OPTIMIZATION METHODS
FOR VIRTUAL PRIVATE NETWORK DESIGN

Ph.D. Dissertation

By

Markosz Maliosz

Research Supervisor:

Tibor Cinkler Ph.D.

Department of Telecommunications and Media Informatics

Budapest University of Technology and Economics

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
AT
BUDAPEST UNIVERSITY OF
TECHNOLOGY AND ECONOMICS
BUDAPEST, HUNGARY
OCTOBER 2005

BUDAPEST UNIVERSITY OF
TECHNOLOGY AND ECONOMICS

Date: **October 2005**

Author: **Markosz Maliosz**

Title: **Optimization Methods for Virtual Private Network Design**

Department: **Telecommunications and Media Informamatics**

Degree: **Ph.D.**

The reviews of this Ph.D. dissertation and the minutes of the Ph.D. discussion are available in the Dean's Office.

Permission is herewith granted to Budapest University of Technology and Economics to circulate and to have copied for non-commercial purposes, at its discretion, the above title upon the request of individuals or institutions.

_____

Signature of Author

Table of Contents

List of Tables

List of Figures

Abstract

The optimal route configuration of Virtual Private Networks (VPNs) over a given network is important regarding the costs of the VPNs. The goal is the configuration of bandwidth guaranteed connections between the endpoints of the VPNs. The VPNs are fully connected, i.e. full mesh demand sets (forming the VPNs) have to be built between the endpoints of the VPNs. The service demands of VPNs are characterized by the bandwidth requirements of node-pairs (pipe-model). Given the capacity matrix of the physical network and the traffic demand matrices of the VPNs, the configuration of multiple VPNs is searched which minimizes the costs of the network. The problem is formulated as a general model without specializing to any particular network type, however the proposed methods can be used for any network that supports resource partitioning, i.e. the setup of bandwidth guaranteed tunnels, such as SDH, ATM, MPLS and WR-DWDM Networks.

The two objectives of the optimization are minimizing the reserved bandwidth and minimizing the topology, i.e. the number of links used by the VPNs. Both objectives correspond to a particular cost for the VPN customer. These two objectives can produce opposite effects, but it is known that network design is always a trade-off.

VPN users also require reliable connections therefore protection schemes are used to provide service guarantees. Path based pro-active dedicated and shared protection methods are investigated for the protected VPN configuration. By the protected VPN configuration a group of disjoint paths are sought that also minimizes the capacity reservation and the topology of the VPNs.

To calculate the global optimum an exact method, namely the integer linear programming (ILP) was used. Integer linear programs were formulated for the different problems classes and ILP solver software (ILOG CPLEX 9.0 [cpl]) was applied to solve them. Solvers need to find the optimal state in a huge state space, which could take very long time.

However, if we are satisfied with a suboptimal solution, the running time can be reduced. One method, that was applied, is to decompose the global problem into smaller integer linear programming problems and then solve them separately. Another method was to apply heuristic algorithms based on shortest path algorithms. The running time of the methods is another trade-off between the optimality of the solution and the required time.

In this dissertation integer linear programming problems are formulated, decomposition of global problems are described, and heuristic algorithms are presented for the optimal bandwidth guaranteed VPN configuration without and with protection in capacitated networks, where multiple VPNs co-exist. Performance metrics are defined to evaluate the capacity and topology objectives. The parameter settings of the different methods are analyzed to get the best solution. The methods are evaluated by simulation on test reference networks with generated VPN traffic patterns. The results of the different methods are investigated and compared. The effectiveness of the heuristic methods are compared with the integer linear programming.

Acknowledgements

I would like to express my gratitude for all people who have supported my research. I thank to my supervisor Dr. Tibor Cinkler for the guidance and consultations.

I thank the continuous support from the High Speed Network Laboratory especially from our research lab leader Dr. Tamás Henk.

Also undergraduate and Ph.D. students that I worked with played very important role in my research. I am thankful especially to Péter Hegyi and Ákos Ladányi.

I am grateful to Dr. István Cselényi, who assisted me towards my Ph.D. studies and to Krisztián Németh, who encouraged the submission of the dissertation. I thank to my parents for the support during my studies.

Chapter 1

Introduction

Virtual Private Networks are networks over networks: they seem to be real private networks for their users, but actually they are realized over the network of one or more providers. Virtual networks are basically logical overlays on a physical network. The broadest definition of a VPN is “any network built upon a public network and partitioned for use by individual customers” [SOL]. A VPN consists of several geographically separated sites that are connected over (possibly multiple) service provider networks. VPNs support the communication requirements of a closed group of users with special handling of privacy and security. Encryption and other security mechanisms are used to ensure that only authorized users can access the network and that the data cannot be intercepted. Typical application of VPNs is remote access for joint project workers, or for a home user to log on to the company intranet.

There are various types of VPNs [KL04] according to the International Standards Organization Open System Interconnection (ISO - OSI) reference model [fS94].

- Layer 1 (physical layer) VPN is a virtual private network made up of leased circuits connecting customer sites. Examples of layer 1 connections are optical or time division multiplexing (TDM) paths. For realizing Layer 1 VPN functionality GMPLS (Generalized Multi Protocol Label Switching) [Man04] mechanisms can be used [TP05].

- Layer 2 (data link layer) VPN supplies a layer 2 point-to-point service or emulates LAN (Local Area Network) service across a WAN (Wide Area Network) [AR04]. Examples for layer 2 VPNs are frame relay, X.25, and ATM (Asynchronous Transfer Mode) networks.
- Layer 3 (network layer) VPNs are implemented with point-to-point IP-over-IP (Internet Protocol) tunnels constructed across shared IP backbones, referred to as "IP VPNs" [GLH+00, CS05]. Examples for tunneling are the IPSec (Secure Internet Protocol) [KA98] or GRE (Generic Routing Encapsulation) [FLH+00] protocol frameworks.

Thus, the term VPN is used in different networks (ATM, Frame Relay, MPLS (Multi Protocol Label Switching) [RR99, RVC01], etc.) and by tunneling protocols too (GRE, IPsec, PPTP (Point to Point Tunneling Protocol) [HPV+99], L2TP (Layer 2 Tunneling Protocol) [TVR+99], etc.).

To provide privacy and guaranteed QoS (Quality of Service) between sites the VPN solution over public network infrastructure gives a cheaper and more flexible alternative compared to leased lines, therefore VPN services have become more popular recently. The protected VPN service is a significant service for companies, because it is a low-cost, easy-to-implement, and convenient solution for remote access and site-to-site networking. The outsourced management of the VPNs is an important income for Internet Service Providers (ISPs). Encryption and security is handled by the upper communication layers, I do not deal with these aspects in this work. However, the topological design of the VPNs is an important issue regarding the costs. In a VPN all sites are interconnected (full mesh) with given bandwidth requirements. The task is to design VPNs over the network of a service provider, i.e. route the demands among VPN sites, which yields the layout of the VPNs.

In the model the total cost consist of *administrative cost* and *bandwidth proportional* cost. The administrative cost is proportional with the numbers of used edges, i.e. the topology of the VPN. If the number of used edges – i.e. the administrative cost – is minimized then the traffic is concentrated on few edges, the topology will be compacted.

To provide network reliability the design must be prepared for failures. The proactive path-based dedicated and shared protection scheme was investigated. The protection scheme is path based, since it assigns a second end-to-end backup path to each demand besides the primary working path. These protection paths are calculated in advance (proactive), not on-line when a failure occurs. Dedicated protection provides a disjoint backup path with the same capacity as the primary. Shared protection assumes that in a given time only one failure is present in the network, therefore the protection capacity can be shared among all protection paths that do not have any common element in their working paths. This common element can be either a link or a node, therefore we distinguish link and node disjoint protection. In contrast, dedicated protection does not allow sharing the capacities of protection paths. It ensures protection for all failures affecting the working paths, but requires more bandwidth to reserve. With shared protection on a particular link it is enough to reserve the capacity of the largest demand protected on that link, if the working paths do not have common links. Of course this becomes more complex if several protection paths use a link and their corresponding working paths have common links. In that case the sharing is not applicable for the considered link for these demands. Shared protection can tolerate multiple failures, if the protection paths do not interfere, but this is not guaranteed.

The rest of the dissertation is organized as follows.

Chapter 2 describes the optimization model. It includes the detailed objective definition and the review of the related work in the literature, as well as the network model and its notation. The selected test networks and the traffic generation methods are also presented together with the usual capacity metrics and the newly introduced topology metrics.

Chapter 3 describes the VPN design problems without protection. Integer linear programming problems are formulated for splittable and unsplittable flows. The decomposition of the global integer linear program and path based heuristic algorithms are also presented and evaluated.

Chapter 4 contains the VPN design problems with protection. Integer linear programming problems are formulated for link and node disjoint protection at the level of paths, and for dedicated protection at the level of VPNs. Heuristic algorithms are given for path

level link disjoint dedicated and shared protection.

In Chapter 5 the application possibilities are outlined in such networks that have resources to be partitioned, such as links or nodes.

Appendix A depicts the topology of the test networks on which the simulations were run.

Chapter 2

Optimization Model

2.1 Objectives

By VPN design or configuration the route selection and the reservation of resources is meant, where the following conditions are assumed:

- bandwidth demands are static
- the network is capacitated
- multiple VPNs exist within a single physical network

The bandwidth demands between the VPN endpoints are static, which is a proper assumption because creating a VPN is a long-term decision, and the "holding time" of the VPN can be several months or years, during this period the bandwidth requirements are fixed and not changing. The network is capacitated, i.e. capacity bounds limit the shortest path connectivity between the VPN endpoints. In the common physical network multiple VPNs co-exist and the joint optimal configuration is the goal of the design. A given endpoint can be part of more than one VPN, i.e. VPNs could overlap.

I introduced the concept of virtual links. Those links are the virtual links of a VPN that carry traffic for a given VPN, i.e. the links that are part of the VPN paths. Thus, a physical link can include several virtual links. All demands of a given VPN are routed over the virtual links of that VPN. These virtual links form a "skeleton" for the VPN. The paths of

the VPN can go only on this skeleton.

The cost model incorporates the following two objectives:

- Topology objective: minimize the number of virtual links
- Capacity objective: minimize the bandwidth reservation

The topology objective is to minimize the number of virtual links, i.e. the topological dispersion of the VPN layout. The goal is to minimize the number of virtual links, thus to force the VPNs not to spread over large areas, i.e. to use lower number of virtual links, and therefore lower number of physical links as well. It makes the VPN topology compact and simple and also simplifies the routing. With this compact topology, on a particular link fewer VPNs carry traffic, therefore the VPN administration, management and maintenance is also less burden. By minimizing the number of virtual links the number of nodes used by a VPN is also minimized.

This objective makes sure that certain VPNs will use as few virtual routers as possible as well, that simplifies the routing, the information flooding, as well as the management and maintenance of these virtual routers.

The topology objective results in the following topologies:

- in an uncapacitated network: tree
- in a capacitated network:
  - tree, if capacity bounds are not restrictive
  - non-tree, if capacity bounds are restrictive

The capacity objective minimizes the bandwidth reservation, therefore it results in a spread topology, i.e. each demand is routed on the shortest possible path reserving the least possible bandwidth. With this objective it is not considered which demand belongs to which VPN, therefore demands are individually routed, as if there were no VPNs.

To create the objective function cost functions are assigned to the topology and capacity objectives. Definition of the combined objective function of VPN optimization is the following:

Cost(VPN) = Cost(number of virtual links) + Cost(bandwidth reservation)

The combined objective results in compacting the topology and minimizing bandwidth usage jointly.

The following figures illustrate the effects of the different objectives. Figure 2.1 **a** shows an empty network with four nodes and five links, each link with 20 unit free capacity. One VPN will be set up, all four nodes are part of the VPN. Between each node pair (A–B, A–C, A–D, B–C, B–D, C–D) the bandwidth requirement is 5 units.

