Budapest University of Technology and Economics
Department of Telecommunications and Media Informatics

Methods for Planning of Access Networks

István Góbor

High Speed Networks Laboratory
Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics

Ph.D. Dissertation

Advisors:
Dr. Gyula Sallai

Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics

Dr. Gábor Magyar

Traffic Analysis and Network Performance Laboratory
Ericsson Hungary Ltd.

Budapest, Hungary
2004
Nyilatkozat

Alulírott Góbor István kijelentem, hogy ezt a doktori értekezést magam készítettem és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

A dolgozat bírálatai, a tézisfüzet és a védésről készült jegyzőkönyv a BME Villamosmérnöki és Informatikai Kar Dékáni Hivatalában elérhető.


..............................

Góbor István
Kivonat

Jelen doktori értekezés a hozzáférési hálózatokban felmerülő topológia tervezési és optimalizálási, valamint forgalomirányítási problémák megoldásában elért eredményeket ismertet. A disszertáció célkitűzése a hozzáférési hálózatok áramkörirányításával és méretezésével kapcsolatos tervezési feladatok vizsgálata, olyan hatékony hálózattervezési eljárások és algoritmusok megalkotása, amelyek eredményesen használhatóak napjaink és a közeljövő hálózataiban.

Az első vizsgált terület a hierarchikus hozzáférési hálózatok tervezése, amely során egy olyan több fából álló topológiát kell meghatározni, amely a költségek szempontjából optimális. Az egységes fának eleget kell tennünk szintkorlátbeli, valamint akár szintenként változó fokszámokra korlátozott megkötéseknél is a feladat nemlineáris elemek, a forgalom szintenkénti összegződése, valamint az egységes hálózati elemek költségének egymára hatása. A feladat megoldásakor megkülönböztettem az interferenciára érzékeny elemeket és tartalmazó hálózatokat az ilyen elemeket nem tartalmazóktól. Az utóbbi eset megoldására egy új, klaszterezés, gráfalgoritmusok és lokális keresésen alapuló, kétlépcsős heurisztikus megoldást javasolt, amely során az első fázisban kapott megoldást szisztematikusan javított tovább. Az interferenciára érzékeny hálózatok tervezésének egyes lépései az érzékeny pont-pont és a pont-multipont elemek különleges tulajdonságait kidöntő új heurisztikát alkalmaztak, majd az így kapott érzékeny részhálózatokat a fenti megoldás felhasználásával egyesítették. Szimulációk segítségével megmutattak, hogy a javasolt tervezési módszerek alacsonyabb költségű hálózatokat eredményeznek, mint az irodalomban ismert egyéb eljárások.

A következő kutatási terület a súlyozott Fermat-probléma általános megoldása és alkalmazása topológiai javítások elérésére, azaz a nyomvonalt optimalizálása szerepelt, hogy a fából mely csomópontokat és milyen módon kötünk össze. Az optimalizálás során megengedjük az ún. Fermat-pontok megjelenését, amelyek segítségével a csomópontok közötti összekötötettségek hossza és kapacitása is csökkenthető. Analitikus vizsgálatokra alapozva egyszerű megoldást adtak a Fermat-pontok helyének leírására. Továbbá bemutattak a probléma és forgalom multiplexálás témaköröknél összekapcsolását, valamint meghatároztak az ily módon elérhető javítás mértékét.

Az utolsóként vizsgált terület az IP (Internet Protocol) alapokon működő hálózatokban alkalmazott Open Shortest Path First (OSPF) útvonal-választó protokoll esetén használt adminisztratív súlyok optimalizálása hozzáférési hálózatokban. Esetében a cél egyfajta inverz legrövidebb út feladatként olyan első súlyok keresése, amelyek alapján az OSPF-alapú OSPF-útvonal-irányítás egy előre megadott üzemi és védelmi útvonalszer megfelelő útvajt találja a legrövidebbnek. Szimulációk segítségével megmutatták, hogy a javasolt súlybeállító eljárás hatékonyabb, mint a jelenleg alkalmazott megoldások, valamint életszerű esetekben tökéletes megoldást ad.
Abstract

This dissertation presents some results in the field of topology planning, optimization and routing optimization problems in access networks. The objective of the dissertation is to analyze these routing, dimensioning and topology planning problems and to create effective network planning methods and algorithms that can be efficiently applied in the networks of our time and the immediate future.

The first investigated area is the planning of hierarchical access networks, where the resulting network topology should be a set of trees with minimal cost. The difficulty of the task lies in the cascading constraint (depth of the trees) and in the degree constraint, which can be different even for each level of the hierarchy. Additional factors also increase the complexity of the planning problem. There are non-linear components of the cost-structure; the traffic is aggregated level by level in the hierarchy; and the costs of the network elements have an effect on each other. In the solution, I differentiate between the networks containing interference-sensitive elements and those networks, which do not contain such elements. For the latter case, I propose a new two-phased heuristic method based on clustering, graph algorithms and local search. This method systematically improves the initial solution given by its first phase. During the planning of the interference-sensitive networks, I have applied a new heuristic method, which exploits the special properties of the sensitive point-to-point and point-to-multipoint network elements. This method applies the above-mentioned solution to merge the sensitive parts of the network. Based on simulations, I showed that the proposed planning methods provide lower cost networks than the existing solutions in the literature.

The next investigated research area is the solution of the weighted Fermat-problem and its applications to topology optimization, that is the trace of the network can be optimized according to how and which nodes are connected to each other in the tree. By the application of the so-called Fermat-points, the length and the capacity of the connections can be decreased. Based on analytic investigations, I gave a compact and descriptive formula to determine the position of the Fermat-points. Furthermore, I elaborated the way how the weighted Fermat-problem can be merged into the topic of traffic multiplexing as well as I analyzed how much improvement can be reached with this technique.

The last investigated research area is the optimization of the administrative weights applied by the Open Shortest Path First (OSPF) routing protocol, which is widely used in IP-based (Internet Protocol) networks. In our case, the planning task can be formulated as an inverse shortest path problem. Here the goal is to find such weights, which provide that the OSPF-based routing finds the proper paths of an a priori given default and backup path system as the shortest paths based on the weights of the links. Based on simulations, I showed that the proposed weight setting method is more efficient than the existing solutions, moreover, this method gives perfect solutions in practical cases.
In memory of
my grandpa, Gegi

this work is dedicated to

my daughter, Virág & my son, Dávid
and
my loving wife, Gabi.
Contents

List of Abbreviations ii
List of Figures ii
List of Tables iii
Acknowledgements v

1 Introduction 1
  1.1 Preface ........................................ 1
  1.2 State of the Art .................................. 3
  1.3 Network Model ................................... 8
    1.3.1 Network Architecture ......................... 8
    1.3.2 Constraints ................................ 10
    1.3.3 Cost Components ............................. 12

2 Cost-optimal Planning of Interference-insensitive Hierarchical Access Networks 14
  2.1 Introduction .................................... 14
  2.2 Problem Statement ............................... 15
    2.2.1 Input and Output ............................. 15
    2.2.2 Problem Formulation ......................... 16
  2.3 Proposed Algorithms ............................ 17
    2.3.1 Operations .................................. 17
    2.3.2 Algorithms for Initial Solution .............. 22
    2.3.3 Algorithms for Improvement Phase .......... 23
    2.3.4 Enhancements ................................ 25
  2.4 Performance Analysis ......................... 25
    2.4.1 Application Example ......................... 26
    2.4.2 Problem Instance Generation ................. 28
    2.4.3 Running Time ................................ 29
    2.4.4 Algorithm Comparison ....................... 30
2.4.5 Robustness and Generality ............................................ 32
2.5 Conclusions ......................................................... 34

3 Cost-optimal Planning of Interference-sensitive Hierarchical
Access Networks ......................................................... 36
3.1 Introduction ......................................................... 36
3.2 Problem Statement ................................................... 37
3.2.1 Input and Output ............................................... 37
3.2.2 Problem Formulation ........................................... 38
3.3 Proposed Algorithms ................................................. 38
3.4 Settlement of Point-to-point Connections .......................... 39
3.4.1 Algorithm for Microwave & Wireline Access Trees ........... 39
3.4.2 Interference-sensitive Microwave Part .......................... 40
3.4.3 Interference-insensitive Wireline Part .......................... 41
3.4.4 Postprocessing .................................................. 41
3.5 Combined Planning of Point-to-point and Point-to-multipoint
Connections ............................................................ 42
3.5.1 Algorithms DIST-TREE, DIST-CLU and COST-CLU ...... 42
3.5.2 Initial Hub Positions and Base Station Allocation Strategies
.......................................................... 43
3.5.3 Deploying Point-to-multipoint Sectors ......................... 44
3.5.4 Local Optimization .............................................. 45
3.6 Performance Analysis .................................................. 45
3.6.1 Lower Bound Approximation of SPC ......................... 45
3.6.2 Numerical Results of SPC ...................................... 46
3.6.3 Graphical Illustration of SPC ................................... 47
3.6.4 Problem Instance Generation of CPPMC ...................... 47
3.6.5 Numerical Results of CPPMC .................................. 50
3.7 Conclusions ......................................................... 51

4 Solution of the Weighted Fermat-problem and its Applications to
Topology Optimization .................................................. 53
4.1 Introduction ......................................................... 53
4.2 A General Solution to the Weighted Fermat-problem ............. 54
4.2.1 The Weighted Fermat-point and the Angle-technique ....... 55
4.2.2 Calculation of the Fermat-angles .............................. 57
4.2.3 Geometrical Properties of Some Special Cases ............... 59
4.3 Application of the Weighted Fermat-point to Topology Optimization
.......................................................... 60
4.3.1 Connection between the Weighted Fermat-problem and the
Traffic Multiplexing Gain ............................................. 61
4.3.2 Applicability Questions of the Angle-technique ................ 61
4.3.3 Analysis of the Gain of the Topology Optimization .......... 64
4.4 Conclusions ......................................................... 68
5 Optimization of OSPF Administrative Weights in Access Networks

5.1 Introduction .................................................. 69
5.2 Problem Statement ........................................... 70
  5.2.1 Input and Output ........................................... 71
  5.2.2 Problem Formulation .................................... 71
5.3 Proposed Algorithm ......................................... 72
  5.3.1 Initialization .............................................. 73
  5.3.2 Iterative weight adjustment ......................... 76
5.4 Reference Algorithms ....................................... 78
  5.4.1 Heuristic Reference Algorithms ..................... 79
  5.4.2 The Exact LP Solution ................................. 80
5.5 Performance Analysis ....................................... 81
  5.5.1 Comparison of the Proposed and the Exact Algorithms ... 82
  5.5.2 Comparison of the Proposed and the Heuristic Reference Algorithms ..................... 82
5.6 Conclusions .................................................... 84

6 Summary ......................................................... 85
  6.1 Scientific Results ........................................... 85
  6.2 Application of the Results ............................... 86
  6.3 Short Outlook ............................................... 87

References ......................................................... 88

Appendix to Chapter 2: Pseudo Codes of Planning Procedures 93
List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATM</td>
<td>Asynchronous Transfer Mode</td>
</tr>
<tr>
<td>CDMA</td>
<td>Code Division Multiple Access</td>
</tr>
<tr>
<td>CPPMC</td>
<td>Combined Planning of Point-to-point and Point-to-multipoint Connections</td>
</tr>
<tr>
<td>DSL</td>
<td>Digital Subscriber Line</td>
</tr>
<tr>
<td>ECMP</td>
<td>Equal-Cost MultiPath</td>
</tr>
<tr>
<td>EDGE</td>
<td>Enhanced Data rates for GSM Evolution</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communications</td>
</tr>
<tr>
<td>ILP</td>
<td>Integer Linear Programming</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>LOS</td>
<td>Line of Sight</td>
</tr>
<tr>
<td>LP</td>
<td>Linear Programming</td>
</tr>
<tr>
<td>MWAT</td>
<td>Microwave &amp; Wireline Access Trees</td>
</tr>
<tr>
<td>OSPF</td>
<td>Open Shortest Path First</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>QP</td>
<td>Quadratic Programming</td>
</tr>
<tr>
<td>PMP</td>
<td>Point-to-multipoint</td>
</tr>
<tr>
<td>PTP</td>
<td>Point-to-point</td>
</tr>
<tr>
<td>RBS</td>
<td>Radio Base Station</td>
</tr>
<tr>
<td>RNC</td>
<td>Radio Network Controller</td>
</tr>
<tr>
<td>SPC</td>
<td>Settlement of Point-to-point Connections</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunications System</td>
</tr>
<tr>
<td>UTRAN</td>
<td>UMTS Terrestrial Radio Access Network</td>
</tr>
</tbody>
</table>
# List of Figures

1.1 Sample network with two aggregation trees. The numbers stand for the traffic of the nodes and links. ........................................... 9
1.2 An illustration of the network model in the case of point-to-multipoint connections. .............................................................. 10
1.3 An illustration of the most important constraints ......................... 11

2.1 The framework of the algorithm ............................................ 18
2.2 The relations between the operations ...................................... 19
2.3 The illustration of operation Recluster ................................... 21
2.4 Comparison of running times .............................................. 29
2.5 Algorithm comparison (instances with 500 nodes) ..................... 31
2.6 Comparison of network solutions with 400 nodes ..................... 33

3.1 Comparison of different networks with 1 length ......................... 47
3.2 Solutions for instances generated by Methods 1 and 2 .................. 48
3.3 Comparison of the deployment methods for the two instance types . 50
3.4 Comparison of the main algorithms for the two instance types ....... 51

4.1 Application of Fermat-problem .......................................... 53
4.2 Fermat-angles .................................................................. 56
4.3 Connection between the sides of $\triangle ABC$ and $|PC|$ ................. 66
4.4 Connection between $\gamma$ and $G$ with different $M$ values .......... 68

5.1 The framework of OWSA .................................................. 72
5.2 Directed circle of backup links ............................................ 74
5.3 Directed circle with default paths merging at the same node ....... 75
5.4 Directed circle with default paths merging at different nodes ...... 76
5.5 Special case: special weight setting .................................... 78
5.6 Overload in the network .................................................. 83
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>RNC, RBS and HUB costs</td>
<td>27</td>
</tr>
<tr>
<td>2.2</td>
<td>Link costs</td>
<td>28</td>
</tr>
<tr>
<td>2.3</td>
<td>Average solution quality of different approaches and instance sizes, (\sqrt{=}) best</td>
<td>31</td>
</tr>
<tr>
<td>2.4</td>
<td>Equipment and link costs</td>
<td>32</td>
</tr>
<tr>
<td>3.1</td>
<td>The quality of the solutions for different networks compared to the lower bound</td>
<td>46</td>
</tr>
<tr>
<td>3.2</td>
<td>Computation table for equipment costs</td>
<td>49</td>
</tr>
<tr>
<td>3.3</td>
<td>Comparison of the running times of the main algorithms</td>
<td>51</td>
</tr>
<tr>
<td>4.1</td>
<td>Precision of the evaluation of (G)</td>
<td>67</td>
</tr>
<tr>
<td>5.1</td>
<td>Average running times at 40% protected nodes</td>
<td>82</td>
</tr>
<tr>
<td>5.2</td>
<td>Overload in the network (optimal solution without overload = (\sqrt{=}))</td>
<td>84</td>
</tr>
<tr>
<td>6.1</td>
<td>Parameter-table of procedure Improvement</td>
<td>101</td>
</tr>
</tbody>
</table>
Acknowledgements

First of all, I would like to thank my advisor Dr. Gyula Sallai for his continuous professional leadership.

I would like to thank Dr. Miklós Boda, former head of Research and Development Unit at Ericsson and Dr. Hans Eriksson, former head of Traffic Analysis and Network Performance Laboratory at Ericsson for their continuous support and encouragement. I would also like to thank Dr. Tamás Henk, head of High Speed Networks Laboratory for his support and valuable comments.

I would like to thank Dr. Gábor Magyar for his valuable support and encouragement as well as for his useful ideas and advices. I would like to thank all my colleagues I have worked together with at Ericsson Traffic Laboratory and High Speed Networks Laboratory, especially János Harmatos, Alpár Jüttner, Dr. László Hévizi, Attila Szlovecskák, Dr. Áron Szentesi, István Maricza, Dr. Zoltán Király and Dr. Tibor Cinkler.

I would like to express my thank to my wife, my daughter, my son, my parents, my brother and my grandparents as well as to the whole family for their continuous support and encouragement.
Chapter 1

Introduction

1.1 Preface

The wider sense considered telecommunications networks were progressively established in the last decades. First they covered only smaller segments of the population, then the networks providing different services (telephone and cable TV network) gradually covered the majority of the population. Because of the gradual enlargement, more emphasis was on the planning of the backbone, the trunk or the core networks (in what follows, all are referred to as core networks), while the planning of the access and the last mile networks played a secondary role. Later the networks became larger and more complex, the core and the access network sharply separated from each other.

Furthermore, there are also significant changes in the purpose of the access networks. The role of the traditional telephone networks is changed because of the DSL technology settled over them. In addition to the cable TV networks, the telephone networks are the main medium of the Internet applications. The cable TV networks give opportunity to use virtual videotheques (e.g. video on demand) as well as other streaming and telemedical services (e-medicine). The evolution of mobile networks from GSM through GPRS and EDGE towards UMTS (3G) involves the enlargement of the data traffic in the networks.

The performance demands of the networks are also changed. In addition to the cost-optimal planning, the robustness and the fault tolerance of the networks are more and more important factors. These requirements may have a great influence on the structure of the networks.

The UMTS networks [1] can be considered as an appropriate example for the planning of future mobile networks (e.g. [2, 3]) consisting of hundreds or thousands of nodes. The dimension of the networks indicates that the networks are built in a hierarchical way and the traffic in the network is aggregated in concentrator nodes placed in each level of the hierarchy. As the high quality access to the networks is
very important (e.g. small delay), therefore the number of the hierarchy levels is limited.

In order to be cost-effective, the access networks usually have tree topology, where the connections between the nodes consist of wireline (or insensitive to interference in general) or microwave systems. The application of microwave systems is motivated by the fast and "cheap" settlement. In this case, however, there are stricter constraints on the network structure in order to avoid the disturbing interference between the microwave systems. The planning of the interference-insensitive networks is discoursed in Chapter 2 and some results concerning the interference-sensitive network are presented in Chapter 3.

The trace of the network can be optimized according to how and which nodes are connected to each other in the tree. In this case, the length and the capacity of the connections can be optimized between three (weighted Fermat-problem) or more (Weber-problem) nodes. By the application of the so-called Fermat-points, the cost of the trace can be decreased (these points are the solution of the Fermat-Weber-problem). In Chapter 4, this problem is analyzed and new theoretical results are shown.

Since the basic structure of the access networks is usually a tree, therefore the failure of an equipment may cause considerable outage in the network. The fault-tolerant topologies (e.g. ring or mesh) could be a solution to the above problem, but they may greatly increase the cost of the network. A possible alternative solution is to keep the basic tree topology and expand it with some additional connections (in what follows referred to as links) at those parts of the network, where faults are critical. The main advantage of the network extension is that an acceptable equilibrium can be found between the increment of the cost and the increment of the fault tolerance capability of the network.

It is a general tendency that data traffic is becoming dominant and the Internet Protocol (IP) is more and more widely used. Where the operation of the network is IP-based, the most commonly used routing protocol recently is the Open Shortest Path First (OSPF) protocol [4]. The OSPF is a link-state dependent routing protocol and uses so-called administrative link weights as metrics in the routing process. Finding the adequate setting of these weights is the most important part of the routing in order to obtain suitable network performance (e.g. maximal utilization, maximization of free capacities). In Chapter 5, some planning questions and solutions from the area of inverse shortest path problems are presented.

In spite of the fact that the algorithms presented in the rest of the dissertation were developed to solve a particular planning task in a given network (see Chapter 6.2), my aim was to work out such algorithms, which can be applied without any major modification to similar planning tasks of other networks. The reached technology-neutrality lies in the constraints, which are handled as parameters inside broad bounds. Beyond that, the handling of the cost-functions as a
"black-box" also serves the neutrality. Further details about networks, constraints and cost-functions are presented in Section 1.3.

The chapters of the dissertation introduce a complete planning process from the (even greenfield) topology planning (Chapter 2 based on [C1, J1, J4] and Chapter 3 based on [C3, C4]) through the trace optimization (Chapter 4 based on [J3]) up to and including the configuration of the network according to the OSPF routing protocol (Chapter 5 based on [C2, J2, J5]). Finally, the dissertation is closed by a summary (Chapter 6).

There are papers in the literature that also deal with the planning of access networks, but they analyze the planning tasks from other aspects, handle relaxed constraints and cost-functions, or treat smaller sized networks. A short survey of some relevant papers is presented in Section 1.2.

1.2 State of the Art

First existing methods related to the planning hierarchical access networks are surveyed, then previous solutions connected to the Fermat-problem are shown, finally the literature of the OSPF weight setting in focus are outlined.

Hierarchical Access Network Planning

The hierarchical access network planning problem outlined in Section 1.1 is a combinatorial optimization problem belonging to hierarchical capacitated facility location and it is computationally hard, as even its significantly simplified versions are proved to be NP-hard [5, 6, 7, 8]. Relevant papers from the literature deal with substantial simplifications of our problem. These simplifications include several of the followings: i) they have a simple hierarchy with only one level of concentrators; ii) they build one tree with a predetermined central node instead of a forest; iii) they have no degree or depth constraints; iv) they have no distinguished constraints for interference-sensitive links; v) they cannot handle point-to-point and point-to-multipoint connections at the same time; vi) they plan the tree levels totally independently; vii) they have no concentrator costs; viii) they have fixed concentrator costs; ix) their link or concentrator costs do not depend on the transferred capacity; x) their cost functions are linear; xi) the transferred capacities are not aggregated through the hierarchy when the costs are computed.

The concentrator location problem and its hierarchical variants are widely studied (see e.g. [9, 10, 11, 12, 13]). Similarly to our case, [12] addresses a hierarchical network design problem, allows varying concentrator types according to cost and capacity, performs traffic aggregation, involves the aggregated traffic in the concentrator cost calculation, and applies a clustering [14, 15] framework for the solution. However, in [12], the proposed algorithm performs only one pass in the upward
direction when selecting the concentrators, therefore the positioning of the concentrators is done totally independently for each network level. Differently from our model, [12] deals with designing only one tree with predetermined central node instead of a forest and performs this without constraints on the tree depth and on the number of incoming links to a concentrator. Furthermore, [12] does not apply link costs and the concentrator costs and capacities are fixed by level, i.e., all concentrators on a given tree level have to be the same type in terms of cost and capacity.

In [10], the authors expand the above model to enable arbitrary concentrator types on the same network level with keeping the aggregation of the demands. They give a Lagrangian relaxation based heuristic solution. It is restricted to work with a predefined central node and it does not handle degree and depth constraints. In addition, its application to problem instances of practical sizes is rather limited due to the computational complexity and the huge amount of decision variables.

Survey papers [16, 17] address relevant network design and location problems. Both of them review tributary/access network design algorithms. Star-star networks work with a predefined central node, which means that the models are restricted to optimize only two levels: one level of concentrators and allocating the lowest level terminals to them. In tree-star (and tree-tree) networks the concentrators can be located in different levels of the hierarchy, these models support the differentiation of concentrators (e.g. by network level, capacity, degree). However, the algorithms discussed in the surveys do not handle varying link and concentrator costs at the same time and do not support demand aggregation. Another shortcoming of these algorithms is that they work with one predefined central node only at the top level. Section 3 of [17] deals with centralized tree-like networks, i.e., capacitated minimum spanning tree (CMST; Section 3.1) and two-level hierarchical networks (Section 3.2). The presented algorithms handle degree and depth constraints, however, in CMST the aim is only to minimize the link cost.

In [18], the author deals with hierarchical telecommunication access network design, but restricts himself to star-star networks. In [7], the author presents a solution to star-tree problems, where only one level of concentrators should be determined and the trees below the concentrators are built with a CMST algorithm. Both [18] and [7] work with a predefined central node.

Several location theory papers [19, 20, 21, 22, 23, 24, 25] deal with linear cost functions but hierarchical facility location. In [21], the authors deal with a tree-tree network. First they determine the position of the concentrators with solving an uncapacitated facility location problem. Then they independently create access trees with keeping the computed concentrators fixed as the root of the trees. Their algorithm handles several trees, but it does not involve intermediate concentrator costs, uses fixed top-level concentrator costs and linear link costs, and does not handle degree and depth constraints. The algorithm in [25] is capable for plan-
ning several access trees and it is based upon a generalization of the Lagrangian relaxation for the p-median problem. It applies a depth constraint in the network, however, the objective is focused only on the link costs, the demand is not aggregated through the hierarchy, and degree constraints are not handled.

Some existing practical approaches [26, C1, 27, 28] in the field of telecommunication access network topology planning use a generally applicable local search metaheuristic [29], such as simulated annealing (SA) [30], which is often combined with greedy heuristic subphases. Basically these approaches can deal with any kind of cost function. However, it is typically hard to define perturbation operators with a suitable neighborhood for the SA method in the case of hierarchical network design. More importantly, the computational requirements of these heuristics limit applications to problem instances with only a few hundred network nodes. A simulated annealing heuristic is presented in [28] for a hierarchical network design problem with capacitated hubs and three-level trees, working with a predetermined central node. Varying concentrator and link costs are used in the model of [27], and a level-by-level algorithm similar to the one in [12] is proposed. However, this algorithm allocates the nodes to the concentrators based on the distance instead of the real cost, and it computes the required capacities along the links and in the concentrators only in the final stage. The combination of the algorithms in [26] and [C1] gives a solution to a problem similar to the presented one’s. However, these algorithms perform only one pass (top-down) in the hierarchy and the quality of their solution is worse than the presented one’s. (See Section 2.4 for details.)

