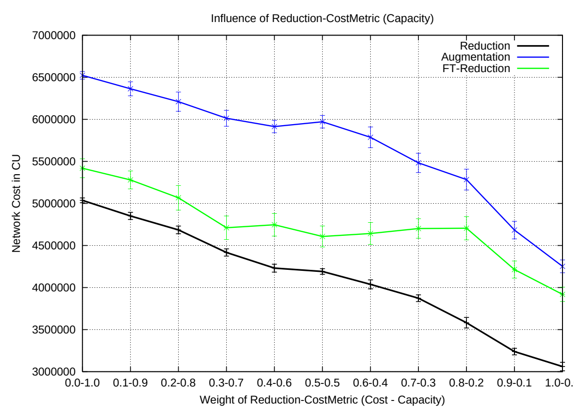

Figure 17. Influence of w_{cost} and $w_{capacity}$ for Reduction–Metric on overall network cost:

Constraints:	$max.Hop = 20$	$max.Util. = 99.999\%$
Random in %:	$pThres_{RED} = 20$	$pThres_{AUG} = 10$
$Metric_e^{AUG}$:	$w_{cost} = 0.7$	$w_{flow} = 0.3$
$Metric_e^{FTR}$:	$w_{cost} = 0.7$	$w_{utilization} = 0.3$

5.4.2 Augmentation– and FT–Reduction-Metrics

Instead of $w_{capacity}$, the cost metrics of Augmentation– and FT–Reduction-Metrics take advantage of w_{flow} and $w_{utilization}$. Subject to hard constraints during the design, the candidate edges for both steps are strictly limited. Hence, Augmentation– and FT–Reduction-Metrics have similar characteristics in terms of optimization results. Hereby, only the optimization results of Augmentation-Metrics are presented and analyzed. Fig. 18 compares w_{cost} and w_{flow} of the Augmentation–Metric for three steps. The change of weights w_{cost} and w_{flow} has no explicit improvement for network design. However, the solutions with w_{cost} close to 1 are a little better than those with w_{cost} near 0.

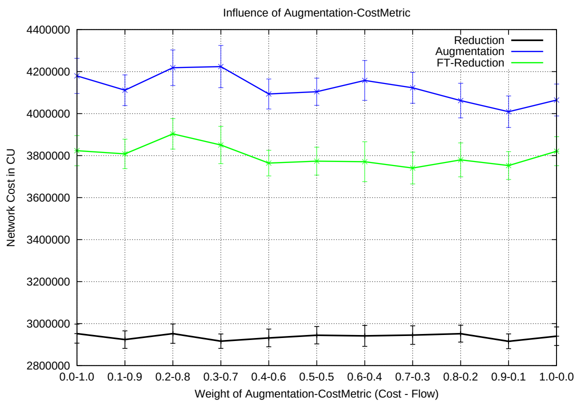

Figure 18. Influence of w_{cost} and w_{flow} for Augmentation–Metric on overall network cost:

Constraints:	$max.Hop = 20$	$max.Util. = 99.999\%$
Random in %:	$pThres_{FTR} = 10$	$pThres_{AUG} = 10$
$Metric_e^{Rdkt}$:	$w_{cost} = 1.0$	$w_{capacity} = 0.0$
$Metric_e^{FTR}$:	$w_{cost} = 1.0$	$w_{utilization} = 0.0$

5.4.3 Maximal Hop Number

The maximal number of hops has been mentioned for *AT&T* network redesign. More test results are shown in Fig. 19 for *AT&T* network new–design. The lower the maximal number of hops is, the more expensive the designed network topology is. In this case more links are required to save hops for point–to–point connections and the node degree (maximal and mean) can increase. Therefore, maximal hop number significantly influences design results. However, if the node degree is limited, the influence of the maximal hop number could be changed.

5.4.4 Potential Threshold

Fig. 20 provides an overview of the optimization results in terms of different potential thresholds for *Reduction*, i.e. $pThres_{RED} \in [0\%, 700\%]$ with constant $pThres_{AUG}$ and $pThres_{FTR}$. The lower thresholds can lead to better solutions. Due to limited candidate edges, different potential thresholds of $pThres_{AUG}$ and $pThres_{FTR}$ have no large change. Hence, they are disregarded in this paper.