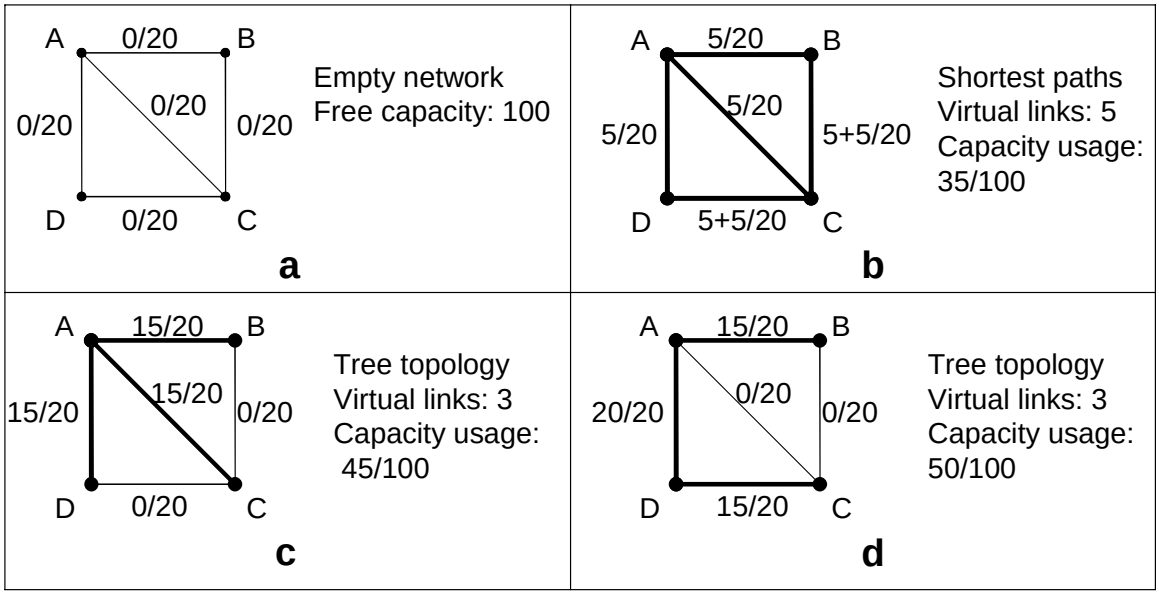


Figure 2.1: Illustration of Different Objectives

Figure 2.1 **b** shows the designed VPN according to the capacity objective. All demands are routed on the shortest path, i.e. on the direct links between the nodes except for B–D. Between B–D there is no direct link, therefore the route is B–C–D (it could be also B–A–D, but this route is equivalent in length and in capacity reservation with B–C–D). The thick lines show the virtual links of the VPN. As in the figure can be seen all five edges are used

by the VPN (therefore the VPN topology is not compact) and the capacity reservation is 35 units.

Figures 2.1 **c** and **d** show two VPN topologies designed according to the topology objective. The layout of the VPNs consists of 3 links, this is the minimal number of links that the 4 nodes can be connected over. Both topologies are trees, therefore the routes between the nodes is unambiguous. While according to the number of virtual links the two VPNs are equivalent, the reserved capacity differs. Figure 2.1 **c** shows the result of the joint topology and capacity objectives.

If the capacity of the links does not allow to build a tree topology (capacity bounds are restrictive), the topology objective still ensures to use minimum number of virtual links. If the bandwidth requirement between B and C is raised to 15 units Figures 2.2 **a** and **b** show the VPNs that can be built. Because the B–C link is full an additional link has to be used. Thus, both VPN topologies use 4 virtual links, therefore the topology is non-tree. The VPNs do not differ regarding the capacity reservation, therefore both VPN layouts correspond to the joint topology and capacity objectives.

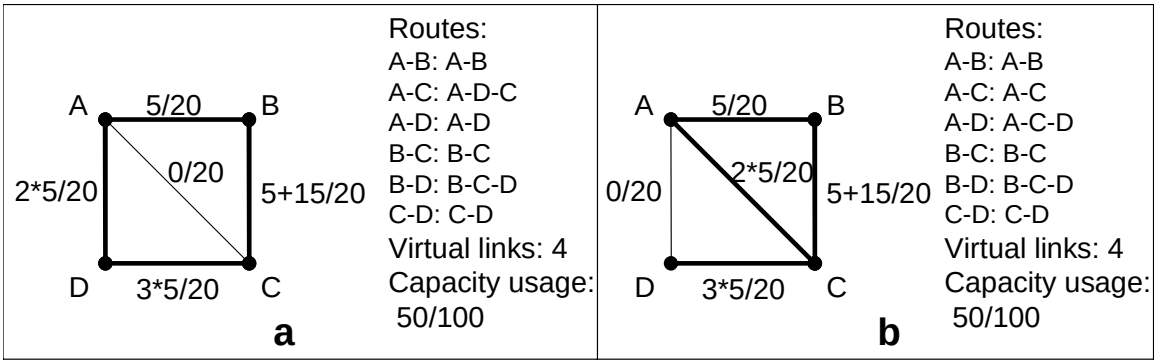


Figure 2.2: Topologies with Restrictive Capacity Bounds

2.2 Related Work

To minimize the bandwidth reservation regardless which demand belongs to which VPN is the same as routing multiple individual demands in the network. This is the classical

minimum cost multicommodity flow problem. Linear programming will suffice for multicommodity flow if fractional (splittable) flows are permitted, however multicommodity integral (unsplittable) flow is NP-complete, even with only two commodities. This problem was extensively studied with emphasis on the approximation algorithms [Rad95, LMP+91, KARR90, Min89, LR88, Hu63, QLLY02].

The VPN topology in most implementations has only two choices: fully-meshed or hub-and-spoke (star). The fully meshed is based on the result of the above-mentioned individual demand routing, i.e. it realizes the capacity objective. The hub-and-spoke topology is a tree, with one node – the hub – as the center, and the others as spokes, i.e. it realizes the topology objective if the capacity constraints permit it. In hub-and-spoke topology only one tunnel is needed at the spokes towards the hub.

Finding a VPN tree is the problem of finding the Steiner minimum tree (SMT). The SMT problem is well-known: the goal is to connect a subset of vertices in the graph by finding a minimum-weight tree that can also use any of the remaining vertices. SMT is NP-complete [GJ79], and thus, so is the optimal VPN topology problem described here.

In contrast with the path calculation in capacitated networks, VPN topology design is not so widely studied [LGN+01, KEKAP02]. Concentrating on dynamic VPNs [IL01] gives a design process which also supports static VPNs. The VPN topology is called VPN *shape* that can be a tree or a cyclic topology (mesh). The objectives are expressed as VPN service provider policies which can also include protection. The topology search is formulated as an optimization problem for which heuristic algorithms are proposed that provide close-to-optimum solution quickly, because the primary goal is the fast provisioning of dynamicly created VPNs. The proposed algorithms concentrate on finding tree topology, for cyclic topology only brute-force search, improved with some simple heuristics is proposed. The algorithms are applied to each created VPN, therefore do not provide global optimum for all VPNs in the network.

The design of multiple VPNs over MPLS, using sink-tree routing is presented in [ST02]. The problem is formulated as a mixed integer programming problem to find simultaneously VPN logical topologies and also a heuristic is proposed.

The problem of determining a layout of VPN tunnels (i.e. the topology of the VPN) is presented in [CK00]. The paper takes into account two factors: the cost of the links over which VPN tunnels are established, and the cost of the core routers (nodes) that serve as end points for the tunnels. Another paper [CLL04] addresses the problem of minimizing VPN tunnels in a weighted VPN network topology, taking into account two constraints: the first on the tunnel path length, and the second on the number of tunnels that can be go through a VPN node. By these means the minimization of the topology is targeted in both papers, however, the capacity of the links and nodes is not limited in the problem formulations.

Path protection techniques will be used in the protected VPN design. Path protection requires to find disjoint paths in the network between sources and destinations. Minimum length (cost) disjoint paths have different variations. The Min-Min problem is to find a link (or node) disjoint path pair such that the length of the *shorter path* is minimized. The Min-Sum problem is to find two disjoint paths such that the *sum of their lengths* is minimized. The Min-Max problem is to find two disjoint paths such that the length of the *longer path* is minimized. To solve the Min-Sum problem for dedicated protection there exists the polynomial time Suurballe algorithm [ST84], however, in the case of so-called shared path protection, no such polynomial time algorithm exists today which can solve the Min-Sum problem optimally. The Min-Min problem [XCX+04] and the Min-Max problem [LMSL90] is NP-complete. Path protection algorithms are studied widely [KL03] especially for optical networks [AQ00, DG01, HH04] together with wavelength assignment problem [ZOM03].

In this dissertation I give an intermediate alternative that does not restrict the topology to be a tree, while minimizing both topology and capacity objectives. The new results of this dissertation are from the joint investigation of capacity and topology objectives in the multiple overlapping VPN design aiming at global optimum without and with protection. Both exact and heuristic methods are defined and evaluated.

2.3 Network Model

The network is modelled as a graph. The network nodes are represented by vertices and the network links are represented by edges. Each edge has an assigned value: its capacity. The notations are as follows:

network graph	$G(V, E)$
set of vertices (nodes)	V
set of edges (links)	E
capacity of an edge	$b_e \in \mathbb{R}^+; e \in E$

The endpoints of the VPNs are subsets of the network nodes. These endpoints are the access points to the VPN for a VPN member, also known as *edge nodes*. All other interior nodes in the network that are part of a VPN, but are not directly connected to VPN endpoints, are called *core nodes*. In the model all network nodes are potential VPN endpoints representing a host or a subnet behind it, i.e. any network node can be an edge node.

The VPNs are defined with the traffic matrix that describes the bandwidth requirement between the endpoints of the VPN.

The notations are as follows:

set of VPNs	P	
set of endpoints in a VPN	$N_p \subset V; p \in P$	
traffic demands in a VPN	$d^p_{(i,j)}; \forall i, j \in N_p, i \neq j; p \in P$	(in short: d)
bandwidth of demand	b_d	
set of traffic demands	$D = \{d^p_{(i,j)}\}$	

2.4 Input Data for the Design Process

The same test networks and VPN traffic matrix sets were used throughout all calculations and simulations to compare the features and performance of different methods.

2.4.1 Networks

Three backbone networks have been chosen that are based on real network topologies. One network from the USA and two networks from Europe have been selected for the tests. NSFNet is a wide-area general purpose backbone network developed under the auspices of the National Science Foundation (NSF) in the USA. The T1 lines non-muxed NSFNet backbone was taken from 1989 [nsf].

From the COST (European Co-Operation in the field of Scientific and Technical Research) Action 266 – Advanced Infrastructure for Photonic Networks – report the Core Topology was taken [IKM03]. A simplified version of the GÉANT Network was taken as the third network topology. GÉANT is a pan-European multi-gigabit data communications network [gea]. The drawings of the topologies can be found in the Appendix.

The links of NSFNet and COST266 core network are of equal size, according to T1 line speed 1536 kbps. The GÉANT network contains 10 Mbps and 2.5 Mbps links.

Network	Nodes	Edges	Average edge degree	Diameter
NSFNet	13	19	2.923	3
COST266 core	16	23	2.875	5
GÉANT	18	30	3.333	5

Table 2.1: Selected Networks

Because the ILP calculation complexity can be high even with these small networks, to compare all design methods with each other such relative small (less than 20 nodes) networks had to be chosen. Of course, heuristic methods are able to give results in acceptable time for much larger networks. The examinations for larger networks with the heuristic algorithms can be found in my publications: for 80 nodes in [**C8**, **C9**], for 20, 50, 100, 200 nodes in [**W1**], for 28, 80, 137 nodes in [**J1**]. These papers show that the results of the heuristics algorithms are similar, independently of the network size. However, my goal was to point out the differences in the results of different methods, thus it was required to get results from the ILP solver in acceptable time.

2.4.2 Traffic

The task of VPN traffic generation is to:

- select the number of VPNs
- select VPN size, i.e. number of endpoints
- select the endpoints in the network
- select the bandwidth between each endpoint pair

The number of VPNs usually does not depend on the network size, therefore three categories were created: 5, 10 and 15 was the number of VPNs. The size of the VPNs (number of VPN endpoints) is limited by the number of nodes in the network. Three categories were created again: small VPNs (number of endpoints ranges from 3 to 50% of the number of network nodes), large VPNs (number of endpoints is over 50% of the number of network nodes) and various size VPNs (from 3 to the number of network nodes). The VPN endpoints were selected according to uniform distribution from the network nodes.