In [31], the applied network model is very close to ours. The authors propose an algorithm for hierarchical network planning, however, their algorithm is strongly connected to a three-level model. They propose an improvement strategy, where the algorithm starts from a star topology and refines it to obtain a tree topology. During the optimization, the nodes are moved only between adjacent levels of the hierarchy, which considerably limits the efficiency of the algorithm in case of networks with more levels of hierarchy. Furthermore, they neglect the equipment costs in the lowest level(s) of the hierarchy and the complexity (speed vs. efficiency) of the algorithm is based on the problem size instead of changing adaptively. In order to utilize the benefits of their work, I have extended their algorithm to efficiently handle any number of hierarchy levels. (See Section 2.3, 2.3.1 and 2.3.3 for details.) Some other papers [32, 33, 34] also deal with similar constraints as our model. The combination of the algorithms presented in [33, 34] first determines the number and position of the central nodes, which nodes will be fixed as the roots of the trees. Then access trees below the central nodes are created for each tree separately. The main shortcoming of this approach is that it computes the central nodes by approximating the access trees with stars and keeps the separated access trees fixed in the later stages. Furthermore, the running time is rather large due to the complexity of the Lagrangian relaxation method [33, 34].
The above cited papers illustrate that there are various kinds of algorithms that can be applied for network planning, however, distinguished degree constraints for interference-sensitive links have not been applied. Although as a rule of thumb [35], an equipment dependent angle separation between the microwave links of a node helps keeping the interference under tolerated limits.

Moreover, the problem of facility location with sectorized hubs (hubs with point-to-multipoint connections besides the "traditional" point-to-point links) has not been addressed in the form as it is presented in Chapter 3.

The planning problem in the focus of Chapter 3 can also be recast as a special set-covering problem [36] in case of "uniform" cost structure. Unfortunately the costs of the nodes in a sector depend on the order of their choosing if more point-to-multipoint sectors are in the hub or these sectors overlap. Thus this case cannot be handled by set-covering methods.

**Fermat-problem**

Local optimization techniques are frequently applied in case of topology planning, where the local optimization means the improvement of the quality of the link structure between the nodes and many times it is restricted to small parts of the entire network.

In [37], the authors present heuristics to determine which links can be merged at extra nodes (Fermat-points) in order to save cost. However, their model handles only distance-based linear cost functions, therefore, they cannot utilize the advantage of traffic multiplexing at all. Moreover, in a linear model, the capacities of the links also cannot be taken into consideration.

There are other exact and heuristic algorithms in the literature to find the Fermat-points, however, the general formulation of the solutions are rather complex if a formulation exists at all.

In case of merging two links, the planning task belongs to the weighted Fermat-problem, which was thoroughly analyzed in [38, 39, 40, 41]. The authors give a geometrical construction method to find the Fermat points by applying mostly the so-called weight triangles and the Simpson lines (see Section 4.2.1 for details). However, they propose no closed formula to describe these points "independently" of the actual problem instance.

In case of merging several links, the planning task belongs to the weighted Weber-problem. A simple iterative heuristic solution to this problem was proposed in [42] more than 65 years ago, however, exact solution to this problem has not been found since then. "Only" heuristic solutions arisen (e.g. [43, 44]), which accelerate the convergence in the Fermat-Weber-problem. In [43], the authors multiplied the predetermined step size of the Weiszfeld algorithm by a factor and they found that the total number of iterations to meet a given stopping criterion is reduced.
substantially by the new step size. In [44], the author propose a Newton-type acceleration to the Weiszfeld algorithm with fast global and local convergence. Her algorithm is suitable for large-scale Euclidean location problems and parallel implementation.

The problem of deciding in advance whether or not the application of such Fermat points results in cost saving and how much the gain will be is an open question.

**OSPF Weight Setting**

Compared to the above two scientific area, the inverse shortest path problem is the newest research area.

The weight setting algorithms usually aim at abolishing the overload of the links and at maximizing the free capacities (e.g., [45]). The formulation of the problem as an inverse shortest path problem is just a way to find the appropriate weighting. Many aspects of the problem are proved to be NP-complete (see e.g., [46]), so solutions based on linear programming [47, 48, 49] are not proposed to dynamic or large-scale problem instances, where heuristic solutions can be effective enough.

In [50], the authors propose a solution to the case, when there are several origin-destination pairs that we want to be optimal under some costs. The authors formulate the inverse shortest path problem as a convex Quadratic Programming (QP) problem, which can be solved by a dual QP method [51]. This methodology is efficient in case of one path system at a time, however, the solution cannot be applied to provide the same weighting for several different path systems. In [52], the authors expand that problem to the case, where there can be one path between each pair of nodes and we search for a weighting that makes all of them shortest paths simultaneously. In this case, it is possible that no weighting can satisfy the requirements. They had also analyzed a more complex case, when we search for a weighting that makes the prescribed path unique minimum weight paths between their endpoints.

In the survey [53], the problem is classified as Shortest Path Tree problem, where the aim is to find exactly one path from a root node to all other nodes. It is known [54] that the shortest path tree problem can be formulated as a minimum cost flow problem, which can be solved by linear programming, which, however, limits the applicability of this approach to relatively smaller network sizes. Furthermore, the resulting weights are valid only for one given "tree" configuration at a time.

The above mentioned formulation of the problem belongs to the network flow problems [54, 55]. Although, the traditional multicommodity flow formulation can be reduced to the minimum cost flow problem, an important shortcoming of the network flow representation still exists. This shortcoming is that the network flow approach aims to find the weight system together with the optimal path system.
Thus there is no guarantee that the network flow representation provide a path system, which is equal to a predefined one.

According to another approach, we can solve the problem indirectly. In that case the main issue is the settlement of an OSPF link weight system in order to achieve near-optimal network throughput for the assumed demand pattern or availability of network resources. In [56], two heuristics (a greedy type and simulated annealing based [57, 58]) are proposed. In both cases, they increase the weights of overloaded links and decrease the weights of underloaded links to achieve the highest network utilization. Since their model is the closest to mine, the comparison of their approach and my solution is presented in Section 5.5.

To sum up, the most important properties of the majority of the existing results are that the optimization is done for one case at a time and they cannot be applied to solve the problem, when we are looking for a uniform weighting satisfying the requirement of more cases.

1.3 Network Model

First the network architecture is presented. Then the constraints are discussed, which have to be taken into consideration during the network planning. Finally general cost components are introduced, which are applied by the proposed algorithms.

1.3.1 Network Architecture

Figure 1.1 gives an illustration of the network model that is used in the case of access network planning. The network is modeled as an undirected graph $G(V, E)$, where vertex set $V$ represents the nodes of the network, while edge set $E$ corresponds to the links. The network topology is a set of spanning trees completed with some additional links (backup links) to provide the pre-defined level of network availability [C2, J2]. As the figure shows, two groups of links are distinguished. In the first group, there are the default links, which can be further divided. They can be insensitive or sensitive to interference disturbances. The former are illustrated by solid bold lines and the latter are illustrated by dashed lines. In the other group, there are the backup links illustrated by dotted bold lines. The nodes are depicted by triangles and the roots are depicted by squares.

In each tree, the traffic flows through concentrator nodes upwards to the root node through transmission links. The concentrator nodes are selected from the nodes, therefore the concentrators have their own traffic demands as well. The root of each tree is a special controller that aggregates the traffic of lower level nodes and forwards it to the core network.
Each node $v \in V$ is associated with its own traffic $\tau_v \in \mathcal{T}$ to be aggregated [59]. Note that the traffic can be a complex data, the only assumption is that the addition operator is defined for it. (For example, this data can be the different bandwidth requirement of different traffic classes.)

In case of the OSPF weight setting problem, the network is handled tree by tree, so then $G(V, E)$ corresponds to a given tree, where the root of the tree is denoted by $r \in V$.

The access network is divided into two logical parts. The first part involves wireline links corresponding to the interference-insensitivity, the second part contains microwave links corresponding to the interference-sensitivity. The wireline links are situated close to the root, while the microwave connections are placed nearer to the leaves of the trees [C3] (since smaller capacity demands can be easily satisfied with microwave systems). In the lower level of the hierarchy, there can be hubs, which can communicate with the nodes connected to it via point-to-point (PTP) and point-to-multipoint (PMP) connections as well [C4]. A hub may contain several PTP links and PMP sectors. The microwave links in the network can be of either PTP or PMP, see Figure 1.2. In that case, the hubs are a part of a tree-star structure, where the hubs are in the middle of the stars with a PTP and a PMP layer.

Reverting to the subject of the network availability, the set of default links is denoted by $E_d$ and the set of backup links is denoted by $E_b$ [J5].

The tree $E_d$ determines the level $l_v$ of each node $v$, i.e. the distance of $v$ and $r$ on this tree.

It is required that exactly one default link and at most one backup link can be originated from each node. Moreover, the backup links can connect only two nodes on the same level or on adjoining levels upwards in order to provide ac-
ceptable network delay in case of single failures (for more information about the technological background see [12]).

We are also given a set of the protected failure scenarios as a set of edge sets $E^f \subseteq E$, \( f \in \{0, 1, 2, \ldots, F\} \) expressing the available edges in the $f$th failure scenario. $E^0$ corresponds to the case when there is no link failure, i.e. $E^0 = E_d$.

For each node $v$ and for each failure scenario $f$ we are given a path $p_v^f \subseteq E^f$. This path is used to carry data from $v$ to $r$ in case of the $f$th failure scenario. The path $p_v^0$ is called default path the others are the backup paths. If the node is not protected against a failure, then $p_v^f = \emptyset$. It is also required to use the default path whenever it is possible, i.e. if $p_v^0 \subseteq E^f$, then $p_v^f = p_v^0$.

A positive integer number $w(e)$ is assigned for each link $e \in E$ as the OSPF weight. We can similarly define the weight of a path: $W(p_v^f) = \sum_{e \in p_v^f} w(e)$.

Only one link failure is considered at a time (which is relevant to the generally accepted network modeling). For the sake of simple configuration, it is required that each backup path contains only one backup link and some default links. This ensures that the backup paths will not be too long.

1.3.2 Constraints

Figure 1.3 illustrates the most important constraints for the trees of the access network from the following ones, where $m$ and $w$ indices denote microwave and wireline, respectively.

1. Cascading (level) constraint denoted by $L_m$ and $L_w$: the maximal number of hops (i.e. the distance) between a node and the root. $L_m \geq 1$ and $L_w \geq 1$.
are upper bounds for the level, which depends on the type of the outgoing link of the node; the roots are in level 0.

2. Degree constraint denoted by $D_w^l \geq 1$ \( l \in \{0, 1, \cdots, L_w\} \): specifies a maximum for the number of incoming and outgoing wireline links of a node. This constraint can vary for different tree levels and depends on the maximal capacity of node equipment installed in the concentration nodes. As the lowest level leaf nodes have no incoming links, $D_w^L = 1$.

3. Minimal pointing angle separation between microwave links denoted by $S_{\text{min}}$: as a rule of thumb [35], there is an equipment dependent minimal angle by which two microwave links of the same node must be separated in order to keep the interference under the tolerated limits. This is a rough way of taking interference into account. In practice, however, sophisticated models are used for evaluating a given network configuration, e.g., in the frequency- and polarity planning phase, but not in the topology planning phase.

4. Maximal pointing angle separation denoted by $S_{\text{max}}$: as a second rule of thumb, there is a complementary angle of $S_{\text{min}}$, stating that the included angle of any two microwave links of the same node must be lower than $S_{\text{max}}$. Hereby, we can avoid that two microwave links look opposite to each other.

5. Maximal length of PTP microwave links. (In practice, a similar constraint for wireline links is hardly ever used.)

6. Coverage range of PMP sectors [km].

7. Beam width of sector antennas providing the PMP sectors (e.g. 90°).
8. Degree of overlapping of the PMP sectors (fully, partially, etc.).

9. Capacity limit for links and nodes.

10. Line of sight (LOS) parameter for all node pairs.

### 1.3.3 Cost Components

The cost components are related to the links and the equipments installed in the nodes of the network. The particular calculation of the cost are detailed in the corresponding chapters, so now only the common properties of the cost components are described.

I note that the proposed algorithms can handle any type of cost structure, since the cost function is considered as a black box function. Hereby operational, site and other kinds of costs also can be taken into account. Moreover, in order to provide general applicability for the algorithms, it is important to handle non-linear, discontinuous cost functions.

If a given device cannot satisfy the technical constraints, then the corresponding cost function returns $\infty$, since the goal of the network planning is to give a solution with minimal cost.

#### Node Equipment Cost

The cost of a node equipment usually depends on the followings.

- **a)** The cost of "ports" belonging to the given node. This is a step-wise function depending on the number and type of other nodes connected to the node in question (including the type of the connection: PTP or PMP, wireline or microwave).

- **b)** The capacity dependent cost of ports. This is also a step-wise function, and describes the cost of the required capacity of the current port.

- **c)** The device cost represents the installation and investment cost of required sub-devices, like processors, boards, PMP sector antennas, etc.

The input function $C_{\text{equip}}(v, l_v, t_v)$ returns the cost of the equipment to be installed into node $v \in V$ if it is placed at level $l_v$ in the tree, and if it aggregates $t_v \in T$ amount of traffic.

#### Link Cost

The link cost usually has a capacity-independent and a capacity-dependent part. The former is proportional to the distance of the two endpoints. The latter is
typically described by a step-wise or a piece-wise function representing the required bandwidth. Of course, both components depend on the type of the link.

In case of microwave links, the dependence of their cost on their length is practically negligible. Instead, they have a maximum range, which is typically smaller for PMP links than for PTP links.

Let us suppose that the cost of the interference-sensitive microwave links is less than the cost of the interference-insensitive wireline links if they can be deployed at a particular node pair. Since the constraints on the wireline links are always weaker in practice, therefore, if a microwave link is more expensive than a "similar" wireline, then microwave links would not be used at all.

The input function $C_{\text{link}}(u, v, t_{uv})$ returns the cost of a link between the nodes $u, v \in V, u \neq v$ if it carries $t_{uv} \in T$ aggregated traffic.

**Total Cost**

The cost of the access network topology is calculated as a sum of link cost and the sum of node equipment cost:

$$C_{\text{total}} = \sum_{(u, v) \in E} C_{\text{link}}(u, v, t_{uv}) + \sum_{v \in V} C_{\text{equip}}(v, t_v, t_v) \quad (1.1)$$
Chapter 2

Cost-optimal Planning of Interference-insensitive Hierarchical Access Networks

2.1 Introduction

This chapter deals with cost-optimal topology planning of hierarchical telecommunication access networks, in which the individual elements of the network have no disturbing influence on the operation of each other, i.e. the network itself is insensitive to interference. The main planning task is to aggregate the user traffic in a multi-level tree-like fashion by the use of intermediate concentrator nodes in a cost-optimal way. Note that the problem statement is general, therefore the proposed algorithms can be applied to several related problems in the areas of facility location, hierarchical location-allocation, etc.

Cost-optimal planning of access networks is a critical issue in designing today’s communications networks, especially in the case of mobile networks (for example, GSM or UMTS, see Section 2.4.1). The typically smaller link capacities in an access network are more sensitive to improper dimensioning than the capacities of large core network links; precise planning of links is crucial in order to meet the quality of service requirements. In addition, access network represents a major part of the total network cost, considering only their sheer size. The size and complexity of mobile access networks is becoming enormous making manual planning a difficult and time-consuming task. The solution obtained by algorithmic optimization is superior to that of manual planning. Quick algorithmic optimization allows the evaluation and comparison of different deployment strategies; this is not possible in case of manual network planning within the very short timeframes. A very important requirement of the optimization procedure is technology-neutrality, as technology can change very rapidly in the telecommunication environment. Cost-
based access network optimization algorithms play a central role in network planning software tools; their quality and efficiency have an important influence on the usefulness of the whole software package.

The problem is to plan an access network, which consists of a number of trees. There are several constraints on these access trees making the problem NP-hard. Therefore, we do not expect to find the globally optimal solution, but we rely on heuristic algorithms. The main idea of the presented approach is to combine clustering and local optimization operators and to iterate the combined operators within the levels of the hierarchy.

The chapter is organized as follows. Section 2.2 defines the optimization problem. Section 2.3 describes the proposed algorithms, with the detailed pseudo codes in the Appendix. Section 2.4 demonstrates the generality, robustness and efficiency of the proposed algorithms through an extensive empirical comparison. (Section 2.4.1 presents an application example from the telecommunications sector; this environment will be used when presenting computational results.) Section 2.5 contains the conclusions and outlines further open questions.

2.2 Problem Statement

The access network is modeled by a graph and it consists of aggregation trees that collect the traffic demands of base station nodes. The aggregation trees have a specified depth limit and there are certain degree constraints for intermediate concentrator nodes in each tree level. In each aggregation tree, the traffic flows through special concentrator nodes upwards to the root node through transmission links. The concentrator nodes are selected from the nodes, therefore the concentrators have their own traffic demands as well. For further details, please see Section 1.3.

2.2.1 Input and Output

The input consists of the following factors.

- The nodes of the network with their location and traffic ($r_v$).
- The cascading (level) constraint ($L_w$).
- The degree constraint for each level ($D_w^l, l \in \{0, 1, \ldots, L_w\}$).
- The cost function of the equipment ($C_{\text{equip}}(v, l_v, t_v)$).
- The cost function of a link ($C_{\text{link}}(u, v, t_{uv})$).

The output contains the links between the nodes, where the decision variable $x_{uv} \in \{0, 1\}$ equals to 1 iff node $u$ is connected upwards to node $v$ in the aggregation
tree containing it, \( \forall u, v \in V \). For each node \( v \in V \), the variable \( 0 \leq l_v \leq L_w \) gives the level of that node in the tree containing it, where \( l_v = 0 \) for a root node, \( l_v = 1 \) for a child node of a root, etc. The total aggregated traffic of each node \( v \in V \) is given by \( l_v \in T \).

A set of auxiliary variables are derived from the above ones: \( l_{vk} \in \{0, 1\} \) equals to 1 iff node \( v \) is at level \( k \) in the tree containing it.

### 2.2.2 Problem Formulation

The access network planning problem is then:

\[
\min \left( \sum_{(u,v) \in E} C_{\text{link}}(u, v, t_{uv}) + \sum_{v \in V} C_{\text{equip}}(v, l_v, t_v) \right) \tag{2.1}
\]

with respect to

\[
l_{vk} = \begin{cases} 
1 & \text{if } l_v = k \\
0 & \text{if } l_v \neq k
\end{cases} \quad \forall v \in V, k \in \{0, 1, \ldots, L_w\} \tag{2.2a}
\]

\[
t_v = \tau_v + \sum_{u \in V, u \neq v} x_{uv} \cdot t_u \quad \forall v \in V \tag{2.2b}
\]

\[
\sum_{v \in V, v \neq u} x_{uv} = 1 - l_{u0} \quad \forall u \in \mathcal{N} \tag{2.2c}
\]

\[
\sum_{u \in V, u \neq v} x_{uv} \leq \sum_{k=0}^{L_w} l_{vk} \cdot (D^k_w - 1 + l_{u0}) \quad \forall v \in V \tag{2.2d}
\]

\[
x_{uv} = 1 \Rightarrow l_u = l_v + 1 \quad \forall u, v \in V \tag{2.2e}
\]

The objective function (2.1) returns the total cost of a configuration, summing up the cost of the installed links and the installed equipment in the nodes. Expression (2.2a) introduces the set of auxiliary variables \( l_{vk} \). Constraints (2.2b) derive the aggregated traffic for each node, i.e., for nodes that have no incoming links, the aggregated traffic equals to the traffic given as input for the node, otherwise the aggregated traffic is calculated by summing the child nodes’ aggregated traffic together with the given input traffic for the node. The traffic can be any kind of complex data. The problem statement is technology-neutral, as the aggregated traffic appear only in the link and equipment cost functions, which are flexible inputs. Equation (2.2c) requires that for each node except a root node (i.e., \( l_{u0} = 1 \)), the outdegree must be 1, meaning that every non-root node is connected exactly to one parent node upwards. Inequalities (2.2d) specify that the degree constraints \( (D^k_w) \) for the number of connected child nodes must be satisfied for each node depending on its current level. Finally, Constraint (2.2e) ensures that only nodes of
adjacent levels can be connected to each other. By this constraint, loops are not possible; together with Equation (2.2c), it ensures that the traffic for each node will reach a root level node in a tree.

2.3 Proposed Algorithms

I propose a two-phase heuristic planning algorithm (partially based on the idea presented in [31]). In the first phase, an initial network solution is constructed and in the second phase, it is improved.

The initial solution fulfills every constraint and limitation, therefore it can be used as it is. It is enough to give us an overall information about how many nodes should be in certain hierarchy levels and let the improvement phase find better interconnections between the nodes. A new algorithm called Top-Down is proposed for constructing an initial solution.

Once we have the initial solution, we continue with the powerful and much more time-consuming improvement phase, see algorithm Full-Iterate. The proposed approach finds low cost connections and suitable hierarchy level for each node. The improvement steps are iterated until no further improvements are found. This phase is solved by an enhanced variant of the algorithm presented in [31]. The enhancement includes new and improved operations (which are the building-blocks of the algorithm) and a new conception to control the complexity and effectiveness of the operations.

To describe the proposed algorithm, basic operations (or operators) have to be defined. The initial solution is built with these basic operations, which also constitute the basis of compound operations. The compound operations are used in the improvement phase. The complexity of the compound operations is adjusted adaptively, where the adjustment is based on the actual performance of the algorithm. Such a way, the algorithm is flexible and depending on the requirements we can offer solutions faster or better solutions relatively slower. The framework of the algorithm is shown in Figure 2.1.

The operations are discussed in Section 2.3.1. The algorithms for initial solution are described in Section 2.3.2 and the improvement algorithm is given in Section 2.3.3. Finally, Section 2.3.4 shortly deals with some built-in enhancement issues, which speed up the computation. The pseudo codes of the algorithms are given in the Appendix.

2.3.1 Operations

There are four basic operations and three compound operations. The basic operations are: Cluster, Tree, Moves and Swaps. The compound operations are:
Recluster, ReclusterLevel and LocalOpt. The definitions of the operations are given below. The relation between the operations is shown in Figure 2.2.

**Basic Operations**

**Cluster.** This operation divides a given input set of nodes \( (P) \) into a number of distinct subsets \( G_i \) (called clusters) \( (i = 1, \ldots, K) \), which represent the best configuration. The centers of the clusters will be on level \( l \), which is an input parameter of the operation. The cost of a configuration is derived from (i) connecting all nodes in each cluster to the corresponding cluster center put in level \( l \) (from now called the median) and (ii) installing equipment in the median nodes, then (iii) connecting these median nodes upwards to higher level. The clustering is based on the well-known \( K\)-means algorithm [14]. To find the best value of \( K \), we repeat the \( K\)-means algorithm for multiple values of \( K \) between \( K_{\text{min}} \) and \( K_{\text{max}} \). The upper bound is \( K_{\text{max}} = |P| - F \), where \( F \) is the number of nodes in \( P \) that are forbidden to be in level \( l \). The lower bound is \( K_{\text{min}} = \left\lceil \frac{|P|}{U_l} \right\rceil \), where \( U_l \) is the maximal number of nodes that can be connected to a median in level \( l \) (\( U_l \) is calculated from the depth and degree constraints). We do not try out all possible values of \( K \) between \( K_{\text{min}} \) and \( K_{\text{max}} \). We start from \( K = K_{\text{min}} \). Then we continue
with $K \leftarrow K + 1$ in the next round. We stop to advance $K$ if we did not get better solution in the last $K_{la}$ rounds, where $K_{la}$ is a look-ahead parameter set to a small value.

Operation Cluster involves three modular elements, which depend on the context in which the clustering is applied: the allocation step (for creating clusters by allocating nodes to selected medians), the relocation step (for choosing a new median in a cluster) and the cost-calculation step (for the evaluation of a network configuration). Since operation Cluster is the basis of compound operation Recluste and the Top-Down algorithm, the above three modular elements will be defined in both procedures separately.

**Tree.** The purpose of this operation is to build up a tree topology from a given input set of nodes, which topology satisfies the level and degree constraints as well as the fixed and forbidden node constraints. The inputs of the operation are a node set $P$, the root $r \in P$ of the tree and desired hierarchy level $l$ of this root. The operation uses two sets of nodes, the set of yet unconnected nodes and the set of nodes that has already been connected to the tree. The latter ones serve as possible parent nodes for the unconnected ones. Initially, only the root $r$ considered as parent. We connect the unconnected nodes one by one to extend the tree, where we choose the least cost node-parent pair in each step. The operation is accelerated by considering only a small random sample from the unconnected node set in each step. Moreover, if we find that the tree could not be extended at a particular parent node after several attempts (because of constraint-, equipment- or link type violation), then we discourage ourselves to try this parent node again.
and we remove it from the set of possible parents. Operation **Tree** is used for quick and greedy cost-approximation of particular tree sections in the *Top-Down* algorithm. When operation **Cluster** have created clusters in level \( l \), we create trees from these star-like clusters to evaluate the actual solution. Operation **Tree** is also used in the **TreePlan** algorithm for creating trees under the already planned parts.

**Moves.** This operation performs local optimization on a candidate solution in order to improve it. Operation **Moves** detaches a node from its original parent node and connects it to a new parent node. The move is accepted if it yields a lower solution cost. Two subsets of \( V \) are the input for this operation. The first set \( C \) includes the candidate nodes that can be moved, while the second one \( P \) contains the parent nodes to be examined as target nodes for a move. We try all candidate-parent pairs for possible moves while they improve the current solution. This operation is used by compound operations **Recluster** and **LocalOpt**.