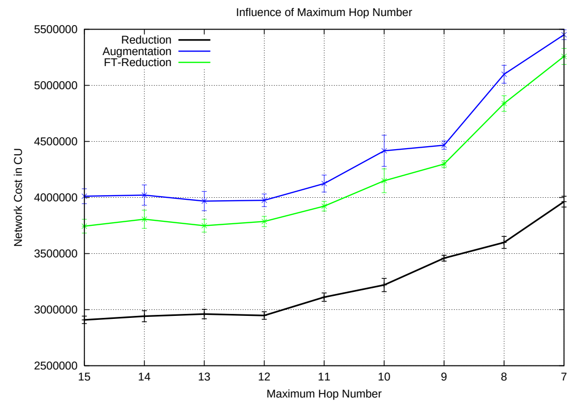

Figure 19. Influence of maximal hop number on network cost:

Constraints:	$max.Util. = 99.999\%$	
Random in %:	$pThres_{RED} = 10$	$pThres_{AUG} = 10$
$Metric_e^{RED}$:	$w_{cost} = 1.0$	$w_{capacity} = 0.0$
$Metric_e^{FTR}$:	$w_{cost} = 1.0$	$w_{utilization} = 0.0$

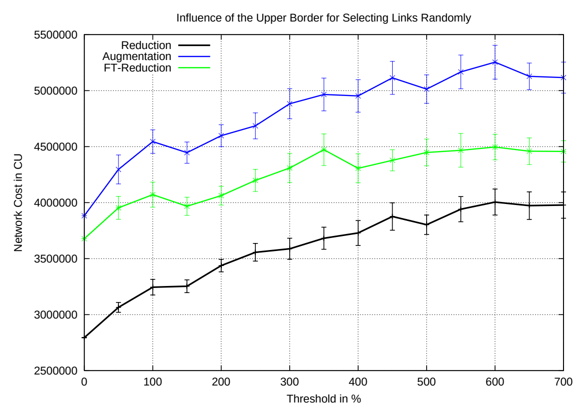

Figure 20. Influence of potential thresholds on overall network cost:

Constraints:	$max.Hop = 20$	$max.Util. = 99.999\%$
Random in %:	$pThres_{AUG} = 0$	$pThres_{FTR} = 0$
$Metric_e^{RED}$:	$w_{cost} = 1.0$	$w_{capacity} = 0.0$
$Metric_e^{FTR}$:	$w_{cost} = 1.0$	$w_{utilization} = 0.0$

5.4.5 Summary

With the different configuration of parameters, several networks have been investigated. Tab. 3 summarizes the most useful parameter set, found empirically during the study.

6 Conclusion

In this paper we proposed a three-steps algorithm for network redesign problems to find a new fully biconnected

Table 3. The best found parameter setting for IP–Network design

Parameters	Values
Reduction:	
$Metric_e^{RED}$	$w_{cost} = 0.9 \quad w_{capacity} = 0.1$
$pThres_{RED}$	10%
Augmentation:	
$Metric_e^{AUG}$	$w_{cost} = 0.8 \quad w_{flow} = 0.2$
$pThres_{AUG}$	$0\% \leftrightarrow 10\%$
FT–Reduction:	
$Metric_e^{FTR}$	$w_{cost} = 0.8 \quad w_{utilization} = 0.2$
$pThres_{FTR}$	0%

cost-effective network topology. Hard constraints, such as hop number, edge utilization and node degree, are considered. In terms of numerical simulation results, the algorithm was verified to be flexibly applicable for redesign existing networks and creating new networks. The implemented algorithm has successfully applied to different network topologies, fulfilling the constraints and considering the biconnectivity, particularly with two-vertex connectivity. The redundant edges can be efficiently removed by the third step - *Fine-Tuning Reduction*, which improves the quality of network redesign in comparison with the previous work. A parameter study helps understanding the sensitivity of the results on algorithm settings.