Three different bandwidth selection methods were used. The first has assigned a constant value to all demands. The second has assigned uniformly distributed values between 50 and 250 bandwidth units. The third has assigned values according to normal distribution with mean = 100 and standard deviation = 25. These three methods created the initial bandwidth configurations.

The set of the tested configurations was the following:

- 3 networks
- number of VPNs (5, 10, 15)
- 3 VPN sizes
- generated bandwidth of demands (constant, 2 different uniform distribution, 2 different normal distribution)

The multiplication of these different types ($3 \cdot 3 \cdot 3 \cdot 5 = 135$) gives 135 test case, which includes 1,350 VPNs and 32,485 demands. So many test cases were evaluated in each design method with multiple, different settings.

To analyze the VPN design processes at the highest possible network load the bandwidth values were scaled in the following manner. In a given initial bandwidth configuration all bandwidth values were multiplied by a factor. The maximum value of the factor was searched, so that the problem still remained feasible.

The factor was determined with successive approximation with at least 8 steps of binary search by running and evaluating one of the design methods. The computationally least complex exact method is the *capacity minimization with splittable flows* for that the ILP contains only non-binary variables (see section 3.1.1). This design method was used to evaluate the feasibility, because its running time is under 1 second, and minimizes only the capacity reservation, but allows flow splitting. Taking the topology objective also into consideration or prohibiting flow splitting can only raise the capacity reservation, but not lessen. Therefore, by using this design method the maximal demand bandwidths can be determined for a given bandwidth distribution that is still feasible and the solution gives the minimal bandwidth reservation with splittable flows.

For all design methods the above described traffic files were used. To have more space for the optimization or to prohibit flow splitting or to include protection requires more capacity or smaller demands. From the two alternatives in the simulations not the bandwidth demands of the VPNs were decreased, but the network capacity was increased.

2.5 Metrics

To compare and evaluate the different design methods the following metrics were used. To evaluate the capacity objective the reserved capacity is determined relative to the available capacity in the empty network.

To evaluate the topology objective several metrics were defined. For unprotected VPN design the number of tree topology VPNs are determined in percentage to the number of all VPNs. For protected VPN design this is not an adequate metric because in protected VPNs each endpoint pair has two disjoint paths – a primary and a backup path – forming a cycle. Therefore, two new topology related metrics were constructed, the *VPN extension* and the *VPN node coverage*, that can be used for the unprotected VPN design as well.

Definition 2.5.1. VPN extension is the number of VPN edges divided by the number of VPN nodes-1.

In the unprotected VPN design, for a tree topology that contains only the VPN nodes the VPN extension is 1, because a tree contains nodes-1 edges. If the VPN extension is greater than 1 then either the VPN contains additional nodes because the VPN endpoints can not be connected directly or if the VPN contains only VPN endpoint nodes it is not a tree or both. Although for protected VPNs the VPN extension is always greater than 1, it is a good measure for their topological dispersion, too. The *VPN extension* metric is calculated for each VPN, the metric for the whole VPN configuration is the average of the individual VPN extension values.

Definition 2.5.2. VPN node coverage is the number of nodes that belong to a VPN (i.e. the VPN has a path that goes trough that node or it is the endpoint of the VPN) divided by the number of physical network nodes.

VPN node coverage specifies how many edges belong to a VPN in percentage to the total number of network nodes. The *VPN node coverage* metric is calculated for each VPN, the metric for the whole VPN configuration is the average of the individual VPN node coverage values.

If the node resources are partitioned among the different VPNs, it is an interesting issue, how many partitions have to be created, because fewer partitions means less overhead in the management system of the nodes. If the individual VPN node coverage values are summed it gives the number that specifies how many partitions must be created on a node on average.

Another metrics were also used that were not directly evaluating the objectives. The average path length for all VPN demands expresses whether the paths follow the shortest possible path or they were diverted to longer detours. The running time of the algorithms were also measured. Comparing the running times in case of exact methods (ILP) shows the complexity of the given problem, while in case of heuristics it shows the efficiency of the heuristic.

Some heuristic design methods can not always solve the problems with the same constraints entirely as the exact methods. In these cases the constraints can be relaxed to get a full solution. Or by keeping the original constraints the proportion of the partial solution can be measured. In the evaluation both ways were examined. In the latter case the granularity of the partial solution depends on the design method. With VPN based decomposition whole VPNs can fail, while with path based decomposition only paths can fail, which results in that some VPNs are not fully connected.

Chapter 3

Bandwidth Guaranteed VPN Design without Protection

3.1 Linear Programming Problem

The word "programming" is used in the sense of "planning". The importance of linear programming derives in part from its many applications and in part from the existence of good general-purpose techniques for finding optimal solutions. The related problem of integer programming (or integer linear programming, strictly speaking) requires some or all of the variables to take integer values. Integer linear programs (ILPs) often have the advantage of being more realistic than linear programs (LPs), but the disadvantage of being much harder to solve. Linear and integer linear programming have proved valuable for modeling many and diverse types of problems in planning, routing, scheduling, assignment, and design. Industries that make use of LP and its extensions include transportation, energy, telecommunications, and manufacturing of many kinds. [oNUL]

To formulate a linear programming problem the following items have to be defined:

- variables with their types and bounds
- objective function
- constraints

The variables are the solution variables, i.e. they will provide the solution. In the VPN design problem the route of the request and the associated network bandwidth has to be calculated in the network for each demand in each VPN. If for all demands on each edge the reserved bandwidth is known, then the route is also given, i.e. it can be determined from the bandwidth reservation. Therefore, a part of the variables will always represent the bandwidth reservation. These variables are assigned to each link–demand pair.

For solving the Integer Linear Programs the ILOG CPLEX software was used. This optimizer is one of the leading products among the commercial optimizers. The integer optimizer employs a branch-and-bound technique that takes advantage of innovative, cutting-edge strategies. The implementation includes among others the CPLEX presolve algorithm and sophisticated cutting-plane strategies such as Gomory, clique and cover, flow cover, GUB cover and implied bound. These features are applied automatically for the entered problem instances, therefore I have built upon these internal features of CPLEX.

3.1.1 Minimizing Capacity with Splittable Flows

Let the variables that represent the bandwidth reservation on a given link be non-binary variables and their values between zero and the bandwidth of the demand. The value of the variable is the amount of bandwidth transferred on the given link for a given demand. This way the value of the variable can be less than the bandwidth of the demand, i.e. *the flows can be split* between the source and the destination onto multiple paths. [**C4**]

With splittable flows the traffic between the source and destination can be transmitted on different paths, but the sum of path bandwidths must be equal to the bandwidth requirement of the demand. Because of the different paths the delay can be different on the different paths, therefore this method is not suitable for delay sensitive VPN applications, only for bandwidth guaranteed services.

The input data is given in section 2.3. These values are constants that will take part in the equations. The following variables are the solution variables associated with the problem: $X_e^{d^p_{(i,j)}}$ denotes the amount of bandwidth that edge e carries for demand $d^p_{(i,j)}$, i.e. in VPN p between endpoints i and j. The short notation is: X_e^d .

Y_e^p denotes the amount of bandwidth that edge e carries for VPN p, i.e. it represents the bandwidth reservation only regarding to which VPN it belongs to.

Both type of variables are integers because for the bandwidth reservation an integer multiplier of a fixed unit bandwidth is assumed.

X_e^d and Y_e^p variables cover the same, namely the used bandwidth in different approaches. While X_e^d shows the link usage according to the end to end demands, Y_e^p shows the whole amount on a link that a particular VPN uses.

The cost of the solution is proportional with the reserved total bandwidth. Therefore the sum of the solution variables gives the total cost of the network. However, it is possible to balance between the above described two points of view with a parameter ($\alpha \in \mathbb{R}; 0 \leq \alpha \leq 1$) in the objective function.

The ILP formulation is the following:
Variables:

$$X_e^d \in \mathbb{Z}, \quad 0 \leq X_e^d \leq b_e \tag{3.1.1}$$

$$Y_e^p \in \mathbb{Z}, \quad 0 \leq Y_e^p \leq b_e \tag{3.1.2}$$

Objective function:

$$\min \left\{ \alpha \sum_{\forall d \in D, \forall e \in E} X_e^d + (1-\alpha) \sum_{\forall p \in P, \forall e \in E} Y_e^p \right\} \tag{3.1.3}$$

subject to

$$\sum_{\forall u: e=(u,v)} X_e^d - \sum_{\forall w: e=(v,w)} X_e^d = \begin{cases} -b_d & \text{if } v \text{ is the source of } d, \\ b_d & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{3.1.4}$$

$$\sum_{\forall p \in P} Y_e^p \leq b_e \quad \forall e \in E \tag{3.1.5}$$

$$\sum_{\forall (i,j): d_{(i,j)}^p \in D} X_e^d \leq Y_e^p \quad \forall e \in E, \forall p \in P \tag{3.1.6}$$

In equation 3.1.3 if $\alpha = 0$, then only the viewpoint of the VPNs is considered in the objective, if $\alpha = 1$, then only individual flows are considered. Equation 3.1.4 is the flow conservation constraint, i.e. it ensures that traffic coming in and going out is the same at each node, when it is not the source or the destination of the flow. The first sum stands for the traffic coming in into node v and the second sum for the traffic going out. Equation 3.1.5 is a capacity constraint which expresses that the total capacity partitioned among the VPNs should not exceed the physical capacity bound. According to the definition of X_e^d and Y_e^p in equation 3.1.6 this constraint supposed to be equal. By using $\leq$ in the equation there is some space left for the optimization process. Because Y_e^p is bounded by eq. 3.1.5 therefore the sum on the left side cannot exceed the physical link capacity limit too.

Because both X_e^d and Y_e^p represent the amount of bandwidth *this optimization problem minimizes only the reserved capacity and not the topology*.

Complexity measured by ILP formulation:

Number of integer variables:	$\|E\|(\|D\| + \|P\|)$
Number of constraints:	$\|V\|\|D\| + \|E\|(2\|D\| + 3\|P\| + 1)$

Results

This design method aims only at the capacity objective, because all variables represent capacity. Thus, the number of tree VPN topologies remains under 6%. The value is above 0% in practice, because there are tree topologies even without the topology objective. For example they are several 3 node VPNs with directly connected nodes (traffic configurations with small size VPNs) that can form tree topology.

Because neither the X nor the Y variables are binary, the time consumption of the solver is very small, all calculations on the sample networks were solved under 1 sec on average.

This design method allows the splitting of the flows. The simulations show that this property is exploited in most cases, however, the number of split flows remains low, around 4% on average. The low time consumption and the small number of split flows suggests a practical way if unsplit flows are required: with the appropriate capacity extension only the

split flows need to be recalculated if unsplittable flows are required.

In this context the α balancing parameter does not influence the results to a great extent, because both X and Y variables represent the capacity reservation. Therefore the network utilization values are the same with different α settings.

However, there are differences regarding the other metrics.

If $0 \leq \alpha < 1$, i.e. Y is included in the objective function, the results are independent of α. In this case the average path length, the number of trees, and the number of split flows are slightly smaller, but not significantly (under 1%). If $\alpha = 1$, then Y variables are not included in the optimization and according to eq. 3.1.5 they can take values up to the capacity bound, therefore the above metrics have larger values.

Figure 3.1 and 3.2 shows these results grouped by the three different input networks (in order: NSFNet, COST-266 and GÉANT). Inside the groups in each pair the first line is for $\alpha = 1$ and the second is for $\alpha < 1$.

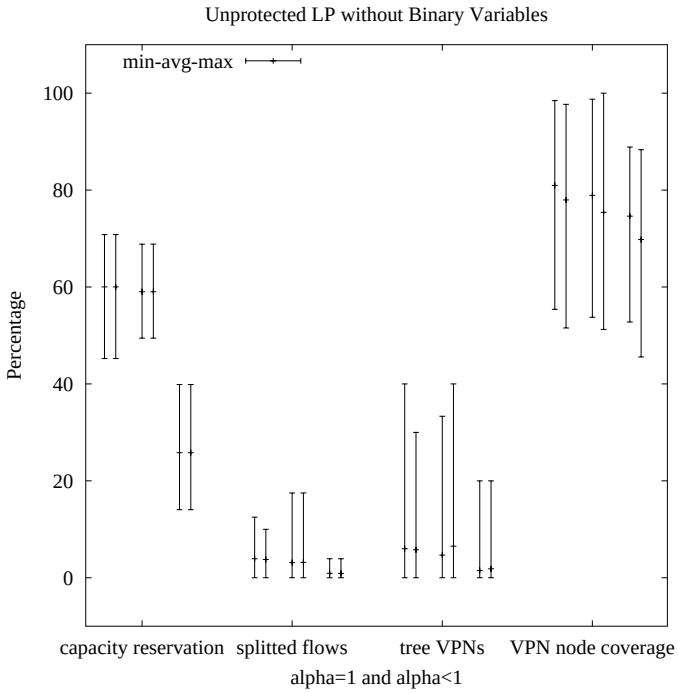


Figure 3.1: Minimizing Capacity with Splittable Flows – Metrics in Percentage

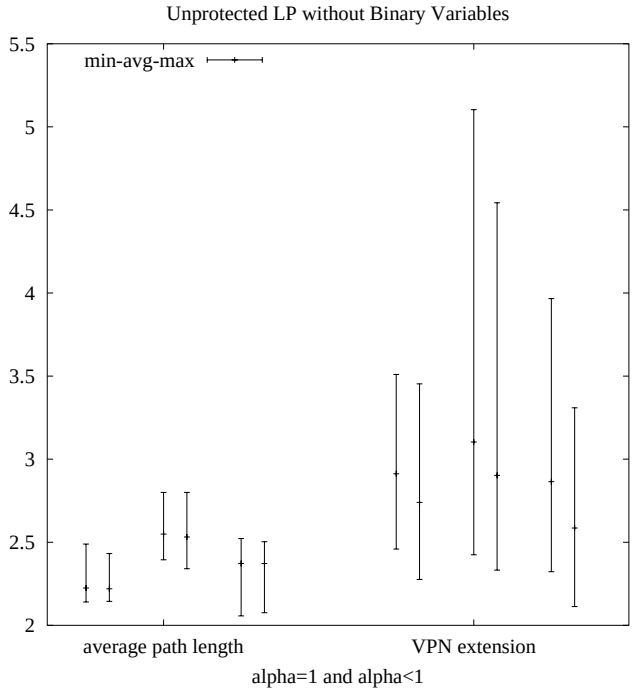


Figure 3.2: Minimizing Capacity with Splittable Flows – Metrics with Absolute Values

3.1.2 Minimizing Capacity and Topology with Splittable Flows

To minimize the topology, i.e. the number of virtual links, part of the variables must be binary to represent whether an edge is used by a VPN or not. Therefore the Y_e^p variables will be binary in the formulation. X_e^d remains integer variable and it represents the amount of bandwidth transferred on the given link for a given demand as in the previous section. Thus, *the flows can also be split* between the source and the destination onto multiple paths. [**C4**]

The following variables are the solution variables associated with the problem: $X_e^{d^p_{(i,j)}}$ denotes the amount of bandwidth that edge e carries for demand $d^p_{(i,j)}$, i.e. in VPN p between endpoints i and j. The short notation is: X_e^d.

Y_e^p denotes that VPN p has traffic on edge e, i.e. e is part of the virtual topology of VPN p.

X_e^d variables are integers because for the bandwidth reservation an integer multiplier of a fixed unit bandwidth is assumed. Y_e^p variables are binary, which is a special case of

integer variables (i.e. the possible values are 0 and 1).

The ILP formulation is the following:

Variables:

$$X_e^d \in \mathbb{Z}, \quad 0 \leq X_e^d \leq b_e \tag{3.1.7}$$

$$Y_e^p \in \{0,1\} \tag{3.1.8}$$

Objective function:

$$\min \left\{ \alpha \sum_{\forall d \in D, \forall e \in E} X_e^d + (1-\alpha) \sum_{\forall p \in P, \forall e \in E} Y_e^p \right\} \tag{3.1.9}$$

subject to

$$\sum_{\forall u: e=(u,v)} X_e^d - \sum_{\forall w: e=(v,w)} X_e^d = \begin{cases} -b_d & \text{if } v \text{ is the source of } d, \\ b_d & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{3.1.10}$$

$$X_e^d \leq Y_e^p b_d \quad \forall (i,j) : d_{(i,j)}^p \in D, \forall e \in E, \forall p \in P \tag{3.1.11}$$

$$\sum_{\forall d \in D} X_e^d \leq b_e \quad \forall e \in E \tag{3.1.12}$$

In equation 3.1.9 if $\alpha = 0$, then only the number of virtual edges are minimized, while if $\alpha \neq 0$ then the bandwidth reservation is also considered.

According to eq. 3.1.9 the cost of the solution is proportional to the reserved total bandwidth and the number of virtually used edges. The balancing parameter α is also very important because values for the reserved bandwidth and the number of virtually used edges differ significantly. The amount of reserved bandwidth is typically greater even by orders of magnitude than the number of virtual links. To emphasize the topology objective the value of α parameter must be less then $\frac{1}{\max(\text{bandwidth})}$, where the maximum is taken on all demands.

Equation 3.1.10 is the flow conservation constraints, i.e. it ensures that traffic coming in and going out is the same at each node, when it is not the source or the destination of the flow.

Equation 3.1.11 is a capacity constraint which expresses that when edge e is part of VPN p, i.e. $Y_e^p = 1$, then the value of X_e^d can be up to the bandwidth to the corresponding demand. However, when edge e is not part of VPN p, i.e. $Y_e^p = 0$ then it cannot carry traffic, thus $X_e^d = 0$. In the constraints for a given Y_e^p only those demands are selected that belong to VPN p, denoted as $\forall(i,j) : d^p_{(i,j)}$.

The constraint in eq. 3.1.12 ensures that the bandwidth utilization does not exceed the physical bounds.

Complexity measured by ILP formulation:

Number of integer variables:	$\|E\|\|D\|$
Number of binary variables:	$\|E\|\|P\|$
Number of constraints:	$\|V\|\|D\| + \|E\|(3\|D\| + 1)$

Results

This design method aims both at the capacity and the topology minimization, because X variables represent capacity and Y variables represent virtual links.

The balance between the capacity and topology cost components is determined by the α parameter. Using this α parameter in the objective function enables that arbitrary balance can be evaluated. α is an input parameter in the optimization process. The actual setting of α is determined by the VPN service provider, that designs the VPNs. The particular value of α depends on the VPN service provider's technology, hardware and software devices, etc., and on the associated cost model with them.

I analyzed which α settings are interesting for the optimization problem, therefore I examined the effects of the α parameter on the results. Table 3.1 shows the average values of the metrics calculated on the NSFNet traffic configurations. The result calculated in the other two networks showed the same tendencies, only with different particular values. If the cost of one unit of bandwidth is equal with the cost of one virtual link, then $\alpha \to 0$ provides the minimal overall cost (denoted as solution value in the table), because the amount of reserved bandwidth is always more than the number of virtual edges, and α cannot be zero, because then the capacity objective would have been omitted from the objective function.

α	0.9	0.5	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
Solution value	31608	17456	3616.46	454.05	112.36	72.56	68.08	67.52	*67.71*
Time (sec)	0.36	0.61	1.29	9.86	1808.00	2722.16	2943.72	3021.72	3377.33
Capacity reservation (%)	59.63	59.63	59.63	61.76	68.87	70.47	*70.33*	70.87	71.11
Path length	2.22	2.22	2.22	2.30	2.53	2.59	*2.58*	2.59	2.60
Tree VPNs (%)	9.88	9.88	9.88	28.15	66.91	69.14	69.75	*67.90*	*65.68*
VPN extension	2.44	2.42	2.39	1.92	1.54	1.53	1.53	1.53	*1.54*
VPN node coverage (%)	72.17	71.98	71.47	63.14	57.37	*57.61*	*57.56*	*57.51*	*57.55*
Time limit reached					+	+	+	++	+++
Minimization effect	cap.	cap.	cap. & top.	cap. & top.	cap. & top.	cap. & top.	top.	top.	top.

Table 3.1: Effects of the α Parameter

The table shows that the overall cost reduction in the solution value is marginal if $\alpha \leq 10^{-5}$. The time consumption shows that the one hour time limit in the solver is reached in increasing number of the test cases if $\alpha \leq 10^{-3}$. This causes that several values in the right side of the table, written in italics, are not exactly following the deceasing or increasing order of the metrics, because they are from a suboptimal solution.

The maximum bandwidth in the test traffic input files were under 1000 unit, therefore $\alpha \leq 10^{-3}$ places definitely more emphasis on the topology objective.

If $\alpha \geq 10^{-1}$, the reserved capacity, the path length, and the tree VPNs are minimal, and are equal for the different α values. Although the VPN extension and the VPN node coverage is decreasing, this is only marginal.

If $\alpha \leq 10^{-4}$, the capacity and topology metrics also reach their extreme values, and decreasing of the α parameter does not cause further significant change in the values.

Therefore, $\alpha \geq 10^{-1}$ realizes mainly the capacity minimization, $\alpha \leq 10^{-4}$ realizes mainly the topology minimization, and $10^{-4} \leq \alpha \leq 10^{-1}$ realizes jointly the capacity and

topology minimization. Thus, the design methods will be evaluated with $10^{-4} \leq \alpha \leq 10^{-1}$ parameter values.

Figure 3.3 and 3.4 show the results with the chosen α settings in groups for the three different input networks (in order: NSFNet, COST-266 and GÉANT).

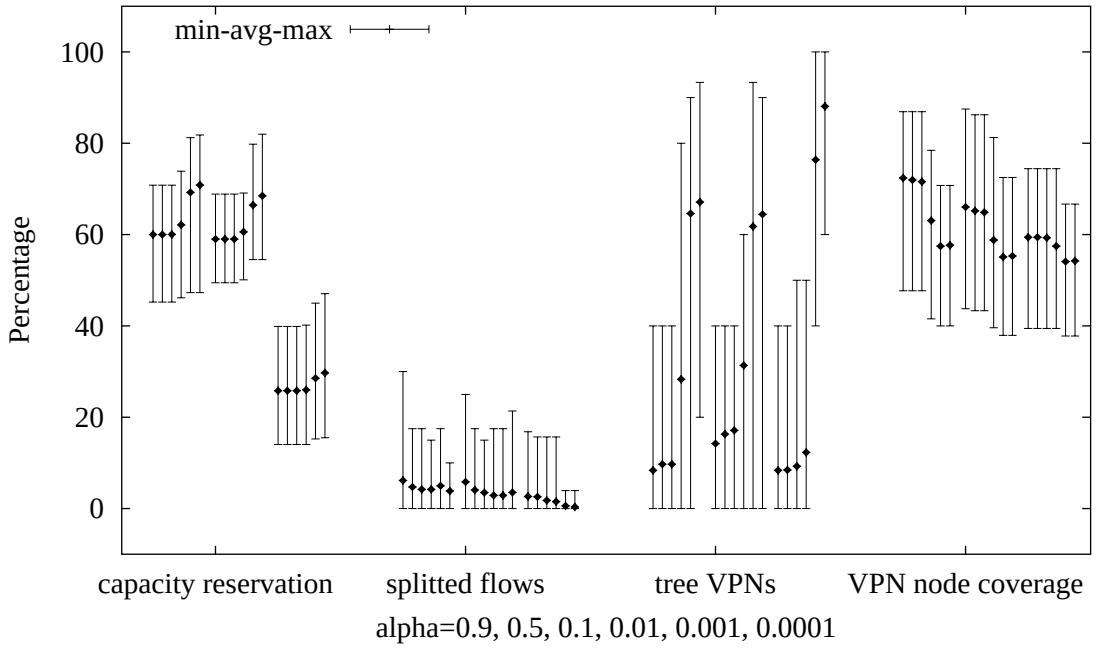


Figure 3.3: Minimizing Capacity and Topology with Splittable Flows – Metrics in Percentage

With $\alpha = 1$ only the capacity objective would be minimized. However in the ILP formulation Y variables represent the topology minimization, therefore to be able to evaluate the topology related metrics Y variables must be present in the objective function. Therefore I also included $\alpha = 0.9$ as the largest α value in the investigations.

If $0.01 \geq \alpha \geq 0.0001$, the capacity reservation is increasing to fulfill the topology objective. The difference is significant if $\alpha \leq 0.001$ up to 10%. The average number of tree topology VPNs is definitely higher, 60–90% instead of 10–20%. The VPN node coverage is also decreasing, but not very significantly, from 60–70% to 55–60%. The number of split flows tend to be decreasing with decreasing α, except for very low α values, where the topology minimization can cause slightly more split flows. The average path length is

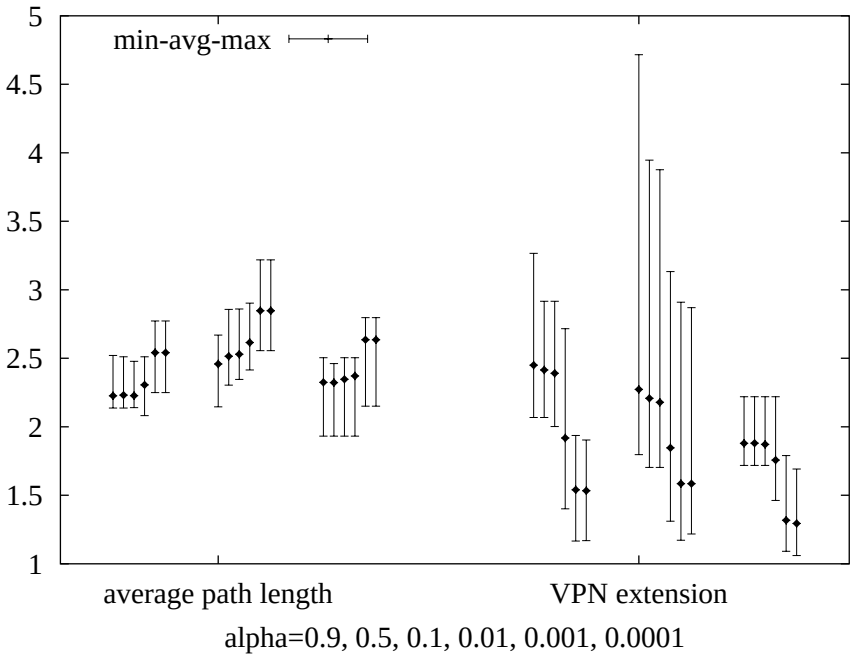


Figure 3.4: Minimizing Capacity and Topology with Splittable Flows – Metrics with Absolute Values

increasing and the VPN extension is decreasing, both have a jump at $\alpha = 0.001$, where the topology objective gets more emphasis.

From the figure it can be also seen that putting even more emphasis on the topology objective with $\alpha \ll 0.001$ has improved the percentage of tree topology VPNs and the VPN node coverage less than from 0.01 to 0.001. Moreover, the VPN node coverage is slightly higher. The reason for this latter is most probably comes from the time limitation of the computation process. Because Y variables are binary, the time consumption of the solver is low, only if the binary Y variables have smaller weight in the objective function. That is, if $\alpha > 0.001$ all calculations on the sample networks were solved under 1 min on average. However, if $\alpha \leq 0.001$ the time consumption was extremely high. To be able to evaluate the 135 test cases with multiple different α parameter values in reasonable time, 1 hour time limit was set in the solver. If the time limit is reached, a suboptimal (so far the best integer) result is returned. This time limit was reached for about 40% of the tests (traffic files with more VPNs) for $\alpha = 0.001$ and about 70% for $\alpha = 0.0001$.

Figure 3.3 shows that the number of tree topologies remains under 70–88%. This is due to the capacity constraints. To evaluate how much more capacity is required to realize tree topologies the capacity bounds were relaxed. Figure 3.5 compares the original (100%) and the relaxed (110% and 150% of the original link capacities) capacity bounds. Unfortunately, relaxing the capacity bounds do not shorten the running time of the solver. Therefore as a compromise $\alpha = 0.001$ was used for the test runs. The results show that for enforcing tree VPN topologies requires considerable amount of additional capacity. Adding 50% free capacity can achieve only 85% on average of the topologies to form a tree, while the VPN node coverage is reduced. The other reason that not all VPN topologies are trees is that flow splitting is not forbidden, and although the number of split flows is very small, it is not 0, thus the topology of these VPNs cannot be a tree.

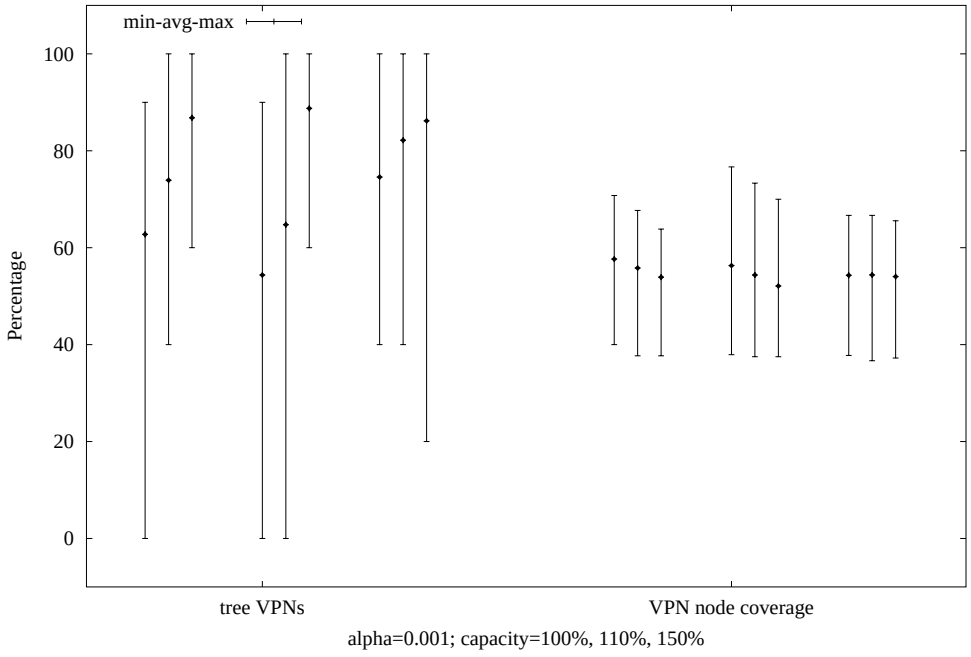


Figure 3.5: Minimizing Capacity and Topology with Splittable Flows – Capacity Bound Relaxation

3.1.3 Minimizing Capacity and Topology with Unsplittable Flows

In typical VPN applications the flow splitting is not allowed, because the different paths between the VPN endpoints in the same VPN can have different Quality of Service parameters apart from the bandwidth, e.g. the delay. Therefore it is not recommended to use multiple paths if constant delay bounds are required.

To achieve this, not only the variables that represent the VPN relationship (Y_e^p) but the variables associated with the bandwidth reservation are also binary (X_e^d). This way the solution of the problem gives *unsplittable flows* between the sources and the destinations and allows to minimize the topology as well. [**C5**]

The following variables are the solution variables associated with the problem: $X_e^{d^p_{(i,j)}}$ indicates that edge e is in the path that carries demand $d^p_{(i,j)}$, i.e. the traffic in VPN p between endpoints i and j. The short notation is: X_e^d .

Y_e^p indicates whether VPN p has traffic on edge e, i.e. e is part of the virtual topology of VPN p.

Both X_e^d and Y_e^p variables are binary, which is a special case of integer variables (i.e. the possible values are 0 and 1).

The ILP formulation is the following:
Variables:

$$X_e^d \in \{0,1\} \tag{3.1.13}$$

$$Y_e^p \in \{0,1\} \tag{3.1.14}$$

Objective function:

$$\min \left\{ \alpha \sum_{\forall d \in D, \forall e \in E} X_e^d b_d + (1-\alpha) \sum_{\forall p \in P, \forall e \in E} Y_e^p \right\} \tag{3.1.15}$$

subject to

$$\sum_{\forall u:e=(u,v)} X_e^d - \sum_{\forall w:e=(v,w)} X_e^d = \begin{cases} -1 & \text{if } v \text{ is the source of } d, \\ 1 & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{3.1.16}$$

$$X_e^d \leq Y_e^p \quad \forall (i,j) : d_{(i,j)}^p \in D, \forall e \in E, \forall p \in P \tag{3.1.17}$$

$$\sum_{\forall d \in D} X_e^d b_d \leq b_e \quad \forall e \in E \tag{3.1.18}$$

In equation 3.1.15, because X_e^d is binary, it is multiplied by the bandwidth of the demand b_d. According to eq. 3.1.15 the cost of the solution is proportional with the reserved total bandwidth and the number of virtually used edges. The setting of parameter α is important because values for the reserved bandwidth and the number of virtually used edges differ significantly.

Equation 3.1.16 is the flow conservation constraint, i.e. it ensures that traffic coming in and going out is the same at each node, when it is not the source or the destination of the flow. Because the variables are binary the right side of the equation is $-1, 0, 1$ instead of the amount of bandwidth.

Equation 3.1.17 is a capacity constraint which expresses that when edge e is part of VPN p, i.e. $Y_e^p = 1$, then it is able to carry traffic, i.e. the value of X_e^d can be 1. However, when edge e is not part of VPN p, i.e. $Y_e^p = 0$, then it cannot carry traffic for demand d, thus $X_e^d = 0$. In the constraints for a given Y_e^p only those demands are selected that belong to VPN p, denoted as $\forall (i,j) : d_{(i,j)}^p$.

The constraint in eq. 3.1.18 ensures that the bandwidth utilization does not exceed the physical bounds.

Complexity measured by ILP formulation:

Number of binary variables: $|E|(|D| + |P|)$

Number of constraints: $|V||D| + |E|(|D| + 1)$

Results

This design method aims both at the capacity and the topology objectives, where the X variables are binary, and therefore the flows do not split.

Because in the previous design methods the splitting of flows were permitted, the same traffic configurations with unsplittable flows cannot be solved in each test case.

The tests has been shown that on average around 10% of the traffic configurations cannot be solved with unsplittable flows, that where solved with splittable flows. However, whether the VPN traffic configuration could be solved with unsplittable flows or not, does not depend individually on the number of split flows in the splittable solution. Except that, if there is a solution with unsplit flows in the splittable case, then this is also a solution in the unsplittable case.

Adding 10% extra capacity resolves this, more than 99% of the test cases could be solved with unsplittable flows if the capacity bounds were relaxed.

The results with different α settings for the three different input networks (in order: NSFNet, COST-266 and GÉANT) with the original capacity bounds in figure 3.6 and 3.7 are very similar to the results for splittable flows.

However, the time consumption of calculating unsplittable flows is definitely larger. The difference is decreasing if the capacity objective are getting more emphasis with decreasing α. Table 3.2 shows this differences where the tests were run on a 2 GHz PC. The unsplittable flow calculation with $\alpha = 0.01$ reached the 1 hour time limit in few cases (around 3%), with lower α values the 1 hour time limit was reached in significant number of the test cases.

α	Average time consumption with splittable flows	Average time consumption with unsplittable flows
0.9	1.29 sec	105.10 sec
0.1	5.75 sec	80.15 sec
0.01	75.96 sec	281.06 sec

Table 3.2: Average Time Consumption of Splittable and Unsplittable Flow Problems

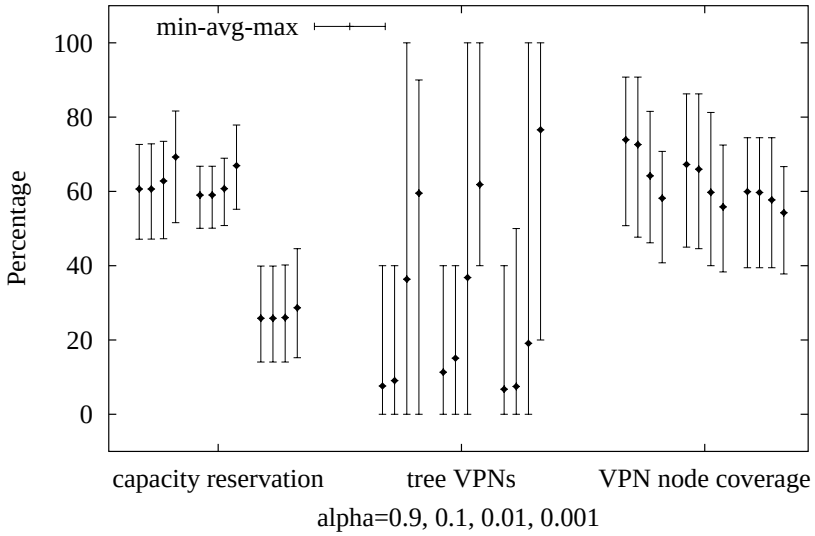


Figure 3.6: Minimizing Capacity and Topology with Unsplittable Flows – Metrics in Percentage

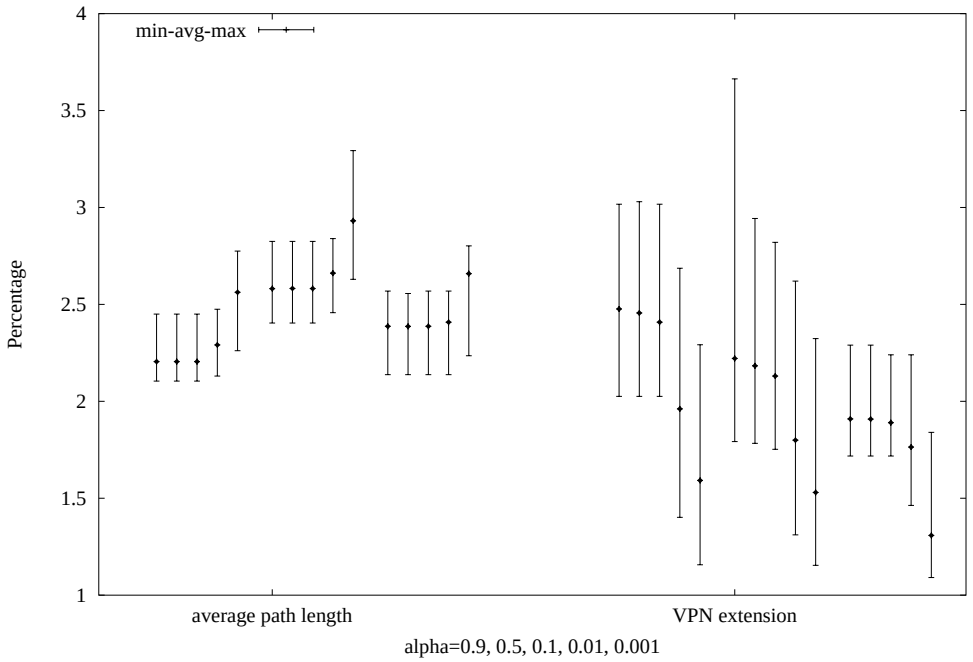


Figure 3.7: Minimizing Capacity and Topology with Unsplittable Flows – Metrics with Absolute Values

3.2 Heuristics

A heuristic is a method that might not always find the best solution, but is aimed to find a good solution in reasonable time. The search space is the set of possible solutions. A heuristic is a way to navigate in the search space. Most heuristics work by reducing the size of the search space, or giving you a way to move from one place to the next within a search space. Heuristics are essentially methods for finding your way around a search space without searching every single point in the space, i.e. avoiding the exhaustive (complete) search (brute-force search).

3.2.1 VPN Based Heuristics with Score Ordering

For the problems in the previous section the global optimization technique is applied, where all VPNs with all their traffic demands are optimized at once, and the global optimum is guaranteed whenever it exists and the solver can provide it in acceptable time. The global optimum means, that all demands are calculated simultaneously and get routed in the network, while the objective function is minimized. But this problem can be quite complex and time consuming (higher in orders of magnitude compared to decomposition and heuristics techniques).

One way to reduce the computational complexity is to decompose the global problem into smaller subproblems that could be solved separately and then build a global solution from the results of the solutions of the subproblems. This kind of solution does not guarantee that the global optimum will be found, but they can give good suboptimal results that are close to the global optimum. The important gain is the reduced running time in return for the sub-optimality.

A straightforward decomposition can be done according to the VPNs, i.e. the VPNs are designed one-by-one [**C7**]. In a real network environment the operator is requested time to time to set up a new VPN for a new group of customers, thus one step of the decomposed design can be used in such situations as well.

The decomposition of the global integer linear programming problem gives the smaller

subproblems that are going to be solved one-by-one. The number of subproblems is the number of VPNs ($|P|$). The complexity measured by problem formulation is reduced because instead of the whole demand set only the appropriate VPN's demand set is included in the subproblems, which yields significantly less variables and constraints, thus reducing the search space.

Integer linear programming problems that are formulated separately for each VPN are solved one after another. After a solution for a VPN is ready the available bandwidth is reduced on the links according to the solution, so the available free capacity will be lessen for the next optimization. Therefore, the order in which the VPNs are calculated plays a crucial role in determining the actual configuration. VPNs that are calculated early in the process have more resources available than those calculated later, because previously calculated VPNs or paths can consume some critical network resources and can hinder new demands – and therefore VPNs – getting accommodated. Therefore the ordering of VPNs influences the results: if the order in which the VPNs or paths are calculated is changed, the resulting configuration will also change.

To goal with the ordering is to route as many VPNs as possible. This is similar to packing problems (knapsack, bin packing) [GJ79]. In the packing problems the volume and the cost of the objects are known. However, when routing a demand the reserved capacity in the network depends on the selected route. Therefore I proposed the following ordering heuristics: each demand gets a score according to the bandwidth requirement of the demand and the distance between the source and destination nodes. This score estimates the "volume" of the path. These demand scores are summed and this will be the score of the VPN. This gives the order in which the VPNs are optimized. [**C7**, **C9**, **J1**]

This kind of VPN-by-VPN optimization method can be built into an off-line planning tool. Such a VPN planning tool should be implemented in the network management center of the operator. After the off-line calculation is completed, the paths can be established in any order because each is installed following the rules of the optimized solution.

The *minimizing capacity with splittable flows* problem can be solved quickly, therefore decomposition is not required for this design method. The other two ILP formulations that

contain binary variables require significantly more time to solve, mainly, if the topology objective is emphasized. Thus, the heuristic was used to compute the *minimizing capacity and topology* problems for both splittable and unsplittable flows.

Results for Minimizing Capacity and Topology with Splittable Flows

The VPN based heuristic does not provide a full solution in all cases. This is due to two reasons:

- The superposition of local optimum cannot always result in a full solution, because each previous solution decreases the available capacity in the network.
- The bandwidth requirements in the traffic files are set to reach the highest possible network load while the capacity minimization problem still remains feasible. Therefore some links become highly loaded.

Thus, in part of the test cases some of the subproblems become unfeasible, because the previous suboptimal solutions of the subproblems have decreased the available capacity in the network in such a way that the bandwidth requirement of the demands of the next VPN(s) cannot be satisfied.

The scoring system – that determines the order – is based on the bandwidth requirement of the demand and the distance between the source and destination nodes. The question is whether to prefer high or low bandwidth demands, near or far endpoints? The basic concept was to prefer high bandwidth demands with near endpoints. However, the test runs have shown, that for splittable flows this is not the adequate preference mode (see Table 3.3). Preferring demands with low bandwidth and with far endpoints results in more full solutions. This is due to the ability to split flows, because the demands that going to be routed later in the process can split to several small bandwidth flows, and therefore the capacity bounds are not so restrictive as by the unsplittable flows.

The balancing parameter between the capacity and topology objectives (α) greatly influences the ratio of the partial solutions, because reducing the topology locally in each VPN concentrates the traffic on fewer number of links, that will be highly loaded. Thus, this can hinder the next VPN(s) to connect their endpoints with guaranteed bandwidth. Figure 3.8 shows the ratio of partial solutions for the test cases and the average number of not

α	Prefer high bandwidth with near endpoints	Prefer low bandwidth with far endpoints
0.9	34.1%	28.9%
0.1	34.1%	29.7%
0.01	47.4%	43.7%
0.001	80.0%	79.3%

Table 3.3: Ratio of Partial Results Depending on Preference Modes

routed VPNs depending on α. The ratio of the partial solutions is increasing from 25% up to 80% and the number of unsuccessful VPN designs from 3% to 16%. This indicates that by emphasizing the topology objective the increase in the unsuccessful VPN designs gets more distributed among the test cases. With $\alpha = 0.001$ the number of unsuccessful VPN designs still remains at 16%, however, these VPNs affect 80% of the test cases.

However, relaxing the capacity bounds decreases the ratio of partial solutions. Figure 3.9 shows that adding only 10% capacity significantly lowers the ratio of partial solutions from 80% to 30%. Adding 50% extra capacity results in achieving full solution for all test cases.

For fair comparison with the results of the Integer Linear Programming, I have chosen such α parameter and capacity extension combinations, that produces enough complete solution with the heuristics. To evaluate the effectiveness of the decomposed design method two instances were chosen, $\alpha = 0.1$ with original capacity bounds, where the unsuccessful ratio is 30% and $\alpha = 0.001$ with the 50% capacity extension, where all test case is successfully solved with the decomposed design. As Figures 3.10 and 3.11 show there is not much difference between the results, if the α parameter and the capacity extension is chosen such a way, that most of the test cases can be completely solved with the heuristics. The capacity reservation and the number of split flows is slightly higher, while the VPN node coverage is slightly lower. The difference in the number of tree VPNs depends on how many free capacity is available: it is lower if the free capacity is limited, and it is higher if there is more free capacity. All differences are below 5%. This means, if the decomposed VPN heuristics is able to give a full solution, then the results are so close to the optimal solution.

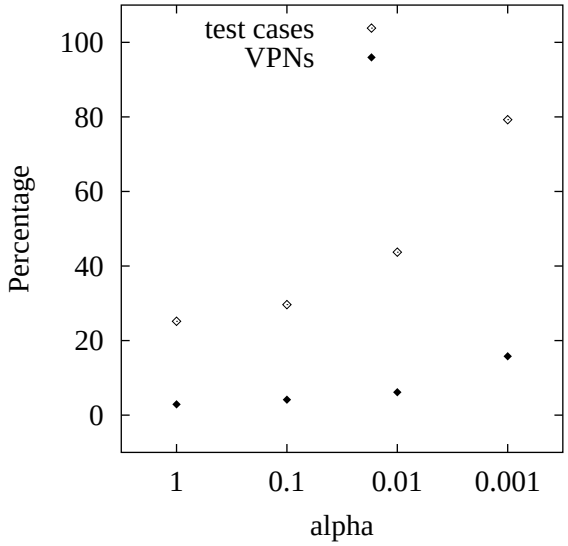


Figure 3.8: VPN Based Heuristic with Splittable Flows – Percentage of Partial Solutions

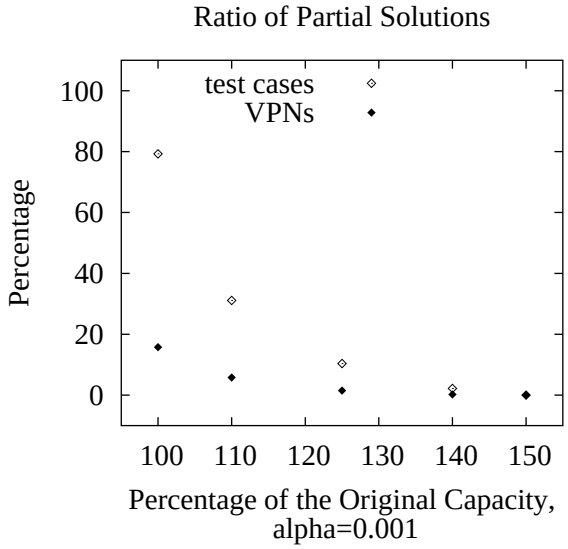


Figure 3.9: VPN Based Heuristic with Splittable Flows – Relaxing the Capacity Bounds

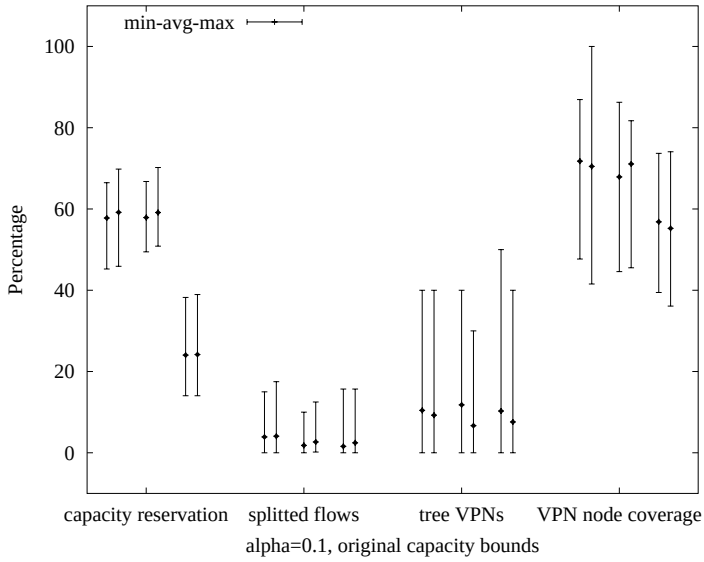


Figure 3.10: VPN Based Heuristic with Splittable Flows – Comparison of Global and Decomposed ILP Solution with Initial Capacity Bounds

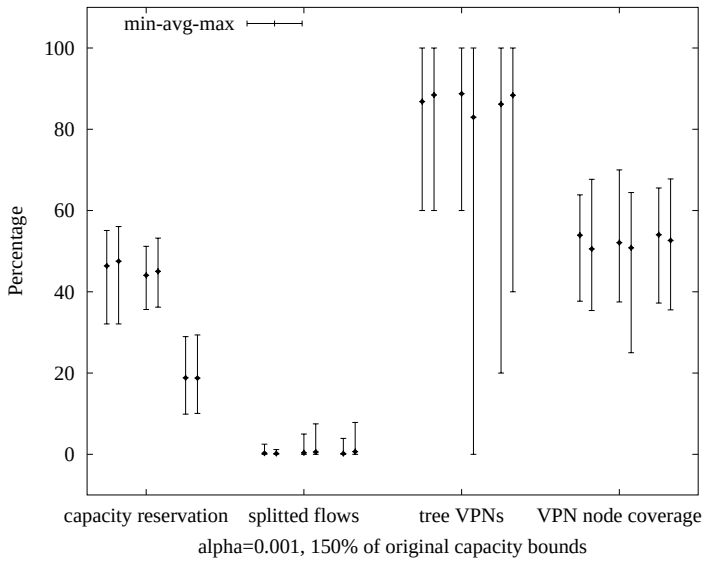


Figure 3.11: VPN Based Heuristic with Splittable Flows – Comparison of Global and Decomposed ILP Solution with Relaxed Capacity Bounds

Results for Minimizing Capacity and Topology with Unsplittable Flows

Because with unsplittable flows 10% of the test configurations could not be solved, only the remaining test configurations were evaluated with the original capacity bounds.

The preference function of the scoring system was analyzed again. Table 3.4 shows that for unsplittable flows the preference does not clearly influence the ratio of partial results depending on different α settings. This is due to that flows cannot be split, therefore the options for the possible paths are fewer than with splittable flows.

α	Prefer high bandwidth with near endpoints	Prefer low bandwidth with far endpoints
0.9	55%	50%
0.1	52.5%	50%
0.01	68.3%	66.7%
0.001	88.3%	91.7%

Table 3.4: Ratio of Partial Results Depending on Preference Modes with Unsplittable Flows

In the following results the demands with high bandwidth and near endpoints were preferred, because with emphasized topology minimization ($\alpha = 0.001$) it gives slightly less partial results. Figure 3.12 shows the ratio of partial solutions and also the ratio of not planned VPNs depending on α parameter. Comparing it with the splittable case (Figure 3.8) the ratio of partial results is higher (over 50% instead of over 30%). This indicates again that solving the problem with unsplittable flows is harder. The table also shows the dependence on α parameter. If the topology objective is emphasized ($\alpha \leq 0.001$), the superposition of locally optimized VPN design hinders the accommodation of lower ranked VPN(s). However, the ratio of unsuccessfully routed VPNs still remains under 20%.

To reduce the number of partial solutions the capacity bounds need to be relaxed again. Figure 3.13 shows that the result are very similar to the splittable design mode (Figure 3.9), i.e. a small amount of capacity extension significantly reduces the number of partial solutions and 50% capacity extension provides complete solution for all test cases.

Comparing Tables 3.5 and 3.2 it can be seen that the decomposition reduces the running time for splittable flows if $\alpha \leq 0.01$. For larger α values the global ILP problem can be

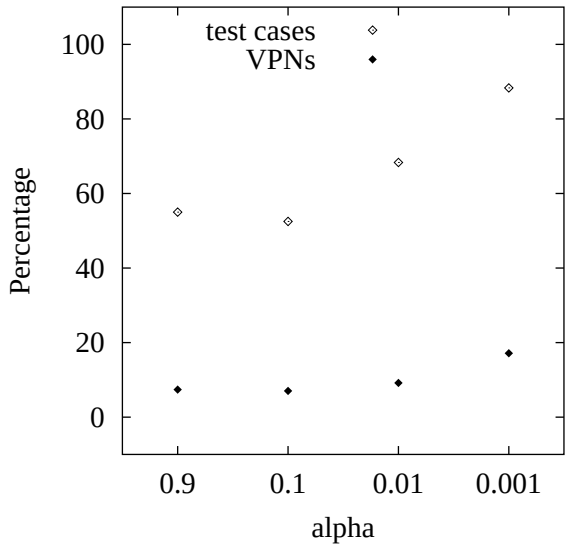


Figure 3.12: VPN Based Heuristic with Unsplittable Flows – Percentage of Partial Solutions

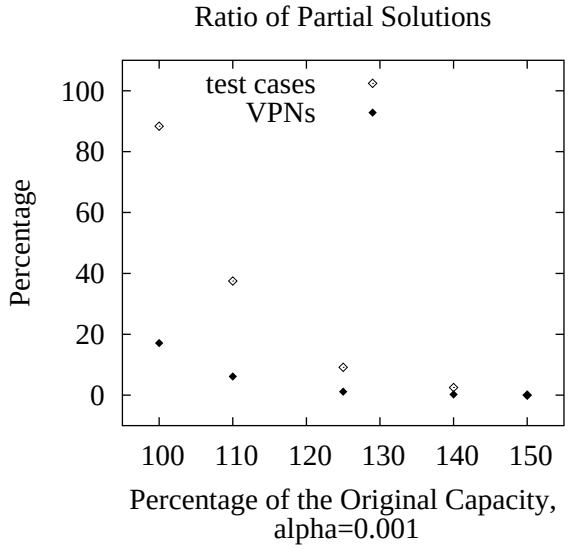


Figure 3.13: VPN Based Heuristic with Unsplittable Flows – Relaxing the Capacity Bounds

solved in shorter time, because the decomposition involves some overhead. In the case of unsplittable flows, for all α values the decomposition reduces the time consumption significantly. It must be noted, that for $\alpha = 0.001$, where the solving of global ILP problem has reached the 1 hour time limitation in significant number of the test cases, the decomposed ILP problem remains under 1 minute both for splittable and unsplittable flows.

α	Average time consumption with splittable flows	Average time consumption with unsplittable flows
0.9	13.1 sec	11.5 sec
0.1	13.0 sec	11.6 sec
0.01	13.4 sec	11.1 sec
0.001	45.6 sec	55.0 sec

Table 3.5: VPN based Heuristics – Average Time Consumption of Splittable and Unsplittable Flow Problems

3.2.2 Path Based Heuristics

A deeper level of decomposition is made when one path is calculated at a time. It is fast and simple enough to be used especially for huge networks. In this case the search space is the possible paths of the demands.

To reduce the search space for a particular demand not all possible paths are examined. Only one appropriate path is searched during the process, which is the shortest path in the weighted graph. To find a path between a source and a destination Dijkstra's shortest path algorithm is used [Dij59]. Dijkstra's algorithm finds the shortest path between two vertices in a directed or undirected non-negative weighted graph in polynomial time. The shortest path algorithm finds one unsplit path, therefore it corresponds to the *binary variables only* ILP formulation.

To solve the splittable flow problem with heuristic, flow algorithms can be applied instead of shortest path algorithms, but this was not further investigated, because telecommunication applications require mostly unsplittable flows.

Weight Assignment

Setting the weights according to the appropriate objectives determines which path will be found by Dijkstra's algorithm [**C7**]. If the goal is the capacity objective, i.e. to minimize the bandwidth reservation, the weights of the edges are set to the bandwidth to be reserved on that edge if the selected demand gets accommodated on that edge.

If the main goal is the topology objective, i.e. to minimize the number of virtual links, then the already used edges get lower weight than the unused ones independently of the bandwidth reservation. This way the new paths prefer to follow the already established ones, not allowing the dispersion of the paths. The new established paths alter the set of already used edges, therefore the weighting of links is modified in each step. This simple weight assignment enforces the topology objective, e.g. a 10:1 weight ratio (10 for not used and 1 for already used edges). In this case the capacity objective is fulfilled only partially, because Dijkstra's algorithm finds the shortest path according to the weights. If there is a possible path containing an edge with 10 unit weight, it will not be selected even if it is shorter in distance than an another possible path containing only 1 unit weight edges, causing higher bandwidth reservation.

The combination of the capacity and topology objectives are expressed in a more sophisticated weighting scheme as follows. The demands are processed one by one. The weight of an edge is the currently processed demand's bandwidth if a path of a demand already uses that edge from the same VPN. This base weight is responsible for the capacity objective. Otherwise the weight is the bandwidth of the currently processed demand plus an offset, which is responsible for the topology objective. This additional weight tries to keep the paths along the already established ones. This offset was chosen to be the average demand bandwidth multiplied by a parameter. Of course the weight of the edges that are not capable to accommodate the demand – because of the capacity constraint – is infinite.

The meaning of the offset is the following. If the multiplier parameter is e.g. 5 and we consider a demand with average bandwidth, the cost of a 6 hops long path consisting of already used edges is equal to a one hop long path (i.e. a direct edge) which is not yet used.

Therefore, for an average size demand if there is an available path containing already

used edges and it is not too long, it will be chosen. If it is too long, then a new not yet used shortcut path will be chosen. Whether a path is regarded as too long or not depends on the multiplier parameter.

A demand with over average bandwidth is less likely to follow the already used edges, while a demand with under average bandwidth is more likely to do so. Thus, the already used edges are preferred, but demands with over average bandwidth are avoided to route on too long paths.

Defining the offset this way includes a multiplier parameter which enables to influence the path length. The another component of the offset is based on the average demand bandwidth therefore the weight differentiates between small and large demands.

Demand Ordering

Score Based Ordering This is a simple heuristic in which the paths are calculated one by one in one run. An ideal order of the demands is determined in which they can get accommodated by the shortest path algorithm. The goal of the ordering is to be able to route as many demands as possible. [**C7**, **C9**, **J1**]

The scoring system is similar to the one used at the VPN based heuristics. Each demand gets a score according to the bandwidth requirement of the demand and the distance between the source and destination nodes. The distance of source and destination is calculated as a hop distance between the two vertices without bandwidth restrictions. Because in this method the demands are individually scored and routed the scoring has greater influence to

the ratio of partial solutions. Therefore I investigated the following scoring variants:

$$\text{Score} = \beta \text{Score}_{\text{distance}} + (1-\beta)\text{Score}_{\text{bandwidth}} \tag{3.2.1}$$

$$\text{Score}^1_{\text{distance}} = \frac{d}{d_{\text{max}}} \tag{3.2.2}$$

$$\text{Score}^2_{\text{distance}} = 1 - \frac{d}{d_{\text{max}}} \tag{3.2.3}$$

$$\text{Score}^1_{\text{bandwidth}} = \frac{b}{b_{\text{max}}} \tag{3.2.4}$$

$$\text{Score}^2_{\text{bandwidth}} = 1 - \frac{b}{b_{\text{max}}} \tag{3.2.5}$$

where b is the bandwidth of the demand, and d is the distance between the endpoints, and the maximum is taken on all demands. β balances between the weighting of the distance and bandwidth based score ($\beta \in \mathbb{R}; 0 \leq \beta \leq 1$). $\text{Score}^1_{\text{distance}}$ prefers demands with far endpoints, $\text{Score}^2_{\text{distance}}$ prefers demands with near endpoints. $\text{Score}^1_{\text{bandwidth}}$ prefers high bandwidth demands, $\text{Score}^2_{\text{bandwidth}}$ prefers low bandwidth demands. Four combinations from these preference functions (high bandwidth – near endpoints, high bandwidth – far endpoints, low bandwidth – near endpoints, low bandwidth – far endpoints) and five different values of distance / bandwidth ratio ($\beta = 0, 0.25, 0.5, 0.75, 1$) were evaluated regarding the ratio of partial results.

The following results are with the 10:1 weight assignment. Figure 3.14 shows the number of demands that cannot be routed and figure 3.15 shows how many full solution is provided by the score ordering variant. From Figure 3.14 it can be seen that preferring demands with near endpoints (with high or low bandwidth) results in less not routed demands on average with $\beta = 0.75$. However, in Figure 3.15 it is clear, that the high bandwidth – near endpoints preference provides the most full solutions, again with $\beta = 0.75$. Therefore the goal to route as many as possible demands is satisfied with the high bandwidth – near endpoints preference with 0.75 distance to bandwidth weighting ratio. It must be noted that even if the ratio of full solutions is under 17.5% the ratio of not routed demands is under 6%, but this 6% of demands are distributed among the VPNs such a way, that only less than 17.5% is not affected.

To compare the effectiveness of the heuristic the results were compared to the results of

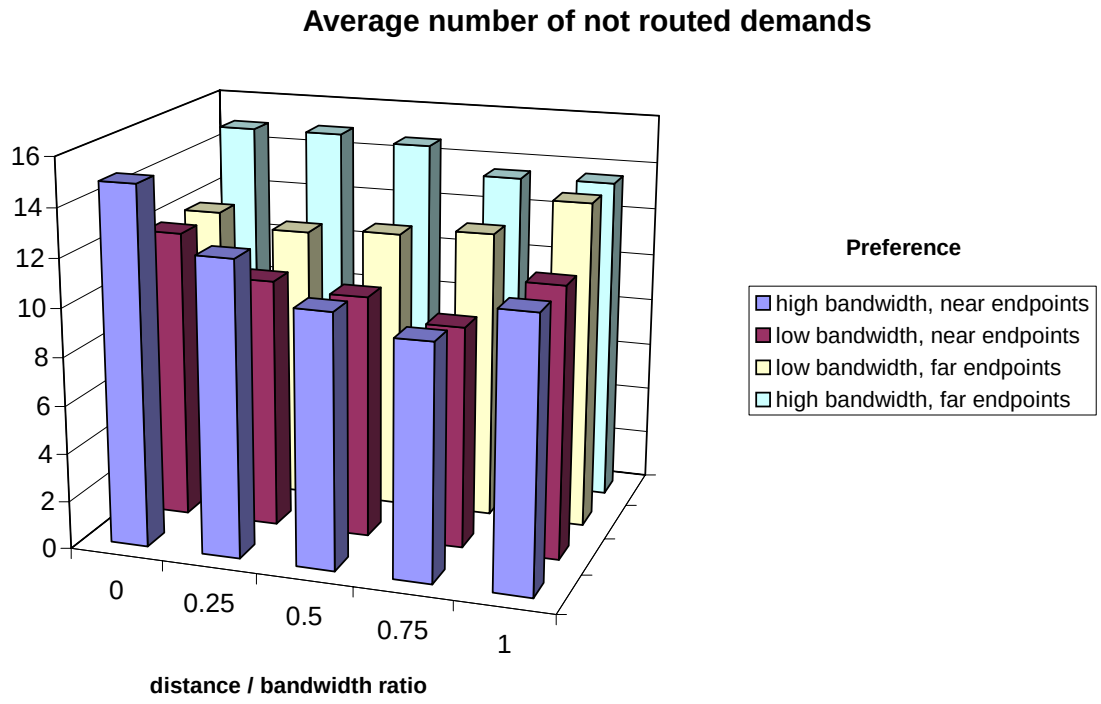


Figure 3.14: Scoring Variants – Average Number of Not Routed Demands

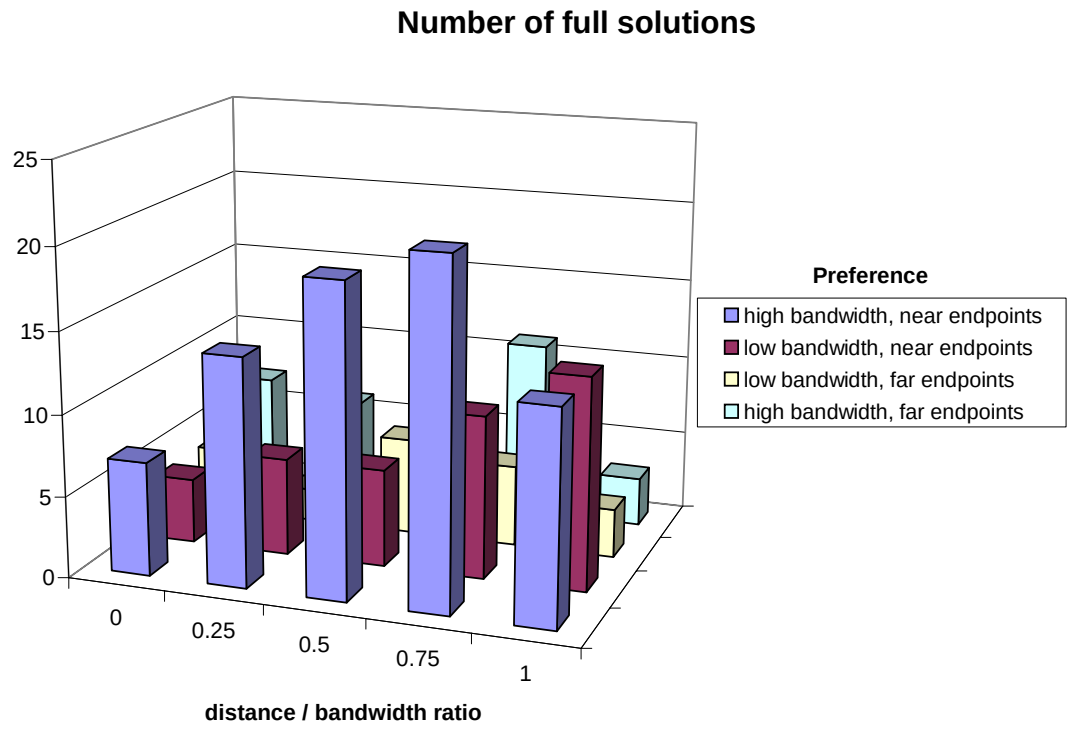


Figure 3.15: Scoring Variants – Number of Full Solutions

the ILP solution. Table 3.6 shows in which region the results of the heuristic are according to the capacity / topology balancing parameter α in ILP formulation. The results of the heuristic are in bold frames. According to the topology related metrics (tree VPNs, VPN extension, VPN node coverage) the results correspond to $\alpha \approx 0.01$, that is, the topology objective is only emphasized partially. However, the average time consumption is very low. The capacity reservation and the average path length is higher than in the ILP solution with emphasized topology objective ($\alpha = 0.001$).

α		0.9		0.1		0.01		0.001	
tree VPNs (%)		8.56		10.53	**24.97**	30.77		66.35	
VPN extension		2.20		2.14	**1.89**	1.84		1.47	
VPN node coverage (%)		66.92		66.00	**64.24**	60.50		55.98	
Time consumption (sec)	**0.10**	32.11		33.76		156.56		1993.79	
Capacity reservation (%)		48.21		48.21		49.56		54.11	**55.63**
Average path length		2.39		2.39		2.45		2.72	**2.92**

Table 3.6: Path Based Heuristic Compared to ILP

Relaxing the capacity bounds can improve again the results of the heuristic. Figure 3.16 illustrates that adding a small amount of extra capacity drastically decreases the ratio of partial solutions.

Regarding the performance of the heuristics: adding 50% extra capacity can achieve the emphasis on the topology objective. The result are nearly as good as the ILP solution with $\alpha = 0.001$ as Table 3.7 shows, and the time consumption is still around 0.1 sec.

Metrics	ILP solution	Heuristic solution
	Initial capacity $\alpha = 0.001$	150% of initial capacity
Tree VPNs (%)	66.35	59.67
VPN extension	1.47	1.51
VPN node coverage (%)	55.98	56.61
Capacity reservation (%)	54.11	59.32*
Average path length	2.72	2.97

*the actually reserved capacity is multiplied by 1.5 for comparison

Table 3.7: Achieving Topology Minimization with Path Based Heuristic

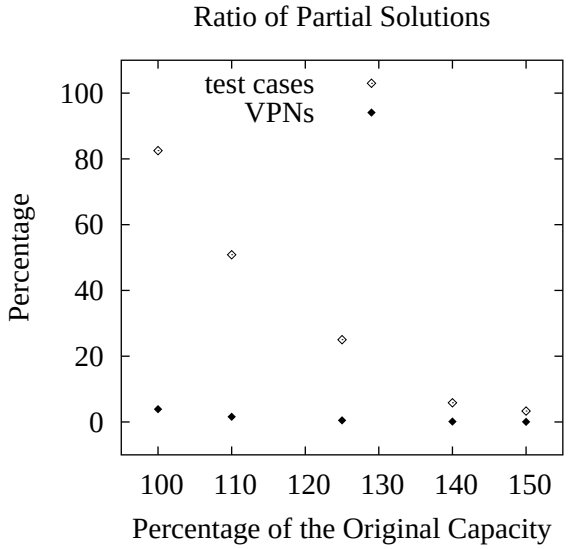


Figure 3.16: Path Based Heuristic – Relaxing the Capacity Bounds

The advanced combination of the capacity and topology objectives with the multiplier parameter was evaluated with 150% relaxed capacity bounds to get more full solutions and to have more space to the heuristics. Figures 3.17 and 3.18 show the results with different multiplier parameter settings. In the figures it can be seen that increasing the parameter value above 5 – emphasizing the topology objective – does not change the results significantly. Moreover, some metrics are not influenced to a great extent by the parameter: the ratio of full solutions is near to 100%, the VPN node coverage is decreasing only by a few percents, the capacity reservation is slightly increasing, and the time consumption is constantly low. However, the ratio of tree topology VPNs is increasing significantly from 25% to 60%, and also the average path length from 2.5 to approximately 3, while the VPN extension is reduced from 1.75 to 1.5. These results also illustrate that the three topology related metrics (tree topology VPNs, VPN node coverage, and VPN extension) measure different characteristic of the VPNs, because while the ratio of tree VPNs are changed, the other two metrics do not for some parameter settings.

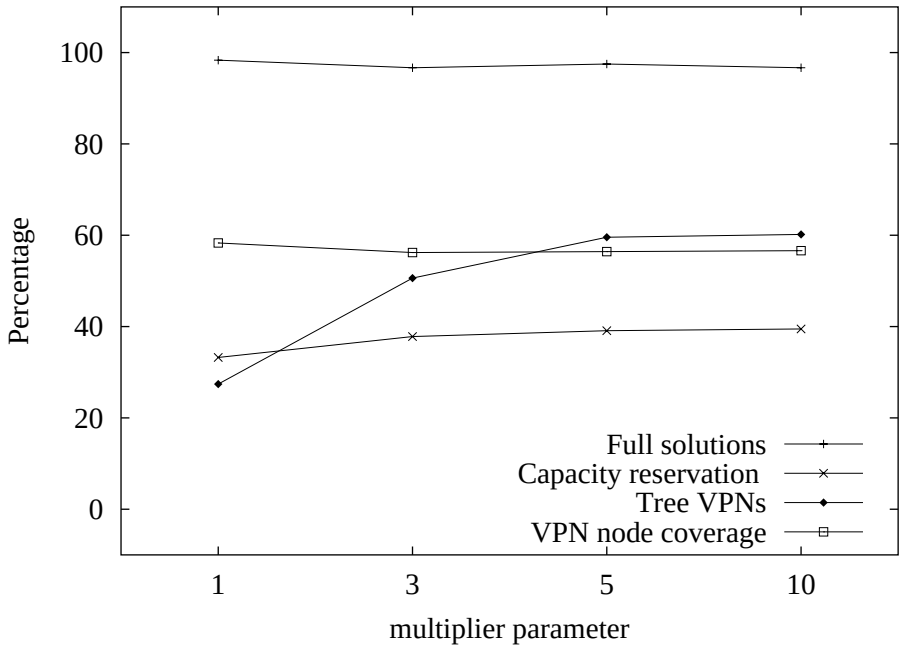


Figure 3.17: Effect of Weight Offset Multiplier – Metrics in Percentage