**Swaps.** This operation also performs local optimization on a given solution. Here the elementary step interchanges (swaps) the parent nodes of two given candidate nodes. We accept a swap only if it results lower cost. The input of **Swaps** is a subset \( C \subseteq V \) containing the candidate nodes that are allowed to be swapped. We try all possible pairs of candidate nodes for swapping while they improve the current solution. This operation is used by compound operations **Recluster** and **LocalOpt**.

**Compound Operations**

**Recluster.** This operation recalculates the number and positions of the concentrators in a selected part of the network and integrates operation **Cluster** with local optimization operations **Moves** and **Swaps**. It has three input parameters: a node \( r \), which defines the subsection of the solution to be reorganized, the type of **allocation step** within **Cluster** and the frequency of local optimization steps. Operation **Recluster** works as follows (see Figure 2.3):

1. Consider the child nodes of \( r \) in level \( l \) and their children in level \( l + 1 \).
2. Select new nodes to be concentrators in level \( l \) from the considered nodes and connect them to \( r \).
3. Connect the remaining nodes to the new concentrators.

There are two options for **allocation step** within **Cluster**. In the first case, we pick a random node and connect it to the median that provides the least cost; then repeat this until all nodes have been allocated to a median. In the second case, we
choose the least-cost connection between a node and a median from all possible node-median pairs. Then we repeat this until all nodes have been allocated. At the relocation step, we try out all nodes in each cluster as a candidate for new median and the one providing the least cost will be chosen as the new median in each cluster, separately. The cost-calculation step is based on the real network costs and it is performed only in the affected parts of the network. Operation Cluster is extended with local optimization steps. We have four options to control the frequency of performing Moves and Swaps within the clustering: (i) no local optimization at all, (ii) we apply Moves and Swaps only at the end of Cluster, (iii) we apply local optimization for each value of $K$ and (iv) we apply local optimization after each allocation and relocation step.

**ReclusterLevel.** This operation iteratively performs operation Recluster in order to find better connections and concentrator positions in an entire level of the hierarchy. We go through the nodes of the given level ($l$) sequentially and choose these nodes as parameter $r$ for Recluster. By this, the concentrators in level $l+1$ will be reorganized in the entire network. We can consider operation Recluster-Level as a horizontal optimization in the hierarchy. This operation is applied in the improvement phase to recompute the concentrators at each level. We start with the reconstruction of level 0, then we continue with levels $1, \ldots, L_w - 1$, by iterating vertically.

**LocalOpt.** This operation performs a neighborhood search by applying operations Moves and Swaps. The nodes of a complete hierarchy level are chosen as the input node set for Moves and Swaps. We can restrict operation LocalOpt to optimize the connections only between two adjacent levels of the network, getting a
level optimization. We can allow *Moves* and *Swaps* between two nodes disregarding their level to achieve a thorough optimization. There is an option to restrict the application of *Moves* and *Swaps* to particular aggregation trees separately. Operation *LocalOpt* is used between the clustering steps of the improvement phase and it helps to *migrate* nodes between clusters.

### 2.3.2 Algorithms for Initial Solution

In this section, three methods to construct initial solutions are introduced. First, the proposed algorithm is presented, called *Top-Down*, which is built on operations *Cluster* and *Tree*. Then an initial solution is given, called *TreePlan*, which adapts two existing approaches based on simulated annealing (TreePlan and TreeMake algorithms, see [26, C1]). Finally, a very fast initial solution, *Random* is given as reference. This algorithm may give an expensive solution because this is a randomized-greedy method without any sophisticated optimization step. It was developed to analyze the performance of the improvement phase.

**Top-Down.** This top-down clustering algorithm first determines where and how many nodes should be in level 0 in the network. It applies operation *Cluster* with special modular elements. Then it selects some nodes to be at level 1, and so on, till it reaches the lowest level. The framework is the following:

1. Apply operation *Cluster* to divide the network into distinct aggregation tree areas. The medians of the resulting clusters will be the root concentrator nodes of the evolving trees.

2. Perform operation *Cluster* separately for each cluster formed in Step 1. This way, within each tree, we get clusters and median nodes that will serve as first-level concentrators. We connect the new medians to the corresponding parent median nodes (namely to the corresponding root concentrators). Thus we have created the first two levels of the hierarchy (level 0 and 1).

3. In the same way, compute concentrators in levels $l = 2, \ldots, L_w - 1$.

In operation *Cluster*, we compose the total cost of the network from two factors. The first factor is the cost of the already planned parts. The second factor is the cost of the yet unknown remaining part, which consists of the most recently determined clusters. In order to give a realistic approximation of the second factor, we construct greedy trees from the nodes of these clusters by operation *Tree* and temporarily extend the already planned parts with the new trees. The second factor will be the cost of these temporary parts.

In operation *Cluster*, we apply the first type of allocation step and the relocation step presented in operation *Recluster*. However, the optimization is done on a geometrical basis because the exact aggregated traffic values of the nodes are unknown.
at this point (these values are known only in the parts already planned).

**TreePlan.** The original *TreePlan* [26] algorithm determines the number and position of nodes that will be at level 0 and assigns all other nodes to them to form trees. Then method *TreeMake* [C1] determines the nodes to be in level 1, ..., \( L_{w} \), separately for each tree created by *TreePlan*. Both algorithms select nodes to be in the considered level \( l \) by a simulated annealing technique, then they build up the rest of the network under level \( l \) by applying operation *Tree*. During the simulated annealing process, the actual node set in level \( l \) is modified by selecting a new node to be in level \( l \) or by removing one from there or by applying both at the same time. The SA process is used to decide whether a modification results in a better network configuration or not. The decision is based on the cost as in the *Top-Down* algorithm.

**Random.** First we select a node randomly and put it into the first level of the hierarchy (level 0). Then, iteratively, we pick a node randomly and connect it to an already connected node (nodes in level 0 are considered as connected). The choice of a parent node for the connection is random. If a new connection conflicts with the limitations or implies infinite cost, then we discourage ourselves to try this parent node again and we do not consider this parent as connected node further on. If no new connection can be established, we put a node randomly to the first level to form a new root.

### 2.3.3 Algorithms for Improvement Phase

In the improvement phase, the connections and the hierarchy level for each node are systematically revised. Operation *ReclusterLevel* and *LocalOpt* are applied to optimize one level at a time. In each round of the algorithm, we step down in the hierarchy. At the end of a round, operation *LocalOpt* is also applied to allow modifications also to nodes situated in non-adjacent levels. The rounds are iterated until no further improvements are found. During the iteration, the complexity of the *compound operations* is adaptively advanced.

The proposed algorithm called *Full-Iteration* is presented by specifying one round of the improvement phase:

1. *(New round)* Improve the actual solution by going through the hierarchy level by level. To achieve this, there is a loop with \( l = 0, \ldots, L_{w} - 1 \), and:
   
   (a) *(Cluster)* Apply *ReclusterLevel* to recompute the number and position of concentrators in hierarchy level \( l \).
   
   (b) *(Level Optimization)* Apply *LocalOpt* to optimize the connections between level \( l \) and \( l + 1 \).
2. *(Thorough Optimization)* Apply LocalOpt with all the nodes independently from their levels in order to further improve the connections.

3. *(Evaluation)* Calculate the network cost and evaluate the solution:

   (a) *(Advance complexity)* If the improvement of the costs between the actual and the previous round is not satisfactory, then advance the complexity of the compound operations if it is possible and go to Step 1.

   (b) *(Termination)* If the solution improved, then go to Step 1 else stop.

The complexity of operations Recluster and LocalOpt controls the actual choice of the modular steps used in these operations. (Note that operation Recluster is the core of ReclusterLevel.) For operation Recluster the modular steps are: the allocation step and the involved local optimization steps. For operation LocalOpt the complexity is controlled by the option of applying this operation tree-by-tree separately or on the entire network.

The advance of the complexity allows us to find an equilibrium between the running time of the algorithm and the quality of the solution. There are 9 degrees of the complexity in the improvement phase:

- Apply LocalOpt tree-by-tree and
  - apply the first type of allocation step in Recluster. Try out all options for local optimization in Recluster (1,2,3,4).
  - apply the second type of allocation step in Recluster. Try out option (ii), (iii) and (iv) for local optimization in Recluster (5,6,7).

- Apply the second type of allocation step and apply option (iv) for local optimization in Recluster. Apply LocalOpt on the entire network (8).

- Same as above (8), but skip the local optimization acceleration, see Neighborhood lists in Section 2.3.4 (9).

We can differentiate four variations of our proposed improvement strategy. All variations depend on the way how operation ReclusterLevel and LocalOpt are used. If we allow to apply both operations, we get Full-Iteration as defined above. The second variation skips Steps 1b and 2 and it applies only ReclusterLevel. This variation is called Clu-Iteration. The third variation skips Step 1a and it applies only LocalOpt. This variation is called Loc-Iteration. In the fourth variation, Clu-Iteration and Loc-Iteration are applied alternately, this variation is called Alt-Iteration.
2.3.4 Enhancements

Recomputing the concentrators within a selected subsection of the solution is computationally much more favorable than treating the node set as a whole and recalculating the concentrators for the entire node set at the same time. In addition, the following features speed up the above operations and algorithms.

**Neighborhood lists.** Operations *Moves* and *Swaps* use special neighborhood lists for checking the geometrical distance of two points. These sorted lists are computed for each node at the beginning of the process. They contain only nodes that are close enough to the particular node. The neighborhood of a node is a small part of all nodes in the network. The final neighborhood list is of size List = max(q \cdot n, List_{min}), where q << 1 and List_{min} << n are parameters. To check whether a node is on the list of neighboring nodes is done in constant time. By using these lists in Step 3 of *Moves* (see Appendix), the positions of node u and candidate parent v are examined. We check whether v is in the neighborhood of u or in the neighborhood of the original parent of u. If it is, we go on with the next test. Otherwise this trial move is skipped and we go to Step 4 of *Moves*. Similarly, in Step 3 of *Swaps*, node u and v are examined. We check whether v is in the neighborhood of u or the parent of v is in the neighborhood of the parent of u and vice versa.

**Calculating cost-difference.** In operation *Moves* and *Swaps*, it is not required to compute the cost of the entire network every time when we check the benefit of the move or swap operation. It is enough to examine only the cost of the modified parts. In case of operation *Moves*, the cost-difference in Step 3c is calculated as shown in the Appendix. In case of operation *Swap*, the cost-difference is calculated similarly as in *Moves*, the only difference is that we work with two node-parent pairs instead of one (see the Steps 1-3 of this calculation).

2.4 Performance Analysis

This section summarizes the results of the empirical tests. All the tests of Section 2.4.3 and 2.4.4 were performed for instances belonging to the problem class described at the application example (see Section 2.4.1). After the description of the problem class, the test instance generation method is given. Then the results concerning the running time are presented. Afterwards the quality of the solution is examined, which was obtained by different algorithm variants. Finally, the flexibility of the approach is demonstrated.
2.4.1 Application Example

In this section, an example is presented by specifying the constraints and the applied cost model from the telecommunications sector. These constraints and costs will be used in Section 2.4 for generating sample numerical results of the proposed algorithms.

The *UMTS Terrestrial Radio Access Network (UTRAN)* [2, 3] is in the focus of the example. The user traffic is collected to the *Radio Base Stations (RBSs)* from the mobile equipment. The geographical location \((X_v, Y_v)\) of each RBS together with its estimated traffic \(\tau_v\) are determined by radio network planning, see e.g. [59]. These RBS locations and associated traffic values form the input nodes. Those special RBSs that are able to concentrate and forward the traffic of other, lower level RBSs, are called *HUBs*.

The *Radio Network Controllers (RNCs)* control and manage the communication connections and convey the traffic to the core network; these are the root nodes in our model. Beyond the total aggregated traffic, the cost of an RNC also depends on the total number of RBSs controlled by it. Based on the problem definition presented in Section 2.2, the HUBs and the RNCs are considered as concentrator nodes.

In the cost-optimal hierarchical access network planning problem, the task is to find a set of aggregation trees, where each tree is controlled by an RNC at the root, all the RBSs are connected to an RNC via HUBs or directly, and the total cost of the resulting configuration is minimal. (A simple network structure is shown in Figure 1.1.) The RNC and HUB nodes are selected from the RBSs and they are given extended functionality. However, the optimal number of RNCs and HUBs are not given. The traffic can have a complex daily profile, but for the sake of simplicity, the input traffic of each RBS is modeled by a constant \(\tau_v \in \mathbb{R} (\forall v \in V)\). To summarize, the input and the constraints are as follows:

- **Nodes**: RBSs with their geographical location \((X_v, Y_v), \forall v \in V\)
- **Traffic**: \(\tau_v \in \mathbb{R}, \forall v \in V\)
- **The maximum level is** \(L_w = 4\),
- **The maximum degrees for the concentrator nodes at each level**:
  \(D_w^0 = 200\) for the RNCs, and \(D_w^l = 50, l = 1, \ldots, L_w - 1\) for the HUBs

Next the applied cost functions are described. The *cost model* is artificial, but it models the typical characteristics and complexity of the real situation. There are three "base cost" parameters, for link, RBS/HUB and RNC costs. The base cost multiplied with a traffic-dependent factor gives the actual cost of the particular equipment or link. The base costs are:
\* \(BC_{\text{link}} = 1\), \(BC_{\text{RNC}} = 1000\) and \(BC_{\text{RBS/HUB}} = 5\).

Two types of equipment costs \(C_{\text{equip}}(v, t_v, t_v)\) are differentiated depending on \(v, t_v, t_v\): one for RNCs and another for HUBs and ordinary RBSs. The first is the \(C_{\text{RNC}}(v, t_v)\), which depends on the number of signal processors needed \((P_v)\) to control the attached RBSs and manage the traffic. The number of needed signal processors is approximated by a linear function of the number of RBSs \((n_v)\) in the tree rooted at RNC \(r\) and the total aggregated traffic \(t_v\): \(P_v = L(n_v, t_v)\). In the model, five different types of RNC equipment are involved with varying number of signal processors. The RNC cost depends upon the needed number of signal processors.

The second type of equipment cost is the \(C_{\text{RBS/HUB}}(v, t_v)\), standing for single RBSs and HUBs, which cost depends on the (aggregated) traffic passing through the node. In the model, there are five RBS/HUB equipment types. Table 2.1 shows both equipment costs:

<table>
<thead>
<tr>
<th>(C_{\text{RNC}}(v, t_v)):</th>
<th>(C_{\text{RBS/HUB}}(v, t_v)):</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1 \cdot BC_{\text{RNC}}) if (P_v \leq 100)</td>
<td>(1 \cdot BC_{\text{RBS/HUB}}) if (t_v \leq 5)</td>
</tr>
<tr>
<td>(2 \cdot BC_{\text{RNC}}) if (100 &lt; P_v \leq 200)</td>
<td>(2 \cdot BC_{\text{RBS/HUB}}) if (5 &lt; t_v \leq 15)</td>
</tr>
<tr>
<td>(3 \cdot BC_{\text{RNC}}) if (200 &lt; P_v \leq 400)</td>
<td>(3 \cdot BC_{\text{RBS/HUB}}) if (15 &lt; t_v \leq 30)</td>
</tr>
<tr>
<td>(4 \cdot BC_{\text{RNC}}) if (400 &lt; P_v \leq 700)</td>
<td>(4 \cdot BC_{\text{RBS/HUB}}) if (30 &lt; t_v \leq 60)</td>
</tr>
<tr>
<td>(5 \cdot BC_{\text{RNC}}) if (700 &lt; P_v \leq 1000)</td>
<td>(5 \cdot BC_{\text{RBS/HUB}}) if (60 &lt; t_v \leq 120)</td>
</tr>
<tr>
<td>(\infty) otherwise</td>
<td>(\infty) otherwise</td>
</tr>
</tbody>
</table>

Table 2.1: RNC, RBS and HUB costs

To sum up the equipment costs we get the following:

\[
C_{\text{equip}}(v, t_v, t_v) = \begin{cases} 
C_{\text{RNC}}(v, t_v) & \text{if } t_v = 0 \\
C_{\text{RBS/HUB}}(v, t_v) & \text{if } t_v > 0 
\end{cases} \tag{2.3}
\]

Finally, the link cost \(C_{\text{link}}(u, v, t_{uv})\) depends on the length of the link and on the traffic that it carries. The length \(d_{uv}\) of a link is the physical distance between the two endpoints \(u, v\) of the link, and ten different link types are used depending on the bandwidth of the link; see Table 2.2:

\[
d_{uv} = \sqrt{(X_u - X_v)^2 + (Y_u - Y_v)^2} \tag{2.4}
\]

It is straightforward to take existing equipment into consideration in the equipment cost function, i.e., for certain nodes the cost of equipment in that node can be forced to be less expensive if there is some previously deployed equipment. For instance, nodes from a GSM network, etc. Similarly, one can take existing links into account to get a less expensive network configuration. There can be different
Table 9.2: Link costs

<table>
<thead>
<tr>
<th>$C_{\text{link}}(u, v, t_u)$:</th>
<th>6 · $d_{uv} \cdot BC_{\text{link}}$ if 34 &lt; $t_u$ ≤ 36</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 · $d_{uv} \cdot BC_{\text{link}}$ if $t_u$ ≤ 2</td>
<td>7 · $d_{uv} \cdot BC_{\text{link}}$ if 36 &lt; $t_u$ ≤ 42</td>
</tr>
<tr>
<td>2 · $d_{uv} \cdot BC_{\text{link}}$ if 2 &lt; $t_u$ ≤ 8</td>
<td>8 · $d_{uv} \cdot BC_{\text{link}}$ if 42 &lt; $t_u$ ≤ 44</td>
</tr>
<tr>
<td>3 · $d_{uv} \cdot BC_{\text{link}}$ if 8 &lt; $t_u$ ≤ 10</td>
<td>9 · $d_{uv} \cdot BC_{\text{link}}$ if 44 &lt; $t_u$ ≤ 50</td>
</tr>
<tr>
<td>4 · $d_{uv} \cdot BC_{\text{link}}$ if 10 &lt; $t_u$ ≤ 16</td>
<td>10 · $d_{uv} \cdot BC_{\text{link}}$ if 50 &lt; $t_u$ ≤ 155</td>
</tr>
<tr>
<td>5 · $d_{uv} \cdot BC_{\text{link}}$ if 16 &lt; $t_u$ ≤ 34</td>
<td>$\infty$ otherwise</td>
</tr>
</tbody>
</table>

types of links available depending on the strategic decisions of the network operator. These can also be taken into account in the chosen link cost function. Other kind of exceptions such as a river between nodes $u$ and $v$ prohibiting the installation of a given type of link can also be embedded in the cost function. There can be previously fixed or forbidden equipment or links in the network to be planned. Fixed equipment means that a node should be placed in a given level (e.g. fixed at level 0 ⇒ there must be an RNC in that node). Forbidden equipment means the opposite. A fixed link means that we must take this link into consideration when we calculate a cost of a new link. A forbidden link means that the cost of a link between the given nodes is $\infty$. It is very important to model the real-world network planning situations by the use of existing, fixed or forbidden equipment/links and further exceptions. The proposed algorithms can handle and support all the above special features of the model.

2.4.2 Problem Instance Generation

The instances used for the testing are generated randomly.\footnote{All test instances are available upon request.} One of the main parameters controls the problem instance generation: the desired size of the instance, $n$. The method used for distributing $n$ nodes on the plane is the following. The nodes are placed into a square area sized according to $n$. The nodes are grouped to cities. The number of cities dropped to the area is selected according to the area size, population density and city size parameters. Afterwards the selected number of cities are generated. First, the city centers are positioned randomly to the area not too close to its borders. Then, the city nodes are positioned in a way that the probability of placing a node far from its city center is smaller than placing it nearer to it. The distribution of nodes is random, therefore smaller and larger cities will be present. The traffic for each node is a random value taken uniformly from the interval [1...2]. The device costs and general constraints are as described in Section 2.4.1. See Figure 2.6(a) for an illustration of a problem instance with 400 nodes and solved by the Top-Down initial solution with Full-Iteration method.
2.4.3 Running Time

The running times for variable problem sizes are presented here. Beyond examining the initial solution methods, the full algorithm combinations were tested, namely the Top-Down initial solution with all four kinds of improvement methods, and the other two kinds of initial solution methods, the Random and the TreePlan methods with Full-Iteration as the improvement algorithm. These together give six algorithm variants: Top-Down + Clu-Iteration, Top-Down + Loc-Iteration, Top-Down + All-Iteration, Top-Down + Full-Iteration, TreePlan + Full-Iteration, Random + Full-Iteration.

**Initial solution.** Figure 2.4(a) shows the running time results for the three initial solution methods. The Random method was so quick that it has finished much before 1 second in all cases regardless of $n$. The figure shows the average running time in seconds for different values of $n$: 10 random instances were averaged in each case. The Top-Down method was much quicker than TreePlan, and for 1000 nodes, it provided its result in less than 25 seconds. However, for larger instances, the Top-Down method also starts to be relatively slower due to the embedded computational complexity: for 2000 nodes - roughly 280 seconds, 3000 nodes - 2500 seconds, 4000 nodes - 28000 seconds.

![Running times of the initial solution methods](image1)

![Average running times of full algorithm variants](image2)

Figure 2.4: Comparison of running times

**Full algorithm variants.** Figure 2.4(b) shows the running time results for the six complete algorithms. Here 30 random instances were averaged for each value

---

2The presented algorithms are implemented in C++ and the tests were performed on a Sun Ultra Sparc 5 machine with 128 MB RAM.
of \( n \) (\( n = 100, 200, 400, 500, 600, 800, 1000 \)). The running times strongly depend on the termination condition of the improvement algorithm. In many cases during the last period of the algorithm the result improves only slightly (see Section 2.4.4 for monitoring of the solution quality versus time). Concerning the final running times, the methods can be divided into three groups. The ones that use only one of the two improvement features (Clu-Iteration and Loc-Iteration) are the quickest, while the Top-Down + Alt-Iteration and Top-Down + Full-Iteration methods are slower. The TreePlan and Random initial solution with the Full-Iteration improvement is the slowest, with nearly 20 minutes running time for \( n = 1000 \). For larger problem instances, it is important to carefully select the included clustering variants, the local optimization type and depth as too complex operations might not terminate in reasonable time.

2.4.4 Algorithm Comparison

Table 2.3 shows the results for the quality of the obtained solution for different values of \( n \), for the six full algorithm variants and the three initial solution methods. The results of 30 instances are averaged for each case. The mark (\( \checkmark \)) stands for the best result in each column. The numerical values express the difference in percentages from the best result of the column, e.g. value 7.72 in column \( n = 500 \) means that the Top-Down + Loc-Iteration algorithm provided a solution, which is 7.72\% worse on average than the best one that is obtained by the Top-Down + Full-Iteration algorithm (\( \checkmark \)). The last column of the table includes the averages of the rows. The Top-Down initial solution with the Full-Iteration or Alt-Iteration improvement method provided the best solutions, but their difference is not significant. The results of TreePlan + Full-Iteration and Random + Full-Iteration methods are a little bit worse. The Top-Down + Clu-Iteration and Top-Down + Loc-Iteration methods are relatively farther from the best result found, roughly 2\% and 8\%, respectively. It can be concluded that the best methods combine clustering and local optimization (Full-Iteration or Alt-Iteration variants), and the ones that use the Top-Down initial solution provide slightly better results. Adding the observation that these latter variants are also quicker, they seem to be a good choice. The results of the initial solution methods alone are modestly farther from the best result found, by roughly 15-20\%, except for the extremely far Random initial solution, which is much worse.

An illustrative investigation was performed with solving 50 random instances for \( n = 500 \), and tracking the improvement of the average solution quality over time, see Figure 2.5. The figure is divided into two parts along the time-axis. The first part shows the beginning 100 seconds, while the second part shows how the algorithms converge to their final solutions. In this latter part the cost-axis is focused to the range where the curves converge. Beyond 100 seconds running time, Top-Down initial solution with Full-Iteration and Alt-Iteration improvement meth-
Table 2.3: Average solution quality of different approaches and instance sizes, √=best solutions. In the beginning, Top-Down + Clu-Iteration without Loc-Iteration is the best, but after approximately 100 seconds we have to use it together with Loc-Iteration to be able to improve the solution. It makes almost no difference to choose Full-Iteration or All-Iteration to combine the clustering and local improvement methods. Algorithms using only Clu-Iteration or Loc-Iteration stop their operation quickly, and finally they lag behind the complex improvement methods. Random + Full-Iteration and TreePlan + Full-Iteration give slightly worse results than Top-Down + Full-Iteration, moreover, their results are always above the best ones.

![Graph 1](image1.png)

![Graph 2](image2.png)

Figure 2.5: Algorithm comparison (instances with 500 nodes)

To summarize, it is suggest to use Clu-Iteration improvement after the Top-Down initial solution to solve hierarchical access network planning problems if the running time is critical. Depending on the available time and the size of the instance, Clu-Iteration can be applied in itself or the Full-Iteration or All-Iteration improvement can be used to find even better solutions.
2.4.5 Robustness and Generality

The above tests showed that the proposed algorithm can successfully solve the presented network planning problem for the given costs and constraints. Note that the structure of the optimal network topology greatly depends on the ratio of the link and equipment costs. In order to examine the robustness and generality of the algorithm, namely how it works in different environments, the algorithm was tested for instances belonging to other problem classes. First a special continuous cost-function is defined in order to approximate the original one (see Section 2.4.1).