In real network design situations, keeping the network operational is the most important task. A cost saving of e.g. 5% will always be critically examined by the network operator, since the occurrence of critical transition states during the migration has to be taken into consideration. Furthermore, network requirements continuously change, due to new services, new customers and new technologies. Thus, for a network operator it is not crucial to have (a temporarily) optimal network, but it is more important to know the optimal network. This knowledge helps making the right (most cost efficient) decision for necessary network extensions. The presented strategy supports network planners in these situations. Its high transparency and the possibility to closely interact by adjusting intermediate results manually assures its applicability.

References

[1] C. Minge, R. Zhao, M. Schweigel, "Improved Optical Network Topology Redesign Ensuring Biconnectivity", The Fourth Advanced International Conference on Telecommunications (AICT2008), p.398–403, Athens, Greece, June, 2008

[2] G.M. Schneider, M.N. Zastrow, "An Algorithm for the Design of Multilevel Concentrator Networks", Computer Networks, vol.6(1), p.1–11, 1982

[3] B. Gavish, "Topological design of Telecommunication Networks-the Overall Design Problem", European Journal of Operations Research, vol.58(2), p.149–172, 1992

[4] J. Petrek, V. Siedt, "A Large Hierarchical Network Star-Star Topology Design Algorithm", European Transactions on Telecommunications, vol.12(6), p.511–522, 2001

[5] A. Bley, T. Koch, R. Wessaely, "Large-scale hierarchical networks: How to compute an optimal architecture?", Proceeding of Networks 2004, p.13–16, Vienna, Austria, 2004

[6] M. Pioro, Deepannkar Medhi, "Routing, Flow, and Capacity Design in Communication and Computer Networks", Elsevier Inc. America, 2004

[7] B. Gavish, Topological design of Telecommunication Networks-local Access Design Methods", Annals of Operations Research, vol.33(1), p.17–71, 1991

[8] S. Chamberland, B. Sanso, O. Marcotte, "Topological Design of Two–Level Telecommunication Networks with Modular Switches", Operations Research, vol.48(5), p.745–760, 2000

[9] A. Girard, B. Sanso, L. Dadjo, "A Tabu Search Algorithm for Access Network Design", Annals of Operations Research, vol.106(1–4), p.229–262, 2001

[10] I. Godor, G. Magyar, "Cost-optimal Topology Planning of Hierarchical Access Network", Computers & Operations Research, vol.32, p.59–86, 2005

[11] R. Zhao, S. Goetze, R. Lehnert, "A Visual Planning Tool for Hybrid Fiber-VDSL Access Networks with Heuristic Algorithms", The 5th International Workshop on Design of Reliable Communication Networks (DRCN 2005), p.541–548, Island of Ischia (Naples), Italy, 2005

[12] M.R. Garey, D.S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-completeness", San Francisco, CA, Freeman, 1979

[13] S. Chopra, "Polyhedra of the Equivalent Subgraph Problem and Some Edge Connectivity Problems", SIAM Journal on Discrete Mathematics, vol.5, p.321–337, 1992

[14] T.L. Magnanti, S. Raghavan, “Strong Formulations for Network Design Problems with Connectivity Requirements”, Networks, vol.45(2), p.61–79, 2005

[15] T.L. Magnanti, S. Raghavan, “Redesigning network topology with technology considerations”, International Journal of Network Management, vol.18(1), p.1–13, 2008

[16] M. Weiss, F. Zeyer, “Redesign of Local Area Networks Using Similarity-based Adaption”, Proc. 10th Conference on AI for Applications, p.284–290, San Antonia, TX, USA, 1994

[17] W.D. Grover, “Mesh–Based Survivable Networks”, Prentice Hall, New Jersey, 2004

[18] R.H. Jan, F.J. Hwang, S.T. Chen, “Topological Optimization of A Communication Network Subject to A Reliablity Constraint”, IEEE Transactions on Reliability, Vol.42, p.63–70, 1993

[19] S. Rai, D.P. Agrawal, “Distributed Computing Network Reliability”, IEEE Computer Society Press Tutorial, USA, 1990

[20] T.G. Robertazzi, “Planning Telecommunication Networks”, IEEE Press, USA, 1999

[21] L. Nederlof, K. Struyue, C. Shea, H. Misser, Y. Du, B. Tamayo, “End-to-End Survivable Broadband Networks”, IEEE Commun. Mag., vol.9, p.63-70, 1995