Next let us examine the effect of changing the number of levels in the trees and the maximal degree of the nodes in each level. Finally the case is considered when there was no distinguished root node in the equipment cost function. Four sample networks are shown in Figure 2.6 representing the solutions for the four different cases. These networks have the same input node set, only the connections between the nodes differ from each other, so one can easily recognize and compare the differences between them.

**Continuous cost.** The first special case is not to use discrete predefined equipment and links, but to apply continuous equipment and link cost functions that approximate the cost values of the original discrete case as follows:

<table>
<thead>
<tr>
<th>Equipment Cost</th>
<th>Link Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{RNC}(v, t_v) : BC_{RNC} \cdot \sqrt{P_v} / 6.5$ if $P_v \leq 1000$</td>
<td>$C_{RNC}(v, t_v) : BC_{RNC} \cdot \sqrt{P_v} / 6.5$ if $P_v \leq 1000$</td>
</tr>
<tr>
<td>$C_{RBS/HUB}(v, t_v) : BC_{RBS/HUB} \cdot \sqrt{P_v} / 2$ if $t_v \leq 120$</td>
<td>$C_{RBS/HUB}(v, t_v) : BC_{RBS/HUB} \cdot \sqrt{P_v} / 2$ if $t_v \leq 120$</td>
</tr>
<tr>
<td>$C_{link}(u, v, t_u) : BC_{link} \cdot d_{uv} \cdot \sqrt{t_u}$ if $t_u \leq 155$</td>
<td>$C_{link}(u, v, t_u) : BC_{link} \cdot d_{uv} \cdot \sqrt{t_u}$ if $t_u \leq 155$</td>
</tr>
</tbody>
</table>

Table 2.4: Equipment and link costs

It can be observed that the cost of the solutions found using continuous costs is always lower than in the original case. This meets the expectations as the type of a RBS/HUB, RNC, link cannot be a restrictive factor in this case. The shape of the trees converge to a minimal spanning tree, because it has the lowest cost that could be found. Figure 2.6(b) shows an instance for this case with 400 nodes. The quality of the different approaches for this problem class is practically equivalent to the results shown in Table 2.3.

**Modified constraints.** It is an interesting question how the algorithms react upon the modification of the two main constraints. Increasing the maximum number of levels in the tree and decreasing the maximum indegree of the nodes bring a significant change in the problem to be solved. Let $L_w = 10$ and $D_{wi} = 2$ for $1 \leq l \leq L_w$. In order to observe the impact of these constraints on the resulted network topology, the above continuous cost functions were used. The Top-Down algorithm with Full-Iteration and Alt-Iteration improvement methods gave the
best results. However, TreePlan + Full-Iteration got very close to them, but its running time was about four times larger than that of the Top-Down method. The Random + Full-Iteration method yielded a similar network solution, but it was one order of magnitude slower than the best ones. In this case the tree depth is larger,
see Figure 2.6(c).

**Equivalent nodes.** Finally let us assume that there is no distinguished root node. However, the equipment cost dependence from the occupied level is kept. Now the cost of the nodes depends on their aggregated traffic and the occupied tree level. The original RBS/HUB cost is simply divided by the actual level of the node and considered all nodes as an RBS/HUB. Now there is no different cost for the roots (RNCs). The maximum indegree of nodes in level 1 is decreased according to the other levels, i.e., $D_w^0 = 50$. It is expected that there would be much more first level nodes in the network resulting many smaller trees. The tests met the expectations, see Figure 2.6(d). In those areas where the density of the nodes is higher, larger trees evolved, while in sparse areas very small trees have been created. In the previous and this special case, applying the improvement algorithms is even more important, since without these methods the initial solutions, especially *TreePlan* and *Random*, are much less effective.

### 2.5 Conclusions

This chapter focused on cost-optimal hierarchical access network planning, where the network is insensitive to interference. This complex combinatorial optimization problem is defined in a generalized way. It involves arbitrary link cost structures, not only distance-based, but capacity-dependent cost components, as well. Moreover, equipment limitations are taken into account and the costs of equipment can also have an arbitrary structure. The problem has great significance in the mobile telecommunications industry.

Due to the complexity of the problem and the size of practical instances, a heuristic algorithm is proposed. The idea is to combine clustering and local optimization operators and to apply the combined operators iteratively within the levels of the hierarchy. For the individual operations, the degree of complexity is adaptively adjusted runtime in order to provide a solution that is the best one possible within the given time-frame. The method is configured for a trade-off between solution quality and running time; it can be used either to quickly obtain a low-quality solution (a quick estimate for strategic decisions) or to find a near-optimal solution during a relatively longer time.

The empirical analysis showed that the proposed algorithm is robust and it meets all special requirements of the problem statement. The method proved to be efficient for practical network planning purposes. The method can be used for extension planning of existing networks; any of the links and nodes can be fixed by the user at the beginning. The method is independent from the cost structure, therefore different planning conditions and strategies can be supported.

The approach is planned to be applied for practical network planning support.
especially for the challenging area of GSM to 3G/UMTS transition. A further research direction is to complete the presented approach with core network planning algorithms to provide an overall solution for the planning task.
Chapter 3

Cost-optimal Planning of Interference-sensitive Hierarchical Access Networks

3.1 Introduction

Compared to Chapter 2, further aspects of the planning of access networks are presented in this chapter. In this case, the individual elements of the network may have disturbing influence on the operation of each other. In the practice, the disturbing influence appears as interference between the wireless (microwave) network elements.

The concerned network planning problem also belongs to combinatorial optimization. There is a huge set containing the feasible configurations for the given problem. Each feasible configuration is associated with a cost calculated by pricing its individual elements. The aim of network planning is to find a feasible configuration that satisfies all technical constraints and has a relatively good (near-optimal) cost.

Since the network architecture is similar to the one presented in Chapter 2, only the main differences lying in the link types are pointed out.

Microwave links are frequently chosen to build new communication networks, since this type of equipment permits fast rollout, independence from other network infrastructure and low operational costs. These factors make microwave links especially advantageous for the access networks of cellular communication systems. However, the microwave links are sensitive to interference that can be avoid by handling some specific technical constraints.

Beside traditional point-to-point (PTP) microwave links, systems of point-to-multipoint type (PMP) have been available lately. In PMP systems, the traffic of a certain number of nodes is transported through a central radio node, which has
sector coverage, hereafter these nodes are called sectorized hubs. These hubs are specific concentrator nodes placed in the lowest levels of the hierarchy.

Dynamic resource allocation in PMP systems is a great advantage. In addition, PMP equipment can be more economic than an equivalent PTP implementation if a minimum number of nodes are connected to the hub. Sectorized hubs are, however, limited by a shorter range compared to PTP links. Using sectorized hubs and 'traditional' hubs together can lead to significant cost savings regarding the total cost of the access network.

The contribution of the study is threefold. First, it provides a method for interconnecting a set of base stations by microwave links with satisfying hard technical constraints. Second, it incorporates this method into a hierarchical network planning algorithm that provides wireline connections. Third, the presented solution mix PTP / PMP links optimally in the lowest levels of the network hierarchy ensuring the efficient use of sectorized hubs.

3.2 Problem Statement

The access network is modelled by a set of trees. The nodes of the trees are the radio base stations (RBS). The root of each tree is a special controller station (RNC) that aggregates the traffic of lower level base stations and forwards it to the core network. The access network can be divided into two logical parts. The first part involves the microwave links, the second part the wireline links. The wireline links are situated close to the RNCs, while the microwave connections are placed nearer to the leaves of the trees. For further details, please see Section 1.3.

3.2.1 Input and Output

The input consists of the following factors,

- The nodes of the network with their location, traffic \( \tau_w \) and line of sight visibility related information.
- The cascading (level) constraints \( L_w \) and \( L_m \).
- The degree constraint for each level \( D_w^l, l \in \{0, 1, \ldots, L_w\} \).
- The minimal pointing angle separation \( S_{\text{min}} \).
- The maximal pointing angle separation \( S_{\text{max}} \).
- The maximal length of PTP microwave links.
- The coverage range of PMP sectors.
• The beam width of sector antennas providing the PMP sectors.
• The maximal transfer through a PMP sector.
• The degree of overlapping of the PMP sectors.
• The cost function of the equipment ($C_{\text{equip}}(v, l_v, t_v)$).
• The cost function of a link ($C_{\text{link}}(u, v, t_{uv})$).

Note that the cost of a simple RBS is practically irrelevant to the optimization in case of the PMP sector planning, since it has influence only on the lowest levels of the hierarchy.

The output consists of the following factors.

• The links between the nodes with their capacity and type (PTP/PMP).
• The hierarchy level of each node.
• The total aggregated traffic of each node.
• The number and location of hubs.
• The number and configuration of PMP sectors for each hub.

### 3.2.2 Problem Formulation

The objective of the planning problem is to find a feasible network configuration subject to the constraints with minimizing the cost of the network, which network cost is the sum of the cost of the links and equipments.

We have to select a subset of the RBSs that would accommodate the hubs, we have to connect all RBSs to these hubs via FTP or PMP links and we have to direct the PMP sectors for best coverage.

Furthermore, the constraints of the minimal and maximal pointing angle separation for microwave links, the decision whether to use microwave or wireline link for a possible connection makes this problem difficult, besides that hierarchical network planning with black-box costs is already a complex task.

### 3.3 Proposed Algorithms

I split the planning problem into two parts and proposed separate algorithms to solve each task.
3.4 Settlement of Point-to-point Connections

In this section, a method is proposed to find a tree topology that combines microwave and wireline links, that is to solve the Settlement of Point-to-point Connections (SPC). The main framework will be outlined in Section 3.4.1 and its components will be detailed in Sections 3.4.2 – 3.4.4.

3.4.1 Algorithm for Microwave & Wireline Access Trees

The key of the algorithm is to first build a set of tree topologies from microwave links only, then to connect the root nodes of the resulting microwave trees with each other by wireline links in a second phase to form a feasible global solution.

1. (Initialization) Determine the initial number of "center nodes", which will serve as possible root nodes of minor microwave trees.

2. (Choose Centers) Choose the location of the required number of center nodes from $V$.

3. (Iteration):

   (a) (Microwave Part) Build up the microwave trees by a greedy algorithm (see Section 3.4.2).

   (b) (Push Down Levels) Set the level values of the root nodes of the resulting microwave trees to the highest possible value that is allowed by the cascading constraints.

   (c) (Wireline Part) Build up the wireline part by the algorithm shortly described in Section 3.4.3.

   (d) (Pop Up Levels) Set the root nodes of the resulting final trees to be RNCs and propagate the changes below them.
(e) (Evaluation) Evaluate the solution by calculating the cost of the network. Update the best configuration found so far if necessary.

(f) (Perturbation) Modify the number and position of center nodes.

(g) (Termination) Start a new iteration round if the desired terminal condition is not met.

4. (Post Processing) Restore the best configuration found during the search. Apply local changes in this network configuration to improve the final solution, and STOP.

In the Evaluation, the simulated annealing (e.g., [29]) technique was applied to decide whether a new configuration is acceptable or not. To modify the state of the selected center nodes (roots of the interference-sensitive parts) in the Perturbation step, three possibility was considered: a) adding a new, b) deleting or c) moving an existing center node to another position (that is the application of a) and b) together). The types of the modification is selected adaptively based on their past performance. The targets of the modification are selected with respect to the position of unutilized roots, existence of concentration areas and the position of nodes, which are not covered by interference-sensitive links. According to the practice, the terminal condition can be a limit on the number of iterations or a limit on the maximal number of unsuccessful iterations, in which the best-so-far solution failed to improve.

3.4.2 Interference-sensitive Microwave Part

A greedy-type method is proposed to form the microwave trees satisfying all constraints. The idea of this method is to build the microwave trees starting from the roots. In the first step we pick a root and connect as many nodes to it as possible. In the second step we pick another root or a previously connected node such that the largest number of unconnected nodes can be connected to it. We repeat this step until there exist nodes that can be connected to a previously added one.

1. (Initialization) Let \( C \) denote the set of connected nodes in the network (initially the selected centers are put in this set), and let \( U \) denote the set of unconnected nodes (initially \( U = V \setminus C \)). Set the level value of centers to 1 unless the previous run of the wireline part of MWAT has put an RNC to a center position, in which latter case its level will be 0.

2. (New Connection) Choose randomly a node \( v \in C \). Let \( P \subseteq U \) denote the set of nodes, which can be connected to \( v \) by a microwave link. For all \( u \in P \), let \( G_u \) denote the set of nodes that can be feasibly connected to node \( v \) together with node \( u \) and do:
(a) Let $\alpha$ be the "angle" of the link between $u$ and $v$ in the plain. Put $u$ in $G_u$. Let $\beta = \alpha$.

(b) Let $\beta \leftarrow \beta + S_{\text{min}}$. Find the first node $s \in P$, for which the "angle" $\gamma$ of link between $s$ and $v$ is $\beta < \gamma < \alpha + S_{\text{max}}$. If there exist such a node, then put $s$ in $G_u$, let $\beta \leftarrow \gamma$ and repeat this step.

3. (Best Connection) Choose the "best" $G$ from $G_u, u \in P$: The best $G$ has the most elements, with the lowest cost of the connections to break ties. Connect all nodes in $G$ to node $v$.

4. (Update) Set the level of nodes in $G$ to level $l_v + 1$. Take out $v$ from $C$. If $l_v < L_w - 1$, then add the new nodes to $C$ (let $C \leftarrow C \cup G$).

5. (Termination) While $U$ and $C$ are not empty, go back to step New Connection. If $C$ is empty but $U$ is not empty, then add $U$ to the set of the center nodes and STOP. In step Wireline Part of the MWAT algorithm (Step 3c), we give the extended set $C$ of the centers as input.

### 3.4.3 Interference-insensitive Wireline Part

In the wireline parts, new tree(s) are created from the unconnected nodes and the roots of the previous microwave phase. This task is solved by the algorithm proposed in Chapter 2.

The goal of the above planning algorithm is to build up a cost-optimal hierarchical access network, but this algorithm cannot handle the new microwave constraints (minimal- and maximal pointing angle separation) when it settles a new connection between two nodes of the network. This is the reason why this algorithm is applied only to form the wireline segment of the network.

### 3.4.4 Postprocessing

Heuristic local changes can be applied in the network in order to obtain cost saving and to improve the solution at the end.

In a systematic way, we try to find a new, "better" parent for all nodes in the network. (Let $v$ denote the parent node of node $u$.) A candidate new parent $s$ is accepted if the total network cost is lower by using a link between node $u$ and $s$, than between node $u$ and $v$. This link change can also mean a link type change from wireline to microwave and vice versa. We apply this simple method for all $u, v$ node pairs iteratively while the solution quality improves.
3.5 Combined Planning of Point-to-point and Point-to-multipoint Connections

In this section, different solutions are presented and analyzed for the Combined Planning of Point-to-point and Point-to-multipoint Connections (CPPMC). First, three main algorithms and four variants of one of them (COST-CLU) are introduced. All of them are based on a two-phase approach, where the two phases are practically two loops. In the outer loop, we search for the optimal number of hubs, while in the inner loop, we find the position and configuration of these hubs and allocate the remaining RBSs to them. The latter phase includes the configuration of the sector antennas.

3.5.1 Algorithms DIST-TREE, DIST-CLU and COST-CLU

The algorithms are based on the K-means clustering algorithm [14]. This algorithm is extended with PMP sector deployment ability and local optimization. First, the general frame of the algorithms is given:

1. (Initialization) Calculate the minimal number of hubs needed to form a feasible solution. Let \( K \) denote this minimal value. Let the overall best cost \( BC = \infty \) denoting that we do not have any solution so far.

2. (Iteration) Let the best cost found with \( K \) hubs \( BCK = \infty \) denoting that we do not have any solution with \( K \) hubs so far. Iterate the following steps:

   (a) (Initial Hub Positions) Choose \( K \) initial hub positions among the RBS locations (see Section 3.5.2).

   (b) (Clustering Loop) Iterate the following steps while the actual solution improves. During the iteration, if a solution is found with lower cost than \( BCK \), then store that as the best solution with \( K \) hubs and update \( BCK \) accordingly.

   • (Sector Computation 1) Deploy sectors to the hubs to cover a part of the RBSs. (See Section 3.5.3)

   • (RBS Allocation) Create a solution by allocating each uncovered RBS to the appropriate hub. A hub with the connected RBSs forms a cluster. (See Section 3.5.2)

   • (Inner Local Optimization 1) Apply local search operators to improve the allocation of RBSs to hubs in the solution.

   • (Relocation of Hub) In each cluster formed, search for the best location for the hub by trying out all RBS locations within the cluster as candidate position for the hub.
• (Sector Computation 2) It is carried out within the above Reloc-
ation step.

• (Inner Local Optimization 2) Repeat local search operators for im-
proving the allocation of RBSs to hubs in the actual solution.

(c) (Sector Computation 3) Restore the best solution found with \( K \) hubs. De-
ploy sector antennas to the hubs to cover a part of the RBSs by PMP
links.

(d) (Outer Local Optimization) Repeat local search operators for improving
the allocation of RBSs to the hubs.

(e) (Best Solution Update) Store the best solution found with \( K \) hubs as
the overall best solution if its cost is lower than \( BC \) and update \( BC \).

(f) (Termination) If \( K \) is equal to the number of RBSs in the network or
the best solution has failed to improve in the last few (ca. 3) iterations
of Step 2, then continue with Step 3. Otherwise let \( K \leftarrow K + 1 \) and
start a new iteration with Step 2.

3. (End) Return the overall best solution found.

In the naming of the algorithms, DIST means that we apply distance-based
evaluation of the actual solution in the Allocation/Relocation steps, while in case
of COST, we calculate the real network cost.

There are 3 places where sector computation for hubs is possible. This yields
three different algorithm variants depending on the place of sector computation
operations. Four algorithm variants of COST-CLU were investigated in order to
illustrate the idea. In the first variant (COST-CLU-1), the sector computation is
applied only after finding the best solution with \( K \) hubs, at Step 2c. In variant 2, it is applied also within the clustering loop, but only after the relocation step.
In COST-CLU-3 the antenna computation is performed in all possible steps. The
trivial variant using no PMP links will be referred to as COST-CLU-0.

The traditional distance-based evaluation allows fast hub relocation, however,
only the third Sector Computation step is possible in DIST-TREE and DIST-CLU.

3.5.2 Initial Hub Positions and Base Station Allocation Strate-
gies

In Step 2a above, the initial position of hubs is chosen as it was proposed in [15].
These positions should be as far from each other as possible. This technique is used
in DIST-CLU and COST-CLU. In these algorithms, in the RBS Allocation part of
Step 2b, each RBS is greedily allocated to the hub that provides the smallest actual
increment in the total cost. The RBSs are connected to the hub in a random order.
In case of DIST-TREE, a minimum spanning tree is created. Then the tree is split
into the required $K$ parts along the longest links. So we get $K$ groups, and a hub is randomly chosen from every group. Then the nodes are allocated to the initially selected hubs. Practically there is only one round of Clustering Loop in this case. The LOS visibility is taken into account here: only visible RBSs are considered for a hub as possible connections. If step *Sector Computation* 1 is applied (before the *RBS Allocation* step), then all RBSs can be connected to any of the hubs. First, as many RBSs are allocated via sector antennas to the hubs as many are worth. The remaining non-covered RBSs are allocated to the hubs as above.

### 3.5.3 Deploying Point-to-multipoint Sectors

Each hub is tested for placement of PMP sector(s) to cover a set of RBSs. Let us call the RBSs that are already or could be allocated to a hub as the potential RBS set. The procedure that computes the number and the pointing direction of the sector antenna(s) for a given hub is as follows:

1. Find the direction of the first sector antenna such that the cost saving is the greatest compared to the same allocation solution with PTP links. Take out those RBSs from the potential RBS set that can communicate via this antenna. When searching for the direction, it is needless to proceed from 0 to $2\pi$ in fine steps. It is sufficient to examine the directions, where the candidate RBSs are situated.

2. For the rest of RBSs, do:

   (a) Find the best direction for an additional sector antenna as above. According to the demands and available frequency bands, the covered area of the new antenna can partially or completely overlap an existing sector on this hub.

   (b) Remove the RBSs using this antenna from the set of potential RBSs.

   (c) If this reduced set of potential RBSs is empty or there is no more good direction, STOP.

The algorithm can handle arbitrary degree of sector overlapping. If overlapping sectors are allowed, then the number of sector antennas may be reduced. This opportunity exists only if a sector is covered by at least two other antennas. If these two other ones are non-overlapping, then it is examined whether the RBSs in the examined sector can be covered or not by the two other antennas. If all of the RBSs can be covered and that yields a lower network cost, then the antenna in question is omitted. This procedure is repeated until no more antennas can be omitted.
3.5.4 Local Optimization

In the algorithms, two local search operators are applied: *moving* an RBS from its corresponding hub to another one and *swapping* the corresponding hubs of two RBSs. Those move or swap possibilities are omitted, where the new position of a hub would be extremely far compared to its actual one. This extra checking operation saves considerable running time. The two operations are iterated with all possible RBS pairs until none of the operators can decrease the cost.

3.6 Performance Analysis

A short illustration of the behaviour of the proposed methods is presented in this section.

In the field of *Settlement of Point-to-point Connections* (SPC), first a lower bound approximation is given for a simple cost function, then numerical results are presented about the comparison of the solution of our algorithm and the lower bound, finally some graphical illustrations of sample networks are shown. Here, I have to emphasize that the proposed methods are able to handle practically any kind of complex cost functions for both the links and the equipment. Because of the lack of space, here only a substantially simplified, but still illustrative cost model is applied, in which the equipment costs do not play a significant role.

However, in the field of *Combined Planning of Point-to-point and Point-to-multipoint Connections* (CPPMC), the equipment costs are important components of the total cost of the network. First, the generation of the problem instances are presented, then the computational comparison of the proposed methods are given.

3.6.1 Lower Bound Approximation of SPC

Suppose that the cost of an allowed microwave link hardly depends on the length of it. In this case, the cost of the network is the sum of the cost of the required RNCs and \( k \) times the cost of a single microwave link (\( k \) denotes the number of these) and the cost of required wireline links, which forward the traffic of the minor microwave trees to the corresponding RNCs. Let us suppose unit cost for simple RBSs. The maximal size of a microwave tree can be calculated as

\[
MT = \sum_{l=1}^{L_p} \left[ \frac{S_{\text{max}}}{S_{\text{min}}} \right]^l,
\]

where \( L_p = L_w \) (full case). If we consider a node directly under under an RNC, \( L_p = L_w - 1 \) is used. Under an RNC there can be a full tree segment (MT_{full}) and \( n \) times simple tree segment (MT), where \( n \) denotes the number of nodes directly under an RNC. If the number of nodes connected to an RNC is \( R \), then
\[ n_{\text{min}} = (R - MT_{\text{full}})/MT. \] Let \( C_w \) denote the average cost of a wireline link, and \( C_m \) denote the cost of a microwave link. So the cost of the network controlled by that RNC, namely the objective function, can be approximated as follows:

\[ C_{\text{total}} = n \cdot C_w + (R - n) \cdot C_m \quad (3.2) \]

The lower bound is calculated by using \( n_{\text{min}} \) in Equation 3.2 instead of \( n \).

### 3.6.2 Numerical Results of SPC

In case of the above lower bound, we can compare the results of the proposed algorithm with the optimal values. Generally the ratio between the microwave and the wireline links in the results and the same ratio in the optimum is independent from the particular cost values. Very strict angle constraints are used from the practical point of view \( S_{\text{min}} = \frac{4}{3} \pi, S_{\text{max}} = \frac{4}{3} \pi \). Besides, \( L_w = 2 \) and \( L_m = 4 \) are applied. Networks of different sizes were examined at different maximal microwave link lengths. In case of sample networks labelled \( A \), the nodes in the networks were deployed according to a regular quadratic lattice. In the sample networks labelled \( B \), the nodes were uniformly distributed in a square area. The possible length values were 1, 2 or 3 times the distance between the farthest two adjacent points of the quadratic lattice (the diagonal in the lattice).

<table>
<thead>
<tr>
<th>number of nodes</th>
<th>1 length</th>
<th>2 length</th>
<th>3 length</th>
<th>any length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a)</td>
<td>b)</td>
<td>a)</td>
<td>b)</td>
</tr>
<tr>
<td>49</td>
<td>4.17%</td>
<td>14.58%</td>
<td>optimal</td>
<td>4.17%</td>
</tr>
<tr>
<td>100</td>
<td>5.06%</td>
<td>21.21%</td>
<td>2.02%</td>
<td>7.07%</td>
</tr>
<tr>
<td>225</td>
<td>6.54%</td>
<td>26.55%</td>
<td>2.55%</td>
<td>7.96%</td>
</tr>
<tr>
<td>400</td>
<td>7.88%</td>
<td>25.62%</td>
<td>2.71%</td>
<td>7.88%</td>
</tr>
<tr>
<td>900</td>
<td>8.71%</td>
<td>23.22%</td>
<td>3.38%</td>
<td>8.39%</td>
</tr>
</tbody>
</table>

Table 3.1: The quality of the solutions for different networks compared to the lower bound.

Table 3.1 shows that the algorithm can provide near optimal results independently of the network size. The results are given in \( \% \) and express the extra amount of wireline links added to the number of links compared to their number in the lower bound. In case of network 100 \( A \) and 225 \( A \), the results at any length limit are less good as in case of 400 \( A \) and 900 \( A \). The reason is that for smaller networks one extra wireline link can significantly decrease the performance of the final result (in our case it means almost 1\( \% \)). In case of networks labelled \( B \) the results are also close to the lower bound, but in case of "1 length" we cannot find such good results since this lower bound approximation underestimates the real optimum.
Note that with network 900 B in case of "1 length" the solution contains one extra RNC to avoid many long wireline links. Note that in case of smaller $S_{\text{min}}$ values the proposed algorithm can give even better results.

3.6.3 Graphical Illustration of SPC

In Figure 3.1, solutions are presented for sample network 49 A, 49 B and 225 A which also illustrate the network architecture. The bold lines symbolize the wireline links and the normal lines the microwave links. The larger circle symbolize the RNC and the smaller ones the RBSs.

(a) Type A with 49 nodes  (b) Type B with 49 nodes  (c) Type A with 225 nodes

Figure 3.1: Comparison of different networks with 1 length

3.6.4 Problem Instance Generation of CPPMC

Geographical Settlement Scenarios

Two kinds of test instance generation methods were examined. In both cases, the RBSs are dropped into a square area with random co-ordinates. The co-ordinates are according to the uniform distribution in the first case, while certain areas are more probable in the second case. Both methods have two parameters: the number of RBS nodes ($n$), and a real value ($d$ for Method 1, and $e$ for Method 2) which controls the density of the nodes.

Method 1: The size of the square area ($S$) is calculated from $S^2 = n/d$, where $d$ denotes the required number of RBSs per km$^2$. Then the nodes are dropped randomly to the square area $S \times S$ with uniformly distributed $x$ and $y$ co-ordinates between 0 and $S$. The traffic demand for each RBS is chosen randomly from the
interval $[1,2]$ Mbit/sec. See Figure 3.2(a) for an illustration.

(a) A sample result for Method 1   (b) A sample result Method 2

Figure 3.2: Solutions for instances generated by Methods 1 and 2

**Method 2:** Here the test area is a fixed square of $30 \times 30$km. A square grid is laid onto this area and the RBSs can only be placed at grid points. We first place randomly certain concentration points, which represent the centers of settlements. Then the RBSs will be placed to form concentration areas. A control parameter $e$ determines the distribution of the nodes. A small value for $e$ results in a highly; a large value in a less clustered arrangement of RBSs. The traffic for the RBSs is chosen as in case of generation **Method 1.** See Figure 3.2(b) for illustration.

**Cost Model**

Note that the proposed algorithm can handle any type of cost structure, since the cost function is a black box. The following is an illustrative example only.

The cost functions return $\infty$ if the given device cannot satisfy the technical constraints. There are three *base cost* parameters in the cost functions for links, hubs and antennas. The actual cost of the particular equipment or link is computed based these base values: $BC_{\text{link}} = 1$, $BC_{\text{hub}} = 3$ and $BC_{\text{ant}} = 5$.

**Link cost:** In microwave systems, cost typically depends *only* on the data rate of the links, and not as much on the distance of the endpoints. However, a small length-dependent factor is included to ensure that a shorter link with the same data rate is less expensive. As the traffic in the experiments is always below
2 Mbit/sec, the following simplified link cost function was used:

\[
C_{\text{link}}^{\text{PTP}}(u, v) = BC_{\text{link}} + \delta(u, v)/1000 \quad (3.3a)
\]

\[
C_{\text{link}}^{\text{PMP}}(u, v) = BC_{\text{link}}/2 + \delta(u, v)/1000 \quad (3.3b)
\]

where \(\delta(u, v)\) denotes the distance between the endpoints of the link in km. The cost saving of a PMP sector compared to the PTP solution mainly comes from the less expensive link cost.

**Node equipment cost:** It is computed from the total traffic \((t_v)\), from the portion of traffic that is communicated through PMP sectors \((t_s)\) and from the number of sectors \((n_s)\) at the particular hub. (Let the hub traffic \(t_h = t_t - t_s.\)) The cost is computed as follows:

\[
C_{\text{equip}}(v) = (BC_{\text{hub}} + \mu) + (BC_{\text{ant}} + \nu) \quad (3.4)
\]

where \(\mu\) and \(\nu\) are given by Table 3.2.

<table>
<thead>
<tr>
<th>(\mu)</th>
<th>if (t_h = 0)</th>
<th>(\mu = -3)</th>
<th>if (15 &lt; t_h \leq 25)</th>
<th>(\mu = 2.5)</th>
<th>if (60 &lt; t_h \leq 80)</th>
<th>(\mu = 4.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\mu)</td>
<td>if (0 &lt; t_h \leq 5)</td>
<td>(\mu = 1)</td>
<td>if (25 &lt; t_h \leq 35)</td>
<td>(\mu = 3)</td>
<td>if (80 &lt; t_h \leq 120)</td>
<td>(\mu = 5)</td>
</tr>
<tr>
<td>(\mu)</td>
<td>if (5 &lt; t_h \leq 10)</td>
<td>(\mu = 1.5)</td>
<td>if (35 &lt; t_h \leq 50)</td>
<td>(\mu = 3.5)</td>
<td>if (120 &lt; t_h)</td>
<td>(\mu = \infty)</td>
</tr>
</tbody>
</table>

| \(\mu\) | if \(n_s = 0\) | \(\mu = -5\) | if \(n_s > 0\) | \(\nu = n_s\) |

**Table 3.2: Computation table for equipment costs**

Depending on the scenario, the PMP and PTP cost are roughly equal when 5 nodes are connected to a hub.

**Constraints**

- Maximal length of PTP microwave links: 15 km.
- Coverage range of PMP sectors: 3.5 km.
- Beam width of sector antennas: 90°.
- Maximal transfer through a PMP sector: 30 Mbit/sec.
- Maximal number of RBSs attachable to a hub: 50.
- Overlapping is fully allowed.
3.6.5 Numerical Results of CPPMC

The three main algorithms were tested beside the four variants of COST-CLU according to the FMP sector deployment. The effect of the density parameters (\(d\) and \(e\)) to the different algorithms was also investigated. The tests involved 30 instances in each case and the average of the solution costs was used to characterize the particular problem class. Figures 3.3 and 3.4 illustrate the results. In all figures, the cost of the solution is compared to the worst solution cost found and it is displayed in percentage units (worst=100%). All networks in the examples had 300 nodes to demonstrate the applicability of the algorithms. Figures 3.3(a) and 3.3(b) show the effect of the FMP sector deployment variants against the density parameters. Figures 3.4(a) and 3.4(b) show the comparison of the three main algorithms: DIST-TREE, DIST-CLU and COST-CLU-3.

![Comparison of the deployment methods for the two instance types](image)

Figure 3.3: Comparison of the deployment methods for the two instance types

Figure 3.3(a) shows that the deployment of PMP sectors is more beneficial as the density of the RBSs (\(d\)) increases (COST-CLU-0 vs. COST-CLU-1-2-3). However, the procedure that aims the sector antennas only at the end of the optimization (COST-CLU-1) always lags behind the other ones (COST-CLU-2-3). We can recognize a special density value, beyond which COST-CLU-2 starts to give slightly better results than COST-CLU-3. At low densities of RBSs only minor condensation areas could evolve, and COST-CLU-3 can handle better this situation. In case of higher density values, the PMP sectors will be fully loaded, so it is more important to focus on minimizing the length of the PTP links, which is well achieved by COST-CLU-2.

As parameter \(e\) decreases, more distinct condensations of RBSs evolve, and the cost of the network, thanks to the favorable conditions for PMP sectors, declines near to linearly (see Figure 3.3(b)). Getting closer to \(e = 0\), the cost of the network decreases swiftly because all RBSs can be practically connected via sectorized hubs.
Figure 3.3 also back up the intuition that below a given density value we cannot reach better solution than the PTP solution (variant 0). This value of density can be estimated from the cost functions.

![Figure 3.4: Comparison of the main algorithms for the two instance types](image)

In networks with uniformly distributed RBSs (see Figure 3.4(a)), the DIST-TREE algorithm lags more and more behind the other two algorithms as the density increases. Algorithm COST-CLU-3 gives the best results.

The difference between the algorithms is less noticeable for networks where condensation areas exists (see Figure 3.4(b)), all the algorithms can 'easily' find the condensation areas. COST-CLU-3 is still 5 percent better than the others.

<table>
<thead>
<tr>
<th>Node number</th>
<th>DIST-TREE</th>
<th>DIST-CLU</th>
<th>COST-CLU-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>8.7 sec.</td>
<td>1.5 sec.</td>
<td>15.7 sec.</td>
</tr>
<tr>
<td>200</td>
<td>33.8 sec.</td>
<td>10.7 sec.</td>
<td>58.2 sec.</td>
</tr>
<tr>
<td>300</td>
<td>1 min. 29 sec.</td>
<td>29.1 sec.</td>
<td>2 min. 41 sec.</td>
</tr>
<tr>
<td>500</td>
<td>3 min. 33 sec.</td>
<td>1 min. 56 sec.</td>
<td>10 min. 53 sec.</td>
</tr>
</tbody>
</table>

Table 3.3: Comparison of the running times of the main algorithms

The running time also belongs to the performance of the algorithms. Table 3.3 shows that distance-based methods have the advantage that they run faster.¹

3.7 Conclusions

An efficient and robust algorithm was proposed for Settlement of Point-to-point Connections, which algorithm solves a special hierarchical mobile access network

---
¹The presented algorithms are implemented in C++ and the tests were performed on a Sun Ultra Sparc 5 machine with 256 MB RAM.
design task, in which a near-optimal 3G access network has to be planned by using both wireline and microwave links with black-box costs. There exist extra constraints on the minimal and maximal pointing angle separation at which microwave links of the same node can be situated. The method can be applied both for planning real networks and for evaluating network deployment scenarios.

Different methods were proposed for Combined Planning of Point-to-point and Point-to-multipoint Connections, which methods design access networks combining PTP and PMP links. The quality of the proposed solution is significantly better than the one without PMP links. The quality of the solution is also significantly better than the one achievable by deploying the sectorized hubs on a network, which was originally planned with only PTP links. Method COST-CLU gives better results than the traditional distance-based methods. The methods support non-linear, stepwise cost functions for the various links and equipment. The methods are not tailored to any specific cost function, since the equipment cost functions are external procedures to the optimization. The methods work with any type of PMP link, the antenna beam width and the range of coverage can be arbitrary. The computational demand of the algorithms is moderate, therefore they suit well the network planning problems occurring in practice.
Chapter 4

Solution of the Weighted Fermat-problem and its Applications to Topology Optimization

4.1 Introduction

In the field of network planning, local optimization techniques are frequently applied. The local optimization can be the building-block of each step of a planning algorithm or an independent final phase of a given method. In case of topology planning, local optimization means the improvement of the quality of the link structure between the nodes and many times it is restricted to small parts of the entire network. In these circumstances, not only is it possible to determine between which nodes should be a connection, but also links can be merged at extra nodes (weighted Fermat-points or Steiner-points, see [37]) in order to save cost.

In case of wireline links, the cost saving mainly comes from the decrease of the length of the trace (less cable canal is needed "under" the roads and pavements). In case of wireless links, the number of repeaters (relay stations) can be decreased. Of course, this kind of topology optimization is the more effective, the angles between the links to be merged are the smaller.

![Diagram](image)

Figure 4.1: Application of Fermat-problem
Exact and heuristic algorithms also exist to find the extra "merging" points, however, the general formulation of the solutions (the optimal position of the extra points) are rather complex. (The planning task belongs to the weighted Fermat-problem in case of merging two links [41, 40] and to the weighted Weber-problem in case of merging several links [43, 42].) Moreover, the problem of deciding in advance whether or not the application of such extra points results in cost saving and how much the gain will be is an open question. Existing solutions first determine the optimal position of the extra point (referred to as point $P$ in the following) and then calculate the improvement.

In this chapter, the following topics connected to the weighted Fermat-problem will be investigated. First in Section 4.2, some general properties of point $P$ are given and a new coordinate system is presented to describe $P$. Furthermore, the barycentric coordinates of $P$ are shown for a special symmetrical case. In Section 4.3, for the case, when the merging of two links decreases the cost, a lower bound is given for the multiplexing gain (capacity gain of merging links, denoted by $M$) and an upper bound is given for the angle (denoted by $\gamma$) between the two links to be merged. Then the connection of $M$ and $\gamma$ is analyzed. After that, the amount of the cost saving is analyzed as the function of $M$ and $\gamma$. Finally, the chapter is closed by a short conclusion.

4.2 A General Solution to the Weighted Fermat-problem

In this section, I analyze the geometrical properties of the point, which is the solution of the weighted Fermat-problem: a) I give new conditions to decide when the weights of the nodes determines the position of the solution in advance, b) I give a simple mathematical description of how to find the position of the solution in general and special cases, c) I show how the solution point divides the area of the "original triangle" in special symmetrical cases, d) I show the connection between the value of the objective function and the sides of the triangle.

**Definition 4.2.1.** The weighted Fermat-problem can be formulated as follows. Let $\triangle ABC$ be a given triangle with positive weights $w_A$, $w_B$, and $w_C$ associated with the three vertices. For any point $X$ in the plane, let $|AX|$, $|BX|$ and $|CX|$ be the Euclidean distances between $X$ and $A$, $B$, $C$. Then the weighted Fermat-problem is to find a point $P$ such that $F(P) = \min(F(X) \in \mathbb{R}^2)$, where $F(X) = w_A|AX| + w_B|BX| + w_C|CX|$. 

54
4.2.1 The Weighted Fermat-point and the Angle-technique

An important question is how the weights of the nodes determine the position of \( P \). Without the loss of generality, we can assume that \( \triangle ABC \) is labelled such that \( w_C \geq w_B \geq w_A \). Technological considerations focus the analysis on the case, when the weights of the nodes are positive (or may equal to zero). The case, when any of the weights can be less than zero, is out of the scope of the chapter. (A possible solution to this latter case can be found in [41].) If any of the weights can be equal to 0, then there are 4 singular cases.

**Claim 4.2.1.** If \( w_C = w_B = w_A = 0 \), then \( P \) can be any point of the plain.

**Claim 4.2.2.** If \( w_C > 0 \) and \( w_B = w_A = 0 \), then \( P = C \) is the solution.

**Claim 4.2.3.** If \( w_C > w_B > 0 \) and \( w_A = 0 \), then \( P = C \) is the solution.

**Claim 4.2.4.** If \( w_C = w_B > 0 \) and \( w_A = 0 \), then \( P \) can be any point of the section between node \( C \) and \( B \).

If all of the weights are positive, then there are 3 more possibilities to be investigated.

**Claim 4.2.5.** If \( w_C \geq w_B + w_A \), then \( P = C \) is the solution. (Note that if \( w_C = w_B + w_A \) and the nodes of the triangle are collinear as \( C - B - A \) or \( C - A - B \), then any point of the section between node \( C \) and the node in the middle can be also a solution.)

**Claim 4.2.6.** If \( w_C < w_B + w_A \) and the nodes of the triangle are collinear, then the node in the middle is the solution.

**Claim 4.2.7.** If \( w_C < w_B + w_A \) and \( A, B, C \) are not collinear, then \( P \) can be determined by Krrup's construction [41].

In the literature, there are several techniques to find/construct \( P \) (see e.g. [39, 38, 41]). Most of the techniques are based on geometrical construction by applying the so-called weight triangles (the ratio of the sides of these triangles corresponds to the weights) and Simpson lines (connecting a node of the weight triangle to the corresponding node of the original triangle, e.g. \( AA_1 \) in Figure 4.2). However, the general description of \( P \) is practically out of the scope of literature.

Krrup showed for the case described by Claim 4.2.7 that point \( P \) is in the intersection of three circles. These circles are the circumscribing circles of the weight triangles constructed outward the original triangle (see \( ABC_1, CBA_1, ACB_1 \) in Figure 4.2 and Theorem 1. in [41] for details).
Definition 4.2.2. In Figure 4.2, the angles with index $w$ denote the angles of the weight triangles and the angles with hat denote the viewing angle of the sides of $\triangle ABC$. In the following, these angles with hat are referred to as Fermat-angles. Since $PAC_1B$, $PBA_1C$ and $PCB_1A$ are cyclic quadrilaterals, the following equations hold true of the Fermat-angles.

\[
\hat{\alpha} = \pi - \alpha_w \\
\hat{\beta} = \pi - \beta_w \\
\hat{\gamma} = \pi - \gamma_w \\
\hat{\alpha} + \hat{\beta} + \hat{\gamma} = 2\pi
\] (4.1)

Figure 4.2: Fermat-angles

In the following, the intersection of the three viewing circles according to the Fermat-angles (around the sides of the triangle) is considered as a way to find $P$ in case of non-collinear triangles, and this technique is referred to as Angle-technique. In the rest of the section, the conditions are showed, by which the Angle-technique is applicable.

Theorem 4.2.1. If the angles of $\triangle ABC$ are less or equal to the corresponding Fermat-angles (i.e. $\alpha \leq \hat{\alpha}$ and $\beta \leq \hat{\beta}$ and $\gamma \leq \hat{\gamma}$), then $P$ will either be an interior point in $\triangle ABC$ or a vertex of it. Furthermore, only one of the angles can be equal to its corresponding Fermat-angle, otherwise Claim 4.2.1-4.2.6 determine $P$.

Proof. There are four cases to be investigated.

a) If $\alpha < \hat{\alpha}$ and $\beta < \hat{\beta}$ and $\gamma < \hat{\gamma}$, then any two of the viewing circles intersect each other at an interior point of the triangle and at the node, which is common for the corresponding two sides of the triangle. Because $P$ is unique [41], thus $P$
must be an interior point of the triangle, since it is the common intersection of the
circles.

b) Without the loss of generality, we can assume that $\gamma = \hat{\gamma}$ and $\alpha < \hat{\alpha}$ and $\beta < \hat{\beta}$. Then the viewing circle according to $\hat{\gamma}$ is equal to the circumscribing circle of $\triangle ABC$ and the two viewing circles intersect each other at an interior point of $\triangle ABC$ and at node $C$. Thus node $C$ gives the solution.

c) Without the loss of generality, we can assume that $\gamma = \hat{\gamma}$ and $\beta = \hat{\beta}$ and $\alpha < \hat{\alpha}$. Then $\hat{\alpha} = \pi + \alpha$ holds. Since $\hat{\alpha}$ is a viewing angle, $\hat{\alpha} \leq \pi$ also holds. Thus $\alpha \leq 0$. Since $\alpha \geq 0$ is also true in $\triangle ABC$, $\alpha$ must be equal to 0, which means that $\triangle ABC$ is collinear and Claim 4.2.1-4.2.6 determines the solution and this construction cannot be applied.

d) The last case, when $\alpha = \hat{\alpha}$ and $\beta = \hat{\beta}$ and $\gamma = \hat{\gamma}$ is impossible according to Definition 4.2.2.

**Theorem 4.2.2.** If one angle of $\triangle ABC$ is greater than its corresponding Fermat-angle, then $P$ will be the corresponding vertex of $\triangle ABC$ (e.g. if $\alpha > \hat{\alpha}$, then $P = A$). If there is more such angle, then Claim 4.2.1-4.2.5 would determine $P$.

**Proof.** There are three cases to be investigated.

a) Without the loss of generality, we can assume that $\gamma > \hat{\gamma}$ and $\alpha \leq \hat{\alpha}$ and $\beta \leq \hat{\beta}$. According to Definition 4.2.2, $\gamma > \hat{\gamma} = \pi - \gamma_w$; so $\gamma + \gamma_w > \pi$. Martelli [40] proved that in this case $P = C$ is the solution.

b) Without the loss of generality, we can assume that $\gamma > \hat{\gamma}$ and $\beta > \hat{\beta}$ and $\alpha \leq \hat{\alpha}$. Then $\hat{\alpha} + \beta + \gamma = 2\pi$ can be written as $\hat{\alpha} + \beta + \gamma > 2\pi = \pi + \pi$ followed by $\alpha > \pi + \alpha$. According to Definition 4.2.2, $\pi - \alpha_w > \pi + \alpha$, i.e. $0 > \alpha_w + \alpha$. In a non-collinear $\triangle ABC$, $\alpha > 0$, thus $\alpha_w$ is negative, which contradicts to Claim 4.2.7 and Claim 4.2.1-4.2.5 give $P$.

c) According to Definition 4.2.2, all angles of $\triangle ABC$ cannot be greater than the corresponding Fermat-angles.

**4.2.2 Calculation of the Fermat-angles**

The next theorem gives a general description of $P$ in a special coordinate system.

**Theorem 4.2.3.** Independently from $\triangle ABC$, point $P$ is uniquely determined by the Fermat-angles, so $P$ can be given in a Fermat-angle coordinate system as
\[ P = P(\hat{\alpha}, \hat{\beta}, \hat{\gamma}) \] and these angles can be expressed by the weights as follows.

\begin{align*}
\hat{\alpha} & = 2 \arccot \sqrt{\frac{w_A^2 - (w_B - w_C)^2}{(w_B + w_C)^2 - w_A^2}} \quad (4.2a) \\
\hat{\beta} & = 2 \arccot \sqrt{\frac{w_B^2 - (w_A - w_C)^2}{(w_A + w_C)^2 - w_B^2}} \quad (4.2b) \\
\hat{\gamma} & = 2 \arccot \sqrt{\frac{w_C^2 - (w_A - w_B)^2}{(w_A + w_B)^2 - w_C^2}} \quad (4.2c)
\end{align*}

\textbf{Proof.} Theorems 4.2.1 and 4.2.2 prove the applicability of the Angle-technique and the uniqueness of the solution comes from Definition 4.2.2. In order to prove the above formulae, see weight triangle \( CBA_1 \) in Figure 4.2. The cosine law gives that

\[ \alpha_w = \arccos \frac{(BC|w_B)^2 + (BC|w_C)^2 - (BC|w_A)^2}{2(BC|w_B)(BC|w_C)} = \arccos \frac{w_B^2 + w_C^2 - w_A^2}{2w_Bw_C}. \]

Using Definition 4.2.2, one gets that

\[ \hat{\alpha} = \pi - \arccos \frac{w_B^2 + w_C^2 - w_A^2}{2w_Bw_C}. \]

By taking the cosine of the above equation and Equation (4.2a), the following equation have to be proved (note that \( \pi/2 - \arctan x = \arccot x, x \geq 0 \))

\[ \frac{w_B^2 + w_C^2 - w_A^2}{2w_Bw_C} = \cos \left( 2 \arctan \sqrt{\frac{w_A^2 - (w_B - w_C)^2}{(w_B + w_C)^2 - w_A^2}} \right). \]

First, the arctangent expression is changed to arccos, then the double angle formula is applied and finally some basic simplifications are made.

\[ \cos \left( 2 \arctan \sqrt{\frac{w_A^2 - (w_B - w_C)^2}{(w_B + w_C)^2 - w_A^2}} \right) = \cos \left( 2 \arccos \frac{1}{\sqrt{\frac{w_A^2 - (w_B - w_C)^2}{(w_B + w_C)^2 - w_A^2} + 1}} \right) = \]

\[ = 2 \frac{1}{\frac{w_A^2 - (w_B - w_C)^2}{(w_B + w_C)^2 - w_A^2} + 1} - 1 = \ldots = \frac{w_B^2 + w_C^2 - w_A^2}{2w_Bw_C} \]

Equation (4.2b) and (4.2c) can be proven in the same way. \( \Box \)

If two weights are equal, e.g. \( w_A = w_B \), then we can reduce the weights to get \( w'_A = w'_B = 1 \) and \( w'_C = w_C/w_A \). In this case, node \( P \) remains the same.

58
Lemma 4.2.1. If \( w_A = w_B = 1 \), then the corresponding Fermat-angles can be calculated in the following way.

\[
\hat{\alpha} = \hat{\beta} = 2\text{arccot} \sqrt{\frac{1 - (1 - w_C)^2}{(1 + w_C)^2 - 1}} \tag{4.3a}
\]

\[
\hat{\gamma} = 2\text{arccot} \sqrt{\frac{w_C^2}{4 - w_C^2}} \tag{4.3b}
\]

Proof. After substituting \( w_A = w_B = 1 \) into Equation (4.2a), (4.2b) and (4.2c), we get the above equations. \( \square \)

4.2.3 Geometrical Properties of Some Special Cases

Theorem 4.2.4. If \( a = b \) and \( w_A = w_B = 1 \) in the triangle, then the optimal \( P \) point divides the area of the \( \triangle ABC \) into the following areas.

\[
\text{area}(ABP) = \frac{c^2 \cdot w_C}{4\sqrt{4 - w_C^2}} \tag{4.4a}
\]

\[
\text{area}(CAP) = \text{area}(BCP) = \frac{c}{4} \left[ \sqrt{a^2 - \left(\frac{c}{2}\right)^2} - \frac{c \cdot w_C}{2\sqrt{4 - w_C^2}} \right] \tag{4.4b}
\]

Note that the areas of the small triangles equal to the barycentric coordinates of the \( P \) point, so \( P \) can be exactly given in this co-ordinate system as well.

Proof. To calculate the area of \( \triangle ABP \), first calculate the height of it based on the tangent of \( \frac{\pi}{2} \): \( h_{ABP} = \frac{c}{2} \tan \frac{\pi}{2} \). After substituting the Equation (4.3b) for \( \hat{\gamma} \) we get: \( \text{area}(ABP) = \frac{c \cdot h_{ABP}}{2} = \frac{c^2 \cdot w_C}{4\sqrt{4 - w_C^2}} \).

Then we can calculate the areas of triangles \( CAP \) and \( BCP \). The areas of them are equal, so they can be expressed as: \( \text{area}(CAP) = \text{area}(BCP) = \frac{\text{area}(ABC) - \text{area}(ABP)}{2} = \frac{c}{4} \left[ \sqrt{a^2 - \left(\frac{c}{2}\right)^2} - \frac{c \cdot w_C}{2\sqrt{4 - w_C^2}} \right] \). \( \square \)

Claim 4.2.8. If all the weights are equal in a general triangle, then the sum of the distances (\( S \)) from the optimal \( P \) to the vertices of the triangle (\( A, B, C \)) can be expressed as

\[
S = \frac{a^2 - b^2}{|BP| - |AP|} = \frac{a^2 - c^2}{|CP| - |AP|} = \frac{b^2 - c^2}{|CP| - |BP|} \tag{4.5}
\]

(In case of isosceles triangles, the equation with the equal sides cannot be used. In case of equilateral triangles, the equations can be reduced as: \( S = a\sqrt{3} = b\sqrt{3} = c\sqrt{3} \).)

59
Proof. In that case, \( \hat{\alpha} = \hat{\beta} = \hat{\gamma} = \frac{2\pi}{3} \). According to the cosine law, we get that

\[
\begin{align*}
a^2 &= |BP|^2 + |CP|^2 + |BP| \cdot |CP| \\
b^2 &= |AP|^2 + |CP|^2 + |AP| \cdot |CP| \\
c^2 &= |AP|^2 + |BP|^2 + |AP| \cdot |BP|. \\
\end{align*}
\]

Combining the above equations, we get that

\[
\begin{align*}
a^2 - b^2 &= (|BP| - |AP|) (|AP| + |BP| + |CP|) \\
a^2 - c^2 &= (|CP| - |AP|) (|AP| + |BP| + |CP|) \\
b^2 - c^2 &= (|CP| - |BP|) (|AP| + |BP| + |CP|),
\end{align*}
\]

from which the statement of the lemma follows.

If \( a = b = c \), then Fermat-point is equal to the center of gravity, whose distance from the vertices is equal to \( \frac{1}{\sqrt{3}} \). Because of symmetrical reasons, \( S = 3 \frac{2}{\sqrt{3}} = a\sqrt{3} = b\sqrt{3} = c\sqrt{3} \).

\[\square\]

4.3 Application of the Weighted Fermat-point to Topology Optimization

In this section, I analyze the possible application areas of the weighted Fermat-problem in telecommunications environment: a) connection between the weights and the capacity dependent cost of the traffic demands, b) connection between the weights and the multiplexing gain applied at merging of links. On the other hand, I analyze the attainable gain in case of topology optimization: c) analysis of the efficient applicability of the solution against some parameters of the network, d) the analysis of the attainable gain provided by the solution (maximum, upper bound).

In the field of network planning, the weighted Fermat-problem can be applied to merge two links of a network at an extra point if it results in cost saving. Namely, the cost of the original links \((AC \text{ and } BC)\) is greater than the cost of the links to the extra point \((AP \text{ and } BP)\) and the cost of the merged link \((PC)\).

In practice, the cost of the links can be calculated according to the following formula:

\[
C_{\text{link}} (u, v) = \delta (u, v) \cdot f (t_{uv}) \tag{4.6}
\]

where the first component \( \delta (u, v) \) denotes the length of the link and the second component \( f (t_{uv}) \) denotes the capacity related cost of the link. This latter component can be linear, piece-wise constant, etc. For further details, please see Section 1.3. The so-called multiplexing gain gives how much capacity can be spared by merging links and the multiplexing gain indirectly determines the required capacity on link \( PC \) and the cost of this link.
4.3.1 Connection between the Weighted Fermat-problem and the Traffic Multiplexing Gain

Let us consider the weight of the links to be equivalent to their traffic related cost: \( w_A \equiv f(t_A) \) and \( w_B \equiv f(t_B) \) (where \( t_A \) and \( t_B \) are the traffic of node \( A \) and \( B \), respectively). Then a multiplexing gain \( M \) can be defined (\( 0 \leq M \leq 1 \)), which guarantees that the cost of the multiplexed traffic \( (w_C) \) satisfies the following assumptions:

1. \( w_C \leq w_A + w_B \) and
2. \( w_C \geq \max(w_A, w_B) \).

**Definition 4.3.1.** According to the above demands, the formula for calculating \( w_C \) by the multiplexing gain \( M \) is

\[
w_C = (1 - M) \min(w_A, w_B) + \max(w_A, w_B). \tag{4.7}
\]

Note that Angle-technique presented in Section 4.2.1 is just a way to find the solution of the weighted Fermat-problem. So if Angle-technique cannot be used in the topology optimization, then the weights trivially determine \( P \) (see Claim 4.2.1-4.2.6). However, in most of the cases, \( P \) is equal to \( C \), so improvement cannot be achieved. In some other cases \( P \) is equal to \( A \) or \( B \). The rest of the chapter focuses on the cases, which are connected to Claim 4.2.7.

4.3.2 Applicability Questions of the Angle-technique

**Lemma 4.3.1.** Angle-technique is applicable only if the weight \( w_C \) satisfies the following inequality

\[
w_C \leq \sqrt{\frac{(w_B - w_A)^2 + (w_B + w_A)^2 \cot(\frac{\gamma}{2})^2}{1 + \cot(\frac{\gamma}{2})^2}}. \tag{4.8}
\]

**Proof.** Theorem (4.2.1) says that the Angle-technique is applicable if \( \gamma \leq \tilde{\gamma} \). According to this, we get the following inequality from Equation (4.2c).

\[
\gamma \leq 2\arccot \sqrt{\frac{w_C^2 - (w_B - w_A)^2}{(w_B + w_A)^2 - w_C^2}}
\]

Solving the above inequality, we get that

\[
-\sqrt{\frac{(w_B - w_A)^2 + (w_B + w_A)^2 \cot(\frac{\gamma}{2})^2}{1 + \cot(\frac{\gamma}{2})^2}} \leq w_C \leq \sqrt{\frac{(w_B - w_A)^2 + (w_B + w_A)^2 \cot(\frac{\gamma}{2})^2}{1 + \cot(\frac{\gamma}{2})^2}}.
\]

Since in network planning \( w_C \geq 0 \) always holds, we consider only the positive bound for \( w_C \).
**Theorem 4.3.1.** Angle-technique is applicable only if the multiplexing gain satisfies the following inequality

\[
M \geq \frac{w_A + w_B - \frac{\sqrt{(w_B-w_A)^2 + (w_B+w_A)^2 \cot(\frac{\gamma}{2})^2}}}{\min(w_A, w_B)}.
\] (4.9)

**Proof.** By expressing the value of \( M \) from Equation (4.7), we get that

\[
M = \frac{w_A + w_B - w_C}{\min(w_A, w_B)}.
\]

Angle-technique is applicable if \( w_C \) is not more than the upper bound given in Lemma 4.3.1. So by combining Inequality (4.8) and the above equation, we get that

\[
M \geq \frac{w_A + w_B - \frac{\sqrt{(w_B-w_A)^2 + (w_B+w_A)^2 \cot(\frac{\gamma}{2})^2}}}{\min(w_A, w_B)}.
\]

\[\square\]

In the following, let \( w_{\text{min}} \) denote \( \min(w_A, w_B) \) and \( w_{\text{max}} \) denote \( \max(w_A, w_B) \).

If we know the multiplexing gain, then it may be important to know how great the angle between link \( AC \) and \( BC \) can be in order to efficiently apply Angle-technique (i.e. in this approach consider \( M \) and the weights to be known).

**Theorem 4.3.2.** The Angle-technique is applicable if the angle \( \gamma \) between link \( AC \) and \( BC \) is at most

\[
\gamma_{\text{max}} = 2\arccot \sqrt{\frac{2 - M}{M} \cdot \frac{2w_{\text{max}} - Mw_{\text{min}}}{2w_{\text{max}} + (2 - M)w_{\text{min}}}}.
\] (4.10)

**Proof.** By substituting Equation (4.7) into Equation (4.2c), one get that

\[
\hat{\gamma} = 2\arccot \sqrt{\frac{(w_{\text{min}}(1 - M) + w_{\text{max}})^2 - (w_B - w_A)^2}{(w_B + w_A)^2 - (w_{\text{min}}(1 - M) + w_{\text{max}})^2}}.
\]

It can be easily seen that one can substitute \((w_B + w_A)^2\) with \((w_{\text{min}} + w_{\text{max}})^2\), and \((w_B - w_A)^2\) with \((w_{\text{min}} - w_{\text{max}})^2\). Simplifying the above equation, one get that

\[
\hat{\gamma} = 2\arccot \sqrt{\frac{2 - M}{M} \cdot \frac{2w_{\text{max}} - Mw_{\text{min}}}{2w_{\text{max}} + (2 - M)w_{\text{min}}}}.
\] (4.11)

Since Theorem 4.2.1 says \( \gamma \leq \hat{\gamma} \) must be satisfied, the above equation is an upper bound for \( \gamma \). \[\square\]

62
Before the further analysis of the connection between $\gamma$ and $M$, let us see two interesting connections between $\hat{\gamma}$ and $M$.

**Lemma 4.3.2.** $\hat{\gamma}$ is inversely proportional to $w_C$ and directly proportional to $M$.

*Proof.* Consider Equation (4.2c), the arc cotangent expression is monotonically decreasing function of $w_C$, thus $\hat{\gamma}$ inversely proportional to $w_C$. Consider Definition 4.3.1, $w_C$ inversely proportional to $M$, thus $\hat{\gamma}$ directly proportional to $M$. \square

**Lemma 4.3.3.** If $M = 1$, then $\frac{1}{2}\pi \leq \hat{\gamma} \leq \frac{2}{3}\pi$.

*Proof.* Substitute $M = 1$ into Equation (4.11). Then we get:

$$\hat{\gamma} = 2\arccot\sqrt{\frac{2w_{\text{max}} - w_{\text{min}}}{2w_{\text{max}} + w_{\text{min}}}} = 2\arccot\sqrt{Q}$$

(4.12)

By proving $\frac{1}{3} \leq Q \leq 1$, we prove the statement of the lemma.

Let us change $w_{\text{min}}$ between 0 and $w_{\text{max}}$, where $w_{\text{max}}$ is any fixed positive value. If $w_{\text{min}} = 0$, then $Q = 1$. If $w_{\text{min}}$ increases, then $Q$ decreases until $w_{\text{min}} = w_{\text{max}}$. If $w_{\text{min}} = w_{\text{max}}$, then $Q = \frac{1}{3}$.

Let us change $w_{\text{max}}$ from $w_{\text{min}}$ to $\infty$, where $w_{\text{min}}$ is any fixed positive value. If $w_{\text{max}} = w_{\text{min}}$, then $Q = \frac{1}{3}$. If $w_{\text{max}}$ increases, then $Q$ increases. If $w_{\text{max}} \to \infty$, then $Q \to 1$.

Altogether, the $Q$ is between $\frac{1}{3}$ and 1. Thus $\frac{1}{2}\pi \leq \hat{\gamma} \leq \frac{2}{3}\pi$. \square

From a practical point of view, it is important to analyze the cases, when the cost of the network always or never can be decreased by merging two links at a legal multiplexing gain $0 \leq M \leq 1$. The following two theorems present these cases.

**Theorem 4.3.3.** If $\gamma > \frac{2}{3}\pi$, then there is no legal multiplexing gain $0 \leq M \leq 1$, for which Angle-technique is applicable.

*Proof.* Theorem 4.2.1 says that Angle-technique is applicable only if $\gamma \leq \hat{\gamma}$. Lemma 4.3.3 gives that $\hat{\gamma} \leq \frac{2}{3}\pi$ if $M = 1$. So if $\gamma > \frac{2}{3}\pi$, then $\hat{\gamma} > \frac{2}{3}\pi$ also have to be true. According to Lemma 4.3.2, $\hat{\gamma}$ is directly proportional to $M$, so $M > 1$ also have to be true. But a legal $M$ must be less or equal to 1 (see Definition 4.3.1). Thus if $\gamma > \frac{2}{3}\pi$, then there is no legal multiplexing gain $0 \leq M \leq 1$. \square

**Theorem 4.3.4.** If $\gamma < \frac{1}{3}\pi$, then there always exists a legal multiplexing gain $0 \leq M \leq 1$ for which Angle-technique is applicable.

*Proof.* Theorem 4.2.1 says that Angle-technique is applicable only if $\gamma \leq \hat{\gamma}$. Lemma 4.3.3 says if $M = 1$, then $\hat{\gamma} \geq \frac{1}{3}\pi$. So if $\gamma < \frac{1}{3}\pi$ and $M = 1$, then $\gamma < \hat{\gamma}$. Thus if $\gamma < \frac{1}{3}\pi$, then there always exists a legal multiplexing gain $0 \leq M \leq 1$. \square
4.3.3 Analysis of the Gain of the Topology Optimization

In case of network planning, the most important question is how much cost saving (gain) can be achieved by applying the Angle-technique. The gain can be defined as $1 - \frac{\text{new cost}}{\text{original cost}}$. First let us determine how the three nodes of $\triangle ABC$ have to be placed in the plane to provide the maximal achievable gain. Then let us calculate the value of this maximal gain.

**Lemma 4.3.4.** At any given $w_A$, $w_B$ and legal $M$, the maximal gain is achieved when $\gamma = 0$ and $|AC| = |BC|$.

**Proof.** Equation (4.7) says that $w_C \leq w_A + w_B$ for $0 \leq M \leq 1$. Let us apply the following notations. $S$ denotes the node, which is closer to $C$ and its weight denoted by $w_S$. $L$ denotes the node, which is farther from $C$ and its weight denoted by $w_L$. We want to find the lowest possible new cost for the network, since that case would result the maximal gain.

\[
N_{\text{new}} = w_L|LP| + w_S|SP| + w_C|PC| = w_L|LP| + w_L|PC| + w_S|SP| + w_S|PC| + (w_C - w_L - w_S)|PC| \\
\geq w_L|LC| + w_S|SC| + (w_C - w_L - w_S)|PC|
\]

If node $P$ is both in section $LC$ and $BC$, then $N_{\text{new}}$ is equal to the last expression and $\gamma = 0$. In that case, node $A$, $B$, and $C$ are collinear, so the optimal node $P$ is equal to $S$ (see Property 4.2.5 and 4.2.6). Then $N_{\text{new}}$ is

\[
N_{\text{new}} = w_L|LC| + (w_C - w_L)|SC|.
\]

Independently from $\gamma$, the original cost of the network is

\[
N_{\text{orig}} = w_A|AC| + w_B|BC| = w_L|LC| + w_S|SC|.
\]

It is easy to see if $\gamma$ tends to 0, then the gain of *Angle-technique* converges to the gain of the collinear case

\[
G = 1 - \frac{N_{\text{new}}}{N_{\text{orig}}} = \frac{N_{\text{orig}} - N_{\text{new}}}{N_{\text{orig}}} = \frac{w_L|LC| + w_S|SC| - w_L|LC| - (w_C - w_L)|SC|}{w_L|LC| + w_S|SC|} = \frac{(w_S + w_L - w_C)|SC|}{w_L|LC| + w_S|SC|}.
\]

In order to prove that the gain is maximal if $|AC| = |BC|$, namely $|LC| = |SC|$, it is enough to show the following inequality.

\[
G = \frac{(w_S + w_L - w_C)|SC|}{w_L|LC| + w_S|SC|} \leq \frac{(w_S + w_L - w_C)}{w_L + w_S}
\]

(4.13)
If \( w_C = w_S + w_L \), then both side is 0. Otherwise we can simplify the inequality as follows

\[
\frac{|SC|}{w_L|LC| + w_S|SC|} \leq \frac{1}{w_L + w_S} \\
|SC|(w_L + w_S) \leq w_L|LC| + w_S|SC| \\
|SC| \leq |LC|.
\]

Of course, \( |SC| \leq |LC| \), so at any given \( w_A, w_B \) and legal \( M \), the gain is maximal if \( \gamma = 0 \) and \( |AC| = |BC| \).

**Theorem 4.3.5.** The maximal gain that can be achieved by Angle-technique is

\[
G_{\text{max}} = \frac{w_{\text{min}}M}{w_A + w_B} \quad (4.14)
\]

**Proof.** Lemma 4.3.4 says that at any given \( w_A, w_B \) and legal \( M \), the gain tends to the maximum if \( \gamma \) tends to 0 and \( |AC| = |BC| \). According to Equation (4.13) the maximal gain can be formulated as

\[
G_{\text{max}} = \frac{(w_A + w_B - w_C)}{w_A + w_B}. \quad (4.15)
\]

According to Equation (4.7), the gain can be formulated as

\[
G_{\text{max}} = \frac{w_A + w_B - ((1 - M)w_{\text{min}} + w_{\text{max}})}{w_A + w_B} = \frac{w_A + w_B - w_{\text{min}} - w_{\text{max}} + Mw_{\text{min}}}{w_A + w_B} = \frac{w_{\text{min}}M}{w_A + w_B}.
\]

The above maximal gain is general upper bound of course, however, in particular cases, a tighter upper bound is needed. Before the further analysis of the gain, a connection between the sides of \( \triangle ABC \) and \( |PC| \) is given.

**Lemma 4.3.5.** If \( M \) is legal \((0 \leq M \leq 1)\), then \(|PC| \leq \min(|AC|, |BC|)\).

**Proof.** If \( M \) is legal, then \( w_C \geq w_{\text{max}} \). If \( w_C \) is the greatest weight, then \( \hat{\gamma} \) is the smallest Fermat-angle. Since \( \hat{\alpha} + \hat{\beta} + \hat{\gamma} = 2\pi \) and \( \hat{\alpha}, \hat{\beta}, \hat{\gamma} \leq \pi \), so \( \hat{\alpha}, \hat{\beta} \geq \frac{\pi}{2} \). In Figure 4.3, two viewing circles are shown.

\( V_1 \) is the \( \frac{\pi}{2} \) viewing circle of section \( B_2B \), where \( B_2 \) is constructed by reflecting \( B \) to \( C \). So \( |B_2C| = |B_1C| = |BC| \). The other viewing circle is \( V_2 \), which is the \( \hat{\alpha} \) viewing circle of section \( BC \). Since \( \hat{\alpha} \geq \frac{\pi}{2} \), thus \( V_2 \) is not outside of \( V_1 \). \( P \) is on \( V_2 \), so \( |PC| \) is not greater than \( |BC| \). Consequently, \( |PC| \leq |BC| \). In a similar way, we get that \( |PC| \leq |AC| \), so \( |PC| \leq \min(|AC|, |BC|) \).

65
From a technological point of view, it is important to know the value of the gain by taking the actual $\triangle ABC$ into consideration more accurately. An upper bound for the gain can be given by a linear function of $\gamma$ as follows.

**Theorem 4.3.6.**

$$G(\gamma) \leq G_{\max} \left( 1 - \frac{\gamma}{\gamma_{\max}} \right). \quad (4.16)$$

**Proof.** By definition, $G(\gamma) = 1 - \frac{N_{\text{new}}}{N_{\text{orig}}} = \frac{N_{\text{orig}} - N_{\text{new}}}{N_{\text{orig}}}$, where $N_{\text{orig}} = w_A|AC| + w_B|BC|$ and $N_{\text{new}} = w_A|AP| + w_B|BP| + w_C|PC| \geq \ldots = N_{\text{orig}} + |PC|(w_C - w_A - w_B)$ (see Lemma 4.3.4 for details). According to Equation (4.7)

$$N_{\text{new}} \geq N_{\text{orig}} - w_{\min} M|PC| = N'_{\text{new}}.$$

Then the following inequality holds for $G(\gamma)$.

$$G(\gamma) = \frac{N_{\text{orig}} - N_{\text{new}}}{N_{\text{orig}}} \leq \frac{N_{\text{orig}} - N'_{\text{new}}}{N_{\text{orig}}} \leq \frac{w_{\min} M|PC|}{N_{\text{orig}}}.$$

So we have to prove that

$$\frac{w_{\min} M|PC|}{N_{\text{orig}}} \leq G_{\max} \left( 1 - \frac{\gamma}{\gamma_{\max}} \right) = \frac{w_{\min} M}{w_{\min} + w_{\max}} \left( 1 - \frac{\gamma}{\gamma_{\max}} \right),$$

which is true if $M = 0$ or $w_{\min} = 0$. By transposition we get that

$$\frac{w_{\min}|PC| + w_{\max}|PC|}{w_{\min}Z_1 + w_{\max}Z_2} \leq 1 - \frac{\gamma}{\gamma_{\max}}, \quad (4.17)$$

where $Z_1 = |AC|$ if $w_A = w_{\min}$ else $Z_1 = |BC|$ and $Z_2 = |BC|$ if $w_B = w_{\max}$ else $Z_2 = |AC|$.
According to Lemma 4.3.5, $|PC| \leq \min(Z_1, Z_2)$, so Inequality (4.17) is true for both $\gamma = 0$ and $\gamma = \gamma_{\max}$. Since for $0 \leq \gamma \leq \gamma_{\max}$ the left side is a monotonously decreasing convex function and the right side is a monotonously decreasing linear function, so Inequality (4.17) holds on the whole region. \hfill \Box

If $w_{\max}$ is much greater than $w_{\min}$, then the upper bound given by Equation (4.16) gives a practically satisfying estimation. If the weights are relatively close to each other, then an estimation formula of the gain can be given.

**Claim 4.3.1.** If $\frac{w_{\max}}{w_{\min}} < 10$, then the gain can be estimated by the following formula:

$$\max_{\alpha, \beta} G(\gamma) \approx G_{\max} \left[ \vartheta_1 \left( \frac{\gamma}{\gamma_{\max}} \right)^2 - \vartheta_2 \frac{\gamma}{\gamma_{\max}} + 1 \right]$$

where

$$\begin{align*}
\vartheta_1 &= 1 - M \left( 0.0846 + 0.0679 \frac{w_{\max}}{w_{\min}} \right) \\
\vartheta_2 &= 2 - M \left( 0.0503 + 0.0691 \frac{w_{\max}}{w_{\min}} \right).
\end{align*}$$

(Note that the above formula underestimates the gain if $\gamma$ is close to $\gamma_{\max}$, however, then $G$ tends to 0.)

The precision of the formula is showed in Table 4.1. A short calculation based on the results shows that the cost of the network is given with 99.2% accuracy at the attainable maximal gain (at any given $\gamma$).

<table>
<thead>
<tr>
<th>$M$</th>
<th>$1$</th>
<th>$1.5$</th>
<th>$2$</th>
<th>$4$</th>
<th>$8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.002%</td>
<td>0.003%</td>
<td>0.002%</td>
<td>0.002%</td>
<td>0.006%</td>
</tr>
<tr>
<td>0.2</td>
<td>0.011%</td>
<td>0.010%</td>
<td>0.012%</td>
<td>0.005%</td>
<td>0.020%</td>
</tr>
<tr>
<td>0.3</td>
<td>0.024%</td>
<td>0.023%</td>
<td>0.024%</td>
<td>0.007%</td>
<td>0.044%</td>
</tr>
<tr>
<td>0.4</td>
<td>0.048%</td>
<td>0.035%</td>
<td>0.043%</td>
<td>0.011%</td>
<td>0.067%</td>
</tr>
<tr>
<td>0.5</td>
<td>0.070%</td>
<td>0.056%</td>
<td>0.070%</td>
<td>0.021%</td>
<td>0.102%</td>
</tr>
<tr>
<td>0.6</td>
<td>0.106%</td>
<td>0.086%</td>
<td>0.106%</td>
<td>0.040%</td>
<td>0.133%</td>
</tr>
<tr>
<td>0.7</td>
<td>0.137%</td>
<td>0.129%</td>
<td>0.152%</td>
<td>0.061%</td>
<td>0.164%</td>
</tr>
<tr>
<td>0.8</td>
<td>0.185%</td>
<td>0.185%</td>
<td>0.193%</td>
<td>0.110%</td>
<td>0.184%</td>
</tr>
<tr>
<td>0.9</td>
<td>0.244%</td>
<td>0.223%</td>
<td>0.269%</td>
<td>0.167%</td>
<td>0.192%</td>
</tr>
<tr>
<td>1.0</td>
<td>0.270%</td>
<td>0.307%</td>
<td>0.339%</td>
<td>0.261%</td>
<td>0.183%</td>
</tr>
</tbody>
</table>

**Table 4.1:** Precision of the evaluation of $G$

In the rows, different values for $M$ are evaluated. In the columns, different $w_{\max}$ and $w_{\min}$ ratios are evaluated. The results show how the formula estimates the
maximal gain. The results are the average of the absolute difference between the 'exact values' and the estimation for $\gamma = 0 \ldots \gamma_{\text{max}}$. (Note that the 'exact values' numerically can be computed by the combination of the Weiszfeld algorithm and a successive approximation process.)

![Diagram](image)

Figure 4.4: Connection between $\gamma$ and $G$ with different $M$ values

For example, the achievable gain at $w_{\text{max}} = w_{\text{min}}$ is shown in Figure 4.4. The lowest curve is for $M = 0.1$, the uppermost is for $M = 1$ and between them $M$ increases in 0.1 steps.

### 4.4 Conclusions

The Fermat-angles are proposed as a new representation of $P$, which is the solution to the weighted Fermat-problem. The Angle-technique is proposed as a new construction method of $P$. Several geometrical properties of the solution are analyzed including the connection of the Fermat-angles with the angles of the triangle and with the weights of the vertices.

For the case of topology optimization applications (when the merging of links may result in cost savings), some formulae are given, which tells us whether it is worth to construct $P$ at all and if it is worth, then the formulae approximate the achievable cost saving in advance (without constructing $P$). Based on the presented results, some applications using the solution of the weighted Fermat-problem for topology optimization may be accelerated or may be extended to the weighted case (e.g., [37]).
Chapter 5

Optimization of OSPF Administrative Weights in Access Networks

5.1 Introduction

In this chapter, the configuration of OSPF-based [4, 45] fault-tolerant access networks is analyzed. The planning problem itself is introduced through a technological example based on the UMTS Terrestrial Radio Access Networks (UTRAN, see Section 2.4.1) and the analysis of the proposed solution is also connected to this technology (see Section 5.5).

The routing in the initial releases of UTRAN is trivial task, because the topology determines the paths by itself. In order to get fault-tolerant networks, however, complex topologies have to be applied, which make the OSPF configuration complicated. First the UTRAN and OSPF will be introduced, then the fault-tolerant related issues will be presented.

Because of the original tree-like topology of UMTS terrestrial access networks, these networks are very sensitive to the failures and in case of larger networks, even a single failure can occurs the loss of high amount of data. One solution is to use fault-tolerant topologies, such as ring or mesh like topologies, however, the cost of them can be significantly greater than the simple tree topology. A possible alternative solution is to keep the basic tree topology and expand it with some additional links at those parts of the network where faults are critical. The main advantage of the network extension is that an acceptable equilibrium can be found between the increment of the cost and increment of the fault tolerance capability of the network.

Solving this kind of network extension problem is not an obvious task, because the state space of the problem is large and the goal is twofold, we want to increase
the fault-tolerance with minimal investment. We analyze the properties of this kind of network extension solution [J2] and propose a planning algorithm to this problem. It is shown that our algorithm gives high quality solutions.

In the first phase, the transport technology of the UMTS terrestrial access networks will be ATM (Asynchronous Transfer Mode) based, but the ultimate goal is to apply IP technology here as well. Because the most wide-spread routing protocol of IP networks is OSPF protocol, it seems to be used in UTRANs, too. The OSPF uses the shortest path for routing packets according to the given weights of the links and it may apply the so-called Equal-Cost Multipath (ECMP) principle in cases of multiple shortest paths. The OSPF routing procedure works essentially as follows. A positive integer number is assigned to each link (called weight), and these values are sent to the network nodes using the OSPF link-state flooding mechanism. Every intermediate node along a path determine the next hop of the packet towards the destination node using local routing table. Building up the routing tables of the nodes, the shortest paths are calculated on the basis of the link weights.

Our model used in [J2] is able to handle only the explicitly defined paths (this is the case in ATM based access network), and cannot consider the special properties of OSPF routing. This algorithm is kept to determine the position and capacity of the extra links required to obtain the defined protection level, and to determine the routes of protection paths along with the required capacity increment (if any). In this case, the problem is that the backup paths are determined explicitly, therefore, a method is needed, which computes an OSPF weight system reproducing the original paths.

5.2 Problem Statement

The goal of this section is to outline the used network model (for details, please see Section 1.3.), as well as the clarification of the exact optimization problem to be solved and the considered conditions.

The network topology is a spanning tree containing the default links. This tree is completed with some additional backup links to provide the pre-defined level of network availability.

It is required that at most two links can be originated from a node: exactly one default link and at most one backup link, which can connect only nodes on the same level or on adjoining upper levels.

It is required that each backup path contains only one backup link and some default links, since only one link failure is considered at a time.

As a consequence of OSPF routing, all used paths in a given state (normal working operation or any of the failure-cases) must form a tree.

70
5.2.1 Input and Output

The optimization proceeds from a network with known topology and link capacities \( (C(e)) \) given as input. For each node \( v \) and for each failure scenario \( f \) we are also given a path \( p_v^f \subseteq E^f \). Where the edge set \( E^f \subseteq E, f \in \{0, 1, 2, \ldots, F\} \) expresses the available edges in the \( f \)th failure scenario. \( (E^0 \) corresponds to the case when there is no link failure.) The path \( p_v^0 \) is called default path the others are the backup paths.

The output is a positive integer number \( w(e) \) assigned for each link \( e \in E \) as the OSPF weight. If reproducing the predefined path system is possible, then the output is an adequate OSPF weight for each link.

If there exists no such a weight system, then it is proposed where and how to modify some backup paths to achieve an OSPF conform path system. Furthermore, in infeasible cases, the output weight system will minimize the overload on the links compared to the input link capacities.

5.2.2 Problem Formulation

We search for an adequate \( w(e) \) weight for each \( e \in E \), which weight provides that the traffic of each node \( v \) is routed according to the pre-defined path system \( (p_v^f) \) in all cases (normal working and failure cases).

Let \( P_v^i \) denote the \( i \)th possible path from node \( v \) to the root. The planning problem is to find a weight system where:

\[
W(p_v^f) = \min_{P_v^i \in E^f} W(P_v^i), \quad \forall v \in V \land f \in \{0, \ldots, F\} \tag{5.1}
\]

with respect to

\[
W(p_v^f) < W(P_v^i), \quad \forall v \in V \land f \in \{0, \ldots, F\} \tag{5.2a}
\]

\[
w(e) \in \mathbb{Z}^+, \quad \forall e \in E \tag{5.2b}
\]

Equation 5.1 ensures that the predefined path \( p_v^f \) will be minimal in the network. In that case, however, there can be other paths, whose weight is equal to \( W(p_v^f) \). Equation 5.2a provides the required uniqueness of the resulting path system. Since the weights will be a parameters of a real telecommunications system, the weights must a positive integer number (see Equation 5.2b).

If there exists no a weight system satisfying the above requirements, then the task is to minimize the overload in the network:

\[
\min \sum_{e \in E} \left( \max_{f \in \{0, \ldots, F\}} \sum_{P_v^i \in P_v} \tau_v^e - C(e) \right) \tag{5.3}
\]

71
where $P_v$ is the shortest path from node $v$ to the root in the $f$th failure scenario. If there exists a unique $P_v$, then $\tau_v^* = \tau_v$. If there are multiple shortest paths of node in a given case, then $\tau_v^*$ have to be calculated according to the ECMP principle. (E.g. if there are triple shortest paths, then $\tau_v^* = \frac{1}{3}\tau_v$.)

### 5.3 Proposed Algorithm

As a solution, I propose to combine the strategy of path "restoration" with the strategy of overload decrement. Thus if the predefined path system cannot be reproduced for some reason, then the proposed solution can still minimize the overload of the links. The framework of the proposed OSPF Weight Setting Algorithm (OWSA) is illustrated in Figure 5.1.

![Diagram of OWSA](image)

**Figure 5.1: The framework of OWSA**

Figure 5.1 shows that OWSA iteratively uses 5 procedures (for details see the following subsections) in order to solve the inverse shortest path problem in the
following way:

1. *(Initialization)* Initial settings and analysis of the network.
   
   (a) *(Initial Weights)* Initialize the weights of the links (by procedure IWS).
   
   (b) *(Decomposition)* Search for infeasible situations caused by circles of backup links. If it is possible to decompose the circles, then decompose them (by procedure DEC_1 or DEC_2), else STOP.

2. *(Weight Adjustment)* Iterative adjustment of the link weights.
   
   (a) *(Basic Adjustment)* Eliminate the link overload in the network by modifying the weight of the links (by procedure BWS), and repeat this step until the overload decreases.
   
   (b) *(Advanced Adjustment)* Eliminate the difference between the predefined and the actual path system by modifying the weight of the links (by procedure AWS), and repeat this step until the difference decreases.
   
   (c) *(Special Adjustment)* There can be paths correlating to each other, so they cannot be corrected one after the other in Step 2b. Correct these paths (by procedure SWS) until the difference between the predefined and the actual path system decreases.
   
   (d) *(Termination)* If all repairable errors are corrected, then STOP. Else go to Step 2a.

If it is required to decompose the circles, then the same backup path system cannot be reproduced as in the predefined case. In most of the cases, however, the overall performance of the network still remains the same after the decomposition.

The computational results presented in Section 5.5 back up that the proposed algorithm practically can solve the problem of mapping dedicated paths into OSPF-based paths.

The following subsections describe the above mentioned procedures applied by OWSA.

### 5.3.1 Initialization

First of all, initial link weights are given (see procedure IWS) and the existence of possible circles of backup links is analyzed and the decomposition of them is done. It is shown that no weight system can reproduce the predefined path system in case of circles, however, in most of the cases, the overall performance of the network can remain the same by the decomposition strategies shown for the two possible configurations of circles (see procedure DEC_1 and DEC_2).
**Initial Weight Setting (IWS)**

According to the conditions, the traffic paths must use only default links in case of regular network operation and each backup path must contain only one backup link. In order to meet these conditions, let us set the weight of each default link to 1 and set the weight of the backup links to 3. However, this weight setting does not provide that the traffic between two nodes actually uses its predefined paths. Therefore some links may become overloaded, so further refinement is needed in the weight setting.

**Directed Circles of Backup Links**

**Claim 5.3.1.** A circle of backup links causes an infeasible situation if they are on the same level in the network and form a directed circle according to the assumptions in Section 5.2. Figure 5.2 illustrates such a circle.

![Directed circle of backup links](image)

Figure 5.2: Directed circle of backup links

*Proof.* For the sake of simplicity, consider a circle of 3 links. Let us suppose to the contrary that a proper weight system exists. This weight system the following conditions are held:

\[
\begin{align*}
w(AB) + w(BR) &< w(CA) + w(CR) \\
w(BC) + w(CR) &< w(AB) + w(AR) \\
w(CA) + w(AR) &< w(BC) + w(BR)
\end{align*}
\] (5.4)

Summing up these in inequalities, we get the same on both sides, which is a contradiction. \(\square\)

That means that there are no proper weight system to map the dedicated routing. (The proof is similar for the case of circles with more nodes.) However, the overall performance of the network reliability can be preserved if we enable
some modifications in the predefined path system. In the next sections, these techniques are presented. For the sake of simplicity, the circles of the examples consist of 3 links. Of course, both techniques can be extended to circles of more links.

Circles with Default Paths Merging at the Same Node (DEC₁)

According to the above proof, we cannot solve the directed circle problem, but if the default paths of the nodes in the circle merge in the same node, then we can decompose the circle. This means that by modifying the backup path system we can delete a link from the circle without degrading the performance of the network. Figure 5.3 illustrates the case when the default paths merge in the root. (Let $\tau_A$, $\tau_B$ and $\tau_C$ denote the traffic demand of the nodes, thus the required capacity of the links, as well.)

![Directed circle with default paths merging at the same node](image)

Let $\tau_C$ be the smallest traffic demand (or one of them if there are more). If there is a failure between node $C$ and $R$, then the traffic $\tau_C$ can go also towards node $B$. Since one-failure scenario is assumed, this traffic $\tau_C$ can also go through node $B$ towards node $R$, because it is not greater than traffic $\tau_A$ (there is backup path on link $BR$ for node $A$, whose default path still works). These altogether mean that the backup link $CA$ is needless. So we can delete this backup link, use link $BC$ as backup link in both direction and solve the problem with procedure AWS. So the overall performance of the network remains the same.

Circles with Default Paths Merging at Different Nodes (DEC₂)

Figure 5.4 illustrates the situation, when some default links of the nodes in the circle merge before all of them are merged: default paths of node $A$ and $C$ are merged at node $D$, but the default path of node $B$ is merged to them only at node $R$.
The result of the decomposition depends on the amount of the traffic of the nodes:

- **Success**: If node $B$ or $C$ has the smallest traffic, then the circle can be decomposed as in case of DEC$_1$.

- **Failure**: If node $A$ has the smallest traffic, then only its traffic can be rerouted. Since $C$ is not so protected as $A$ (node $C$ is not protected for failure of link $DR$, but $A$ is), node $A$ would not have backup path for all failures protected by the predefined case. So the circle cannot be decomposed and the planning problem cannot be solved.

### 5.3.2 Iterative weight adjustment

In the second part of OWSA, an iterative weight adjustment is proposed. First the link overload is decreased in the network (see procedure BWS). Then the difference between the predefined and the actual path system is decreased (see procedure AWS). After that those paths are corrected, which correlate to each other (see procedure SWS). Finally these three steps are repeated until all repairable errors are corrected.

**Basic Weight Setting (BWS)**

A simple procedure (called BWS) is applied to find a weight system, which provides that there are no overloaded links in the network\(^1\). The framework of the method is the following:

\(^1\)If there is an overloaded link, then the actual path system surely differs from the predefined one.
1. Calculate the total network overload (the sum of the overload on the links; denoted by $O_{\text{old}}$).

2. For all $v \in V$ do:
   
   (a) Simulate each failure scenario $f$ for $p_v^f \neq \emptyset$. If a link is overloaded in any of the scenarios, then increase its weight by 1.
   
   (b) Repeat Step 2a while there is an overloaded link in case of the failures scenarios.

3. Calculate again the total network overload (denoted by $O_{\text{new}}$).

4. If $O_{\text{new}} < O_{\text{old}}$, then go to Step 2, else Stop.

**Advanced Weight Setting (AWS)**

A method is proposed to take the default-backup system of our model into consideration and decreasing the difference between the predefined and the actual path system. This procedure can correct almost all overload situations, which may arise in the network. Note that procedure AWS is more efficient if the initial weights of the backup links are set to a value larger than 3. It is proposed to correct the weight system of the nodes by going down the topology from the root to the leaves:

1. Let $L = 1$.

2. Each failure scenario $f$ is simulated for each $v \in V$ for which $l_v = L$ and $p_v^f \neq \emptyset$. Find the illegal paths (non-backup and non-default paths) used by $v$. Let the weight of these path be $W_f$. If such an illegal path exists, then do the followings:
   
   (a) Find a still existing backup path for $v$, which has the smallest weight $W_{\text{org}} := W(p_v^f)$.
   
   (b) Correct the weight of the first default link which can be found on each illegal path from $v$ to the $r$. The correction is to increase their weight by $W_{\text{org}} - W_f + 1$. (The weight of a particular corrected link is increased only once even if it is in several illegal paths.)

3. $L \leftarrow L + 1$. If $\exists v \in V, l_v = L$, then go to Step 2.

4. If the difference between the predefined and the actual path system is decreased, then go to Step 1, else STOP.

Note that it is more beneficial to start the correction of the illegal paths with those nodes, which have shorter illegal paths. Since the procedure modifies only the weight of the default links, it must be completed with some special cases presented in the followings.
Special Weight Setting (SWS)

In some situations, the failures can be corrected separately, but if they are combined together, they cannot be solved by procedure AWS. Figure 5.5 shows an example for this special situation.

![Figure 5.5: Special case: special weight setting](image)

Node $A$ has a backup path $A - F - E - D - R$ and node $F$ has a backup path $F - A - B - C - R$. If we wanted to correct node $A$ in case of failure on link $CR$, then we should increase the weight of link $AB$ (see the round brackets). If we wanted to correct node $F$ in case of failure on link $DR$ then we should increase the weight of link $FE$ (see the brackets). It is easy to see that the correction of node $A$ and $F$ after each other cause an infinite cycle for procedure AWS (without the termination condition). Therefore, if we want to correct both node $A$ and $F$, then we have to modify links $CD$ and $DC$ (see the braces). The new weights for these links are as Equation 5.5:

$$w(CD) = w(DC) = \max(w(AF) + w(FE) + w(ED), w(FA) + w(AB) + w(BC))$$ (5.5)

Of course, this method can be easily extended to the case, when there are other nodes between node $B$ and $C$ and between $E$ and $D$.

### 5.4 Reference Algorithms

In this section, two main reference algorithms are described, which will be applied in Section 5.5 during the performance analysis.
5.4.1 Heuristic Reference Algorithms

In this section, a randomized weight setting algorithm (based on the one proposed in [56]) is presented, which is used as reference method in the evaluation of the proposed weight setting procedure. This algorithm is based on the well-known Simulated Annealing [57, 58] meta-heuristic, which is an efficient technique for solving complex optimization tasks with large state space. The most important steps of the reference algorithm are listed below:

- **(Initialization)** The initial weight setting method (presented in the previous Section) is used to adjust the initial weights for the default and backup links.

- **(Demand allocation)** According to the current weight system all demands are routed and all failures are simulated. Then the required link capacities are calculated and the overloaded links are searched, as well as the total network overload is calculated (denoted by $O_{old}$).

- **(Weight modification)** In this phase we can select from two possibilities, the first is a simple weight adjustment, and the second one is a sophisticated strategy.

1. One overloaded link is selected randomly and its weight is incremented by 1.

2. Select a link randomly and modify its weight by 1 according to the probability $P_{mod}$: if the link is overloaded, then its weight should be increased and $P_{mod} = \frac{\text{overload}}{\text{maximal overload}}$; if the link is underutilized, then its weight should be decreased and $P_{mod} = \frac{\text{underutilization}}{\text{maximal underutilization}}$. If we do not modify weight of the link, then select another link randomly.

Then all failures are simulated, the overloaded links are searched and the total network overload is calculated again (denoted by $O_{new}$).

- **(Evaluation)** Using the so-called stochastic acceptance criteria of simulated annealing it is decided whether the modification of the weight is acceptable or not. The network modification will be accepted with the probability

$$P_{accept} = \min\left\{1, \exp\left(-\frac{O_{new} - O_{old}}{T}\right)\right\}$$

(5.6)

where $T$ is the so-called temperature, which decreases exponentially during the running of the algorithm. If the overload in case of the new weights is lower than in case of the weights before the modification the new weight system is always accepted. If the new overload is greater than the old one, then the acceptance depends in the above criteria. At the beginning of the
optimization the probability of the acceptance of an overloaded state is close to 1, while later this probability decreases significantly.

- (Termination) The optimization continues at the Demand allocation step if the desired terminal condition is not met.

According to the practice, the terminal condition can be a limit on the number of iterations or a limit on the maximal number of unsuccessful iterations, in which the best-so-far solution failed to improve.

### 5.4.2 The Exact LP Solution

This section exhibits a linear programming based exact solution to the problem of finding OSPF weight setting with various objective functions.

To give an LP formalization of the problem we can use the following well know duality theorem of shortest paths.

Let $G = (V, E)$ is a directed graph, and $w : E \rightarrow \mathbb{Z}^+$ a weight function on the edges. A $\pi : V \rightarrow \mathbb{Z}$ potential is called feasible if

$$w(u, v) \geq \pi(v) - \pi(u)$$ \hspace{1cm} (5.7)

holds for all edges $(u, v) \in E$. An edge $(u, v)$ is called $\pi$-tight or tight if $w(u, v) = \pi(v) - \pi(u)$.

**Claim 5.4.1.** For any fixed node $s \in V$ there exists a feasible potential $\pi$, such that a path $p$ from $s$ to an arbitrary node $t$ is a shortest $s\rightarrow t$ path if and only if each edge of $p$ is $\pi$-tight.

**Proof.** Let us define $\pi(v)$ to be the length of the shortest $s\rightarrow v$ path. It is easy to check that this potential is feasible and meets the requirements above. $\square$

The following claim is also easy to see.

**Claim 5.4.2.** If a path $p$ is not a shortest path, then there exists no feasible potential $\pi$ such that each edge of $p$ is $\pi$-tight.

Using these claims, we are able to formulate our problem. Our goal is to find a weight function of the edges, by which the shortest path of the nodes equal to their predefined paths in all states of the network. These states are the normal work state (0) and the states of the possible default link failures ($1 \ldots F$, where $F$ is equal to the number of nodes). To sum up, we are looking for such weight system, by which the potentials are feasible in all states and all the links used in $f$th state (referred to as $E^f$) are $\pi$-tight and for all other links $w(u, v) > \pi^f(v) - \pi^f(u)$ holds (where $\pi^f(u)$ is the potential of node $u$ in the $f$th state). The linear program formulates as follows:
For each edge \((u, v)\) a variable \(w(u, v)\) expressing the OSPF weight is introduced. Moreover for each fault scenario \(f\) (0 for the normal work state and \(1 \ldots F\), for all possible default link failures) and for each node \(u\) we introduce a variable \(\pi^f(u)\).

\[
\begin{align*}
    w(u, v) &\geq 1 \quad \forall (u, v) \in E \\
    \pi^f(u) &\geq 0 \quad \forall u \in V, f = 1 \ldots F \\
    w(u, v) &= \pi^f(v) - \pi^f(u) \quad \forall (u, v) \in E^f \\
    w(u, v) &\geq \pi^f(u) - \pi^f(v) + 1 \quad \forall (u, v) \in E^f \\
    w(u, v) &\geq \pi^f(u) - \pi^f(v) + 1 \quad \forall (u, v) \in E \setminus E^f
\end{align*}
\]  

(5.8a, 5.8b, 5.8c, 5.8d, 5.8e, 5.8f)

Claim 5.4.3. The above linear program is feasible if and only if there exists an appropriate weightings.

Proof. The above linear inequalities express that \(\pi^f\) is a feasible potentials in the fault scenario \(f\) and exactly the elements of \(E^f\) are tight. According to Claim 5.4.1 and Claim 5.4.2 this means that the required set of paths are unique shortest path with respect to the weighting \(w(u, v)\).

On the other hand, if \(w(u, v)\) is a proper weightings, then let \(\pi^f(u)\) be the distance of the node \(u\) from the root in the \(f\)th fault scenario. This gives a feasible solution of the above linear program.

If our aim is to find solution with "small" weights i.e. we want to minimize the largest weight, then we can introduce an additional auxiliary variable \(Z\) and minimize it keeping the above constraint and also the followings.

\[
Z \geq w(u, v) \quad \forall (u, v) \in E
\]

(5.8g)

5.5 Performance Analysis

The goal of the analysis is to test the OSPF Weight Setting Algorithm (OWSA) in case of real network sizes, so let us use networks with 50, 100, 150, 200 nodes. Ten networks of each size are generated, the topology of the networks satisfies all the technological requirements and constraints (maximum 2 incoming links per node, the depth of a tree is 5) and it is very close to an optimized UMTS access network. Furthermore, there were no circles from backup links (see Section 5.3.1) in any instance networks. In the literature, there are no exact values of what percentage of the RBSs are planned to be protected in the access network, therefore 4 different protection level scenarios are examined for each network sizes: 10%, 20%, 30% and 40% of the RBSs have a backup link in the scenarios, respectively. (Note that above
these backup link values ring or mesh-like topologies would be more efficient to be applied instead of the extension of trees.)

Because of OWSA is deterministic, only one run is enough to obtain its result. However, the heuristic reference solutions are based on randomized procedure, which means that more than one run is required to obtain a typical, average result. Thus the reference algorithms were run 10 times for each network sizes and protection level.

5.5.1 Comparison of the Proposed and the Exact Algorithms

Since the problem can be formulated as an integer linear programming (ILP) task, let us compare the running time of OWSA and the LP solution given by the LP-solver package [60].

The solution of this task is exact, of course. In case of larger network, however, this solution cannot be applied, since extreme long time is needed to find this solution.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Number of nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50</td>
</tr>
<tr>
<td>OWSA</td>
<td>4.16 sec</td>
</tr>
<tr>
<td>ILP</td>
<td>≈ 1 hour</td>
</tr>
</tbody>
</table>

Table 5.1: Average running times at 40% protected nodes.

Table 5.1 shows that OWSA can solve large problem instances in approximately half a minute, however, ILP cannot reach solution even for 100 nodes in three days. Only an LP-relaxation of the problem can be solved, however, fraction numbers cannot be used as OSPF weights, which have to be integer numbers. The phrase "uncertain" in the table refers to these latter ILP/LP properties.

5.5.2 Comparison of the Proposed and the Heuristic Reference Algorithms

Table 5.1 backs up that heuristic solution have to applied in case of practical network sizes. In this section, the results of OWSA are compared with the two simulated annealing based Reference algorithms (RefSol-1, RefSol-2) and the Initial Weight Setting (IWS) procedure.

The methods mentioned above are compared with each other according to the percent of lost traffic (value of overload in the network), and to the percent of number of overloaded links. The results are summarized in Figure 5.6. The detailed results are shown in Tables 5.2.
Figure 5.6: Overload in the network

Figure 5.6(a) shows that procedure IWS does not solve the OSPF weight setting problem and the overload in the network increases as the number of backup links increases. When 11% of the links are backup link, the RefSol-1 can solve the problem for all network sizes, however, RefSol-2 can solve the problem in case of smaller networks (50 and 100 nodes). Moreover, in case of more backup links, the overload increases, but it still remains under 0.6%. This result is acceptable compared to procedure IWS, nevertheless, OWSA can always solve the problem.

Figure 5.6(b) shows that the number of overloaded links has a very similar behavior to the overload. In case of procedure IWS the number of overloaded links is mostly depends on the number of backup links. In case of the Reference solutions, the number of overloaded links slightly increases as the number of backup links increases. Moreover, the number of overloaded links is the multiple of the overload in the network. The reason of this is that the Reference solutions try to minimize the overload in the links, so they prefer such solutions, where the overload is shared between several links. Of course, in case of OWSA, there are no overloaded links in the network.

The above tests back up that OWSA can solve the OSPF weight setting problem (or at least keep the performance of the network on the same level as in the dedicated case if there are directed circles in the network).
<table>
<thead>
<tr>
<th>Protected RBs</th>
<th>Number of nodes</th>
<th>Average overload [%]</th>
<th>Average number of overloaded links [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CWSA</td>
<td>ReSol-1</td>
</tr>
<tr>
<td>10%</td>
<td>50</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>20%</td>
<td>50</td>
<td>√</td>
<td>0.099%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>√</td>
<td>0.077%</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>√</td>
<td>0.05%</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>√</td>
<td>0.085%</td>
</tr>
<tr>
<td>30%</td>
<td>50</td>
<td>√</td>
<td>0.114%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>√</td>
<td>0.107%</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>√</td>
<td>0.283%</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>√</td>
<td>0.19%</td>
</tr>
<tr>
<td>40%</td>
<td>50</td>
<td>√</td>
<td>0.151%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>√</td>
<td>0.13%</td>
</tr>
<tr>
<td></td>
<td>150</td>
<td>√</td>
<td>0.284%</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>√</td>
<td>0.42%</td>
</tr>
</tbody>
</table>

Table 5.2: Overload in the network (optimal solution without overload = √)

5.6 Conclusions

In this chapter, the analysis was focused on the configuration of OSPF weight system in protected UMTS access networks. The goal was to propose a weight setting strategy that makes the default path be the shortest path in the network in case of nominal operation and the backup paths to be the shortest path in the network in case of any single link failure. It is showed that the simple, greedy type weight setting fails to solve this task, so it results very significant overload and traffic loss, so it is not applicable. The known weight settings result measurably better network configuration, but some overload still remains. Therefore a more sophisticated method has been worked which (a) considers the special properties and structure of UMTS access networks, as well as (b) finds those part of the explicit path structure, which does not conform with the OSPF routing rule.

Based on some tests, it is showed that the proposed method is able to set the OSPF weight in a proper way in cases of different network topologies and different number of protected demands. Moreover, the results also show that the proposed method is able to solve the OSPF weight setting task in real UMTS/OSPF access networks and it is an efficient, useful tool in the network planning process.
Chapter 6

Summary

6.1 Scientific Results

1. Cost-optimal Planning of Interference-insensitive Hierarchical Access Networks (Chapter 2)

   - *An Algorithm for Finding a Valid Initial Solution*: I proposed a new heuristic algorithm based on clustering and graph algorithms. The algorithm determines the level of each node in the hierarchy and creates the connections between the nodes.

   - *An Enhanced Algorithm for Iterative Improvement of a Solution*: I proposed an enhanced variant of the algorithm presented in [31]. The algorithm is extended to efficiently handle any number of hierarchy levels and to systematically improve any valid solution by advancing in the hierarchy level by level. The enhancement includes new and improved operations (which are the building-blocks of the algorithm) and a new conception to control the complexity and effectiveness of the operations.

2. Cost-optimal Planning of Interference-sensitive Hierarchical Access Networks (Chapter 3)

   - *Settlement of point-to-point connections*: I proposed a new heuristic algorithm, which satisfies the constraints on the point-to-point connections in order to avoid the damaging interference. The algorithm minimizes the cost of the network by applying the methods of Chapter 2).

   - *Combined planning of point-to-point and point-to-multipoint connections*: I proposed new algorithms based on clustering, which combine the point-to-point and the point-to-multipoint connections with each other in a cost-effective way. In order to construct the point-to-point
connections and create the constraint-trees, the results of point-to-point planning and the algorithms Chapter 2) were applied, respectively.

3. Solution of the Weighted Fermat-problem and its Applications to Topology Optimization (Chapter 4)

- **A General Solution to the Weighted Fermat-problem:** I analyzed the geometrical properties of the point, which is the solution of the weighted Fermat-problem: a) I gave new conditions to decide when the weights of the nodes determines the position of the solution in advance, b) I gave a simple mathematical description of how to find the position of the solution in general and special cases, c) I showed how the solution point divides the area of the "original triangle" in special symmetrical cases, d) I showed the connection between the value of the objective function and the sides of the triangle.

- **Application of the Weighted Fermat-point to Topology Optimization:** I analyzed the possible application areas of the weighted Fermat-problem in telecommunications environment: a) connection between the weights and the capacity dependent cost of the traffic demands, b) connection between the weights and the multiplexing gain applied at merging of links. On the other hand, I analyzed the attainable gain in case of topology optimization: c) analysis of the efficient applicability of the solution against some parameters of the network, d) the analysis of the attainable gain provided by the solution (maximum, upper bound).

4. Optimization of OSPF administrative weights in access networks (Chapter 5)

- **Approximate heuristic solution:** After an initial weight setting, the proposed algorithm analyzes the network and finds those demands in case of each link failure, which demands are not routed in the predefined paths. Then it corrects these errors demand by demand, which means the modification of the weights in order to obtain the predefined path system.

6.2 Application of the Results

The algorithms presented in Chapter 2 (for the planning of interference-insensitive access networks) are used in the UMTS Network Planning and Analysis Tool of Ericsson Telecommunication Ltd.

The research work related to the interference-sensitive access networks (Chapter 3) and to the optimization of OSPF administrative weights (Chapter 5) was
carried out in the projects of Ericsson Research Hungary in cooperation with the Product Units of Ericsson.

The theoretical and applied research results presented in Chapter 4 can be widely used in the field of network planning and geometry-based optimization.

6.3 Short Outlook

The presented algorithms are planned to be applied for practical network planning support, especially for the challenging area of GSM to 3G/UMTS transition. A further research direction is to complete the presented approach with core network planning algorithms to provide an overall solution for the planning task.

Another future task is to apply more sophisticated model of interference-sensitivity, especially in case of constraints to avoid interference between equipment and links. Besides the practical use of "rule of thumb" constraints, some interference calculation based feedback can help the algorithms to find more accurate solutions.

As in the case of configurations with 3 nodes (Fermat-problem), faster or exact solutions are needed in case of configurations with at least 4 nodes (Weber-problem) in order to further improve the network. (E.g. symmetrical and regular polygon topology configurations are promising targets of research.)

Finally, in case of OSPF-related network configuration, the investigation of more complex and reliable topologies and backup path scenarios constitutes the basis of future tasks.
References


[34] B. Gavish: Topological design of computer communication networks - the overall design problem, European Journal of Operational Research, vol. 58, no. 2, pp. 149-172, 1992


**Publications**

**Journal papers**


92

Conference and workshop papers


Appendix to Chapter 2: Pseudo Codes of Planning Procedures

In this appendix, a pseudo code-like mathematical formulation of the planning procedures of Chapter 2 are presented. All the basic and advanced operations and the planning algorithms are detailed in the following few pages.

Procedure Cluster\((P, K_{\text{min}}, l)\)

1. \((\text{Initialize})\) Let \(C_{\text{best}} = \infty\), denoting the cost of the overall best configuration found. Let \(K = K_{\text{min}}\), where \(K_{\text{min}}\) is the minimal number of clusters to be formed from the set of input nodes \(P\).

2. \((\text{K-Loop})\) Initialize \(K\) median nodes \(M_j\) from set \(P\) on level \(l\) randomly, and let each cluster \(G_j\) contain the selected median \(M_j\) only, \(j = 1, \ldots, K\). Let \(C_K = \infty\), denoting the best cost found if using \(K\) medians.

3. \((\text{Inner-Loop})\) Iterate the following steps while \(C_K\) improves:

   (a) \textit{Allocation}: \(\forall u \in P\), connect \(u\) to a chosen median \(M_j\) in cluster \(G_j\), where \(M_j\) is the median of cluster \(G_j\), \(j = 1, \ldots, K\).

   (b) \textit{Relocation}: \(\forall j = 1, \ldots, K\), recompute the median \(M_j\) in cluster \(G_j\).

   (c) \textit{Cost-calculation}: Compute the cost \(C_{\text{act}}\) of the actual configuration. If \(C_{\text{act}} < C_K\), then let \(C_K = C_{\text{act}}\) and save this configuration as the best one found with \(K\) medians.

4. \((\text{Update})\) If \(C_K < C_{\text{best}}\), then \(C_{\text{best}} = C_K\) and store the found best configuration with \(K\) medians as the overall best configuration.

5. \((\text{Check})\) If \(C_{\text{best}}\) has not improved during the last \(K_{\text{ln}}\) number of steps (\(K_{\text{ln}}\) is a look-ahead parameter, set to a small value), then go to Step 6. Otherwise, let \(K = K + 1\). If \(K \leq |P| - F\), then go to Step 2. (\(F\) denotes the number of nodes that are forbidden on level \(l\)).

6. \((\text{End})\) Return the overall best configuration found.
 Procedure Tree($P, \text{root}, l$)

1. (Initialize) Into set $C$ we collect those already connected nodes of $P$, to which we can connect other nodes of $P$ without violating the topological constraints and without getting a network with infinite cost (due to equipment or link type violation). Initially let $C = \{\text{root}\}$. Let $U$ be the set of unconnected nodes of $P$, and initially let $U = P \setminus \{\text{root}\}$. Let $C_P$ denote the multiset containing punished parents, let it be initially empty.

2. (Loop) Select a set of candidate nodes $S$ from $U$ randomly, where $|S| = \min(|U|, q_{\text{samp}})$. Parameter $q_{\text{samp}}$ denotes the fixed (small) sample size.

3. (Find Best Parents) For all $u \in S$, find its possible best parent $v \in C$. That is, for all $u \in S$ select a parent $v$ from $C$ and denote it by $p_u$ for which the connection cost of $u \rightarrow p_u$ is the minimal possible. The connection has to fulfill the topological constraints (level, degree) and the fixed/forbidden node limitations. Let the best connection cost found for node $u \in S$ be denoted by $c_u$. If the actual cost of a connection between $u \in S$ and $v \in C$ during the search for the best connection cost appears to be $\infty$ (forbidden because of constraints or costs), then put connected node $v \in C$ to $C_P$.

4. (Select Best) Find $s \in S$ for which its best connection cost $c_s$ is minimal. If $c_s = \infty$, then go to Step 8.

5. (Connect) Connect node $s$ to its best parent $p_s$. Remove $s$ from $U$. If $s$ is not placed at the last tree level ($L_w$), then put $s$ to $C$.

6. (Punish Parents) If a node $v \in C_P$ has been punished $m_p$ times, then let $C = C \setminus \{v\}$. Parameter $m_p$ specifies the maximum number of greedy punishments and it is set to a rather small value.

7. (Terminate) If $U$ and $C$ are both not empty, then go to Step 2. Otherwise, if $U$ is empty, then go to Step 9.

8. (Failure) Return with notifying that the greedy tree construction has failed due to constraint or cost violation.

9. (Success) Return the configuration that has been created.

 Procedure Moves($C, P$)

1. (Initialize) The ordered set $C$ holds the candidate nodes, and $P$ holds the possible new parent nodes. Initialize variables $L_C$ and $L_P$ to the first element of $C$ and $P$, respectively. These working variables mark the candidate and the parent nodes participating in the last successful move.
2. *(Loop)* Let \( u \) be the first element of \( C \) and let \( v \) be the first element of \( P \).

3. *(Check & Move)* Check the possible move of node \( u \) to new parent \( v \):

   (a) Check the geometrical position of \( u \) and \( v \). If \( u \) and \( v \) are relatively far (see Section 2.3.4), go to Step 4.

   (b) Check constraints. If move \( u \rightarrow v \) is not possible (e.g. \( u = v \), or the degree/level constraints would be violated, etc.), go to Step 4.

   (c) Check the benefit of the movement. Calculate the cost-difference between the old and the new configuration after the move. If it would not be a successful move (i.e., the cost increases), go to Step 4.

   (d) Accept move. Store the new configuration. Let \( L_C = u \) and \( L_P = v \).

4. *(Next Parent)* If \( v \) is not the last element of \( P \), advance \( v \) to the next element of \( P \) and go to Step 6. Otherwise let \( v \) be the first element of \( P \).

5. *(Next Candidate)* If \( u \) is not the last element of \( C \), then advance \( u \) to the next element of \( C \). Otherwise let \( u \) be the first element of \( C \).

6. *(Terminate)* If \( u = L_C \) and \( v = L_P \), then stop and return the final configuration. Otherwise continue with Step 3.

**Procedure Moves, cost-difference calculation in Step 3c**

1. *(Initialize)* Let \( s \) denote the original parent of \( u \), where \( u \) is the examined node and \( v \) is the possible new parent. Let \( B_{old} \) denote the set of nodes above node \( u \). Let \( B_{new} \) contain node \( v \) and the set of nodes above \( v \). Let \( B_{under} \) contain node \( u \) and the set of nodes in the tree-segment under \( u \). 'Above' and 'under' here means that there exists a path from the node upwards or downwards in its tree to the particular node, respectively.

2. *(Original Cost)* Calculate the cost of the links and equipment in \( B_{old} \) and \( B_{new} \) including the cost of the outgoing link from \( u \). Let the sum of them be denoted by \( C_{orig} \). If \( l_v \neq l_u \), then add the cost of elements in \( B_{under} \) to \( C_{orig} \). (Refer to this case as Level-Change.)

3. *(New connection)* Delete the connection from \( u \) to \( s \) and add a new connection from \( u \) to \( v \). Recalculate the amount of traffic in \( B_{old} \) (decrement) and in \( B_{new} \) (increment). If Level-Change, modify the levels of the nodes in \( B_{under} \) according to the level-difference between the "parents".

4. *(New Cost)* Calculate the new cost \( C_{new} \) according to Step 2.

5. *(Check & Accept)* If \( C_{orig} > C_{new} \), accept the new configuration and Stop.
6. *(Failure)* Restore the original connection between $u$ and $s$. Restore the traffic values. If *Level-Change*, restore the original levels in $B_{\text{under}}$.

**Procedure Swaps($C$)**

1. *(Initialize)* Let $L_1$ and $L_2$ be the first and second element of $C$, respectively. These working variables mark the candidate node pair participating in the last successful swap.

2. *(Loop)* Let $u$ be the first and $v$ be the second element of $C$.

3. *(Check & Swap)* Check the possible swapping of node $u$ and $v$:
   
   (a) Check the geometrical position of $u$ and $v$. If $u$ and $v$ are relatively far (see Section 2.3.4), go to Step 4.

   (b) Check constraints. If swap $u \leftrightarrow v$ is not possible (e.g., $u = v$, or the degree/level constraints would be violated, etc.), go to Step 4.

   (c) Check the benefit of the swap operation. Calculate the cost-difference between the old and the new configuration after the swap. If it would not be a successful swap (i.e., the cost increases), go to Step 4.

   (d) Accept swap. Store the new configuration. Let $L_1 = u$, $L_2 = v$.

4. *(Next-$j$)* If $v$ is not the last element of $C$, then advance $v$ to the next element of $C$ and go to Step 6. Otherwise let $v$ be the first element of $C$.

5. *(Next-$i$)* If $u$ is not the last element of $C$, then advance $u$ to the next element of $C$. Otherwise let $u$ be the first element of $C$.

6. *(Terminate)* If $u = L_1$ and $v = L_2$, then stop and return the final configuration. Otherwise continue with Step 3.

**Procedure Recluster($r, m_{\text{alloc}}, m_{\text{locopt}}$)**

1. *(Select Nodes)* Build up node set $P$ to be clustered according to $r$:
   
   - If $r$ denotes an existing node ($r \in V$), put all the child nodes of the given input node $r$ ($\forall u \in V, l_u = l_r + 1, x_{ur} = 1$) and also the child nodes of these child nodes ($\forall u \in V, l_u = l_r + 2, \exists v \in V : x_{uv} = 1 \land x_{vr} = 1$) into $P$. Let the clustering level be $l = l_r + 1$.
   
   - Otherwise, if $r = 0 \notin V$ (meaning that the clustering of top-level nodes is required), then put all nodes of level $0$ into $P$ (root nodes) and put also the child nodes of these root nodes (i.e., all the nodes of level $1$) into $P$. Let $l = 0$. 

97
2. (Clear Connections) Delete all interconnections between the nodes in set \( P \) and all the connections between node \( r \) (if exists) and the elements of \( P \) from the actual solution before the recluster operation.

3. (Calculate \( K_{\text{min}} \)) Set \( K_{\text{min}} \) according to the number of fixed nodes for level \( l \) and the minimum number of necessary median nodes resulting from the degree and level constraints.

4. Perform Cluster\((P, K_{\text{min}}, l)\) with internal local optimization.

Recluster applies the following sub-procedures in Step 3 of Cluster:

(a) (Allocation) We have two types of allocation methods, which work in a greedy fashion, and the allocation-decision depends on the cost of the next connection. The allocation type parameter \( m_{\text{alloc}} \) decides which method is used, the first is quicker, the second is slower due to its nested loop.

- \((\text{Greedy-1})\) \( m_{\text{alloc}} = 1 \): In each iteration step, for a randomly chosen non-allocated node \( u \in P \), find the median \( M_j \), among the possible medians, where the allocation of \( u \) to \( M_j \) induces the smallest additional cost in the solution. Connect \( u \) to \( M_j \), and repeat this \( \forall u \in P \).

- \((\text{Greedy-2})\) \( m_{\text{alloc}} = 2 \): In each iteration step, instead of choosing a random non-allocated node, calculate the greedy allocation cost described in method Greedy-1 for all non-allocated nodes \( u \in P \), without making any connection. Then choose the least cost allocation possibility among all nodes and perform it. Repeat until all nodes will be allocated to the medians.

(b) (Relocation) For all cluster \( G_j \) sequentially, find its new center \( M_j \) by simple testing of all candidates: \( \forall u \in G_j \), try \( M_j = u \), connect all other nodes in \( G_j \) to \( i \) and calculate the cost of this new configuration. Choose the least cost configuration and set \( M_j \) accordingly.

(c) (Cost Calculation) Calculate the cost of the resulted network configuration according to 2.1 (see Problem Statement). Note that only the cost of the affected elements must be recalculated in the clustering.

Application of Moves and Swaps within Recluster is controlled by parameter \( m_{\text{locopt}} \):

- \((\text{Inner-Loop})\) \( m_{\text{locopt}} = 3 \): Local optimization is done within each Step 3 of Cluster, after each allocation and relocation sub steps, i.e., when we search for the optimal configuration with \( K \) medians. Here let \( C = \bigcup_j (G_j \setminus \{M_j\}) \) and \( P = \bigcup_j \{M_j\} \). Perform Moves\((C, P)\), then perform Swaps\((C)\).
• (Best With K) $m^{\text{loopt}} = 2$: In this case we perform local optimization only after Step 3 (before storing the best configuration with K centers in Step 4) of Cluster procedure, i.e., when we change the number of medians. Procedures Moves and Swaps are performed in the same way as above.

• (At End) $m^{\text{loopt}} = 1$: In this case, local optimization by performing operation Moves and then Swaps with the same parameters as above is done only at the termination of the Cluster procedure, before Step 6, for the overall best configuration found.

Procedure ReclusterLevel$(l, m^{\text{alloc}}, m^{\text{loopt}})$

1. (Case1) $0 \leq l \leq L_w - 2$: Perform Recluster$(u, m^{\text{alloc}}, m^{\text{loopt}})$ for all $u \in V : l_u = l$.

2. (Case2) $l = -1$: Perform Recluster$(0, m^{\text{alloc}}, m^{\text{loopt}})$ in order to recluster the top-level nodes.

Procedure LocalOpt$(l, ByRoots)$

1. (Initialize) If $ByRoots = \text{FALSE}$, then let $P_1 = \mathcal{N}$ and $k = 1$. Otherwise let $P_j = \{u \in V : u$ is controlled by root $j\}, j = 1, \ldots, k$, where $k$ denotes the number of root (level 0) nodes.

2. (Loop) Perform the following two steps for all $P_j, j = 1, \ldots, k$:

3. (Determine Sets) Construct $\mathcal{C}$ and $\mathcal{P}$ depending on parameter $l$:
   • (Case1) $0 \leq l < L_w$: $\mathcal{C} = \{u \in P_j | l_u = l + 1\}$. $\mathcal{P} = \{u \in P_j | l_u = l\}$.
   • (Case2) $l = L_w$, that is, we choose the leaf nodes.
     Let $\mathcal{C} = \{u \in P_j : l_u = L_u \lor \sum_{v \in P_j} x_{vu} = 0\}$. Let $\mathcal{P} = P_j$.
   • (Case3) $l = L_w + 1$: All node pairs. Let $\mathcal{C} = \mathcal{P} = P_j$.
   • (Case4) $l = L_w + 2$: This case is the same as Case3, but here we skip the checking of the geometrical position of the selected node pairs in the Moves and Swaps procedures (cf. Steps 3 of them).

4. (Local Optimization) Perform Moves($\mathcal{C}, \mathcal{P}$) and then Swaps($\mathcal{C}$).

Procedure Top-Down-IniSol$(V)$

1. (Initialize) Let $l = 0$ denote the actual level of the network to be constructed.
   Let $\mathcal{G}$ denote the set of clusters to be processed and initially let $\mathcal{G} = \{V\}$.
   Let $\mathcal{H}$ denote a set of clusters and initially let $\mathcal{H} = \emptyset$. 

99
2. \textit{(Loop)} In order to assign nodes to level $l$, $\forall G_i \in \mathcal{G}$ do:

(a) \textit{(Calculate $K_{\text{min}}$)} Calculate $K_{\text{min}}$ depending on the number of nodes in $G_i$, number of fixed nodes at level $l$ in $G_i$, and the degree and level constraints.

(b) \textit{(Clustering)} Perform $\text{Cluster}(G_i, K_{\text{min}}, l)$. Put the resulting clusters to $\mathcal{H}$. If $l > 0$, connect all new medians $M_j$ (found by $\text{Cluster}$ in $G_i$) to the median of cluster $G_i$ that was found at the previous level.

3. \textit{(Next Level)} Let $\mathcal{G} = \mathcal{H}$ and let $\mathcal{H} = \emptyset$. Let $l = l + 1$. If $l \leq L_w$, then go to Step 2, otherwise Stop.

\textit{Top-Down-IniSol} applies the following sub-procedures in Step 3 of $\text{Cluster}$:

(a) \textit{(Allocation)} As the exact costs of allocations are not known, the allocation decision is done on the shortest distance basis: $\forall u \in P$ find the median $M_j$ ($j = 1, \ldots, K$) that is the closest to node $u$, i.e. $d_{uM_j}$ is minimal (see equation (2.4)). Put $u$ to $G_j$, where $G_j$ is the $j$-th cluster.

(b) \textit{(Relocation)} Again, as the exact costs of the clusters are not known, the cluster center relocation is done on a geometrical basis. $\forall j = 1, \ldots, K$ find a new median $M_j$ in cluster $G_j$: calculate $\text{Geo}_j$ denoting the geometrical center of $G_j$, by averaging the $(X, Y)$ coordinates of the nodes in $G_j$. Then find $u \in G_j$ that is the closest to $\text{Geo}_j$, and let new median $M_j = i$.

(c) \textit{(Cost-calculation)} $\forall j$ set $M_j$ to be in level $l$ and perform $\text{Tree}(G_j, M_j, l)$.

If $l > 0$, then temporarily connect all $M_j$ to the corresponding median in level $l - 1$. Finally calculate the cost of the resulted network configuration according to equation (2.1) (see Problem Statement).

\textbf{Procedure Random-IniSol($V$)}

1. \textit{(Initialize)} Select a random node $v \in V$ for being at level 0, let the set of connected nodes $\mathcal{C} = \{v\}$, and let the set of unconnected nodes $\mathcal{U} = V \setminus \mathcal{C}$. Let the multiset of punished nodes $\mathcal{C}_p$ be empty.

2. \textit{(Loop)} Select a random $u \in \mathcal{U}$. Let the number of trials $q = 0$.

3. \textit{(Parent Search)} While $q$ does not exceed $2 \cdot |\mathcal{C}|$ repeat the following: Select a random parent $s \in \mathcal{C}$ and let $q = q + 1$. Try to connect node $u$ to $s$. If the resulting cost is valid and the connection does not violate any constraint, then perform the connection, take out node $s$ from $\mathcal{C}$ if it reached its degree constraint, and go to Step 5. Otherwise, if connecting $u$ to $s$ is not valid, punish node $s$: put $s$ to $\mathcal{C}_p$. If the total number of punishments for node $s$ exceeds a (small) limit, take out $s$ from $\mathcal{C}$.
4. *(New Root)* If no $s$ satisfying the above conditions was found, let node $u$ be at level 0. (If $u$ is forbidden at level 0, choose another random $u \in U$, which is allowed to be at level 0.)

5. *(End)* Take out $u$ from $U$. Put $u$ to $C$ if it is not at the lowest allowed level. If $U$ is not empty, then go to Step 2.

**Procedure Full-Iteration-Improvement**($V, UseClo, UseLoc$)

1. *(Initialize)* Let $C_{act}$, $C_{best}$ denote the actual cost, the best cost found respectively, and let both be initially equal to the cost of the network planned by the initial solution. Let $A = 1$ assign a column of Table 6.1.

2. *(Advance Loop)* Set the parameters of ReclusterLevel and LocalOpt according to the column $A$ of Table 6.1. In the row of parameter ByRoots, 1 denotes TRUE and 0 denotes FALSE.

3. *(Level Loop)* Sequentially for each level $l = 0, \ldots, L_w$ do:
   
   (a) *(ReclusterLevel)* If $UseClo \land l < L_w$, then perform ReclusterLevel($l - 1, m^{alloc}, m^{locopt}$).
   
   (b) *(LocalOpt)* If $UseLoc \land A > 7$, then perform LocalOpt($l, ByRoots$).

4. *(Extra LocalOpt 1)* Calculate $C_{act}$. If $C_{act} > C_{best} (1 - I) \land UseLoc$, perform LocalOpt($L_w + 1, ByRoots$) and calculate $C_{act}$. $I \in (0, 1) \ll 1$ is a minimal improvement threshold parameter.

5. *(Check Cost)* If $C_{act} < C_{best} (1 - I)$, let $C_{best} = C_{act}$ and go to Step 3.

6. *(Extra LocalOpt 2)* If $A > 9 \land UseLoc$, perform LocalOpt($L_w + 2, FALSE$). Recalculate $C_{act}$.

7. *(Termination)* If $A \leq 9$, let $A = A + 1$. If $C_{act} < C_{best} \lor A \leq 9$, let $C_{best} = C_{act}$ and go to Step 2. Otherwise stop, return the actual solution.

The parameters are changing according to the scheme presented in Table 6.1.

<table>
<thead>
<tr>
<th>Advance Loop</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m^{alloc}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$m^{locopt}$</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>ByRoots</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.1: Parameter-table of procedure Improvement