[22] Y. Liu, D. Tipper, P. Siripongwutikorn, “Approximating Optimal Spare Capacity Allocation by Successive Survivable Routing”, IEEE/ACM Transaction on Networking, vol. 13, No. 1, p.198-211, February 2005

[23] R. Diestel, “Graph Theory”, 2.ed, vol. 173, Graduate Textbooks in Mathematics, Spring-Verlag, 2000

[24] G. N. Fredericson, J. Jaja, “Approximation Algorithms for Several Graph Augmentation Problems”, SIAM Journal on Computing, vol.10(2), p.270–283, 1981

[25] S. Khuller, R. Thurimella, “Approximation Algorithms for Graph Augmentation”, Journal of Algorithms, vol.14(2), p.214–225, 1993

[26] S. Khuller, B. Raghavachari, A. Zhu, “A Uniform Framework for Approximating Weighted Connectivity Problems”, Proc. 10th Annual SODA (ACM-SIAM), p.937–938, Baltimore (short paper), 1991

[27] I.D. Ljubic, J.J. Kratica, “A Genetic Algorithm for the Biconnectivity Augmentation Problem”, Proceedings of the 2000 Parallel Problem Solving from Nature VI Conference, vol.1917, p.641–650, 2000

[28] K.-D. Hackbarth, A. Menèndez, C. Diaz, A.J. Portilla, “An approximative algorithm for fully biconnectivity and its application to network design”, Euro-NGI.–D.JRA 3.1.2, p.95–106, 2004

[29] T. Erlebach, “Ausfallsicherheit von Netzen (reliability of networks)”, Lecture manuscript, ETH Zürich, 2003

[30] S. Pracht, D. Hardman, W. Horneff, “Echotest in IP-Netzwerken (echo test in IP-networks)”, vol.1, p.44–46, 2001

[31] J. Evans and C. Filsfils, “Deploying Diffserv at the Network Edge for Tight SLAs (Part 1)”, IEEE Internet Computing”, vol.Jan, p.61–65, 2004

[32] L. Kleinrock, “Queueing Systems, Volume II: Computer Applications”, John Wiley & Sons, Canada, 1976.

[33] R. Zhao, H.J. Liu, R. Lehnert, “Topology Design of Hierarchical Hybrid Fiber-VDSL Access Networks with ACO”, The Fourth Advanced International Conference on Telecommunications (AICT2008), p.232–237, Athens, Greece, June, 2008

[34] R. Zhao, Y. Zhang, R. Lehnert, “Topology Design of Hierarchical Hybrid Fiber-VDSL Access Networks with Enhanced Discrete Binary PSO”, The Third International Conference on Access Networks (AccessNets08), Las Vegas, Nevada, USA, October, 2008

[35] K. Menger, “Zur Allgemeinen Kurventheorie”, Fundamenta Mathematicae, vol.10, p.96–115, 1927

[36] R. Sedgewick, “Algorithms”, 2.ed, Princeton University, Addison-Wesley, 1988

[37] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, “Network Flows: Theory, Algorithms, and Applications”, Prentice Hall, United States Ed edition, 1993

[38] D. Jungnickel, “Graphenm, Netzwerke und Algorithmen”, 3.ed, BI-Wissenschaftsverlag, Mannheim/Leipzig/Wien/Zuerich, 1994

[39] G. Chartrand, “Introductory Graph Theory”, p.45-49, New York, Dover, 1985.

[40] R.E. Tarjan, “Depth First Search and Linear Graph Algorithms”, SIAM Journal on Computing, vol.1, p.146–160, 1972

[41] M. Weinkopf, D. Reimer, “Entgeldantrag fuer digitale Standard- und Carrier-Festverbindungen, fuer den Comfort-Service (dSFV) und die Express-Entstoerung (CFV)”, Technical Report, 2004

[42] NetWorks, http://www.networks.detecon.com/en/, 2008

[43] R. Baessler, A. Deutsch, “Nachrichtennetze (Telecommunications Networks)”, VEB Verlag Technik Berlin, Berlin, 1989

[44] Germany National Research and Education Network, http://www.dfn.de/, 2008

[45] A. Gous, A. Afrakhteh, “Traffic Engineering through automated optimisation of routing metrics”, Case Study, Cariden Technologies, 2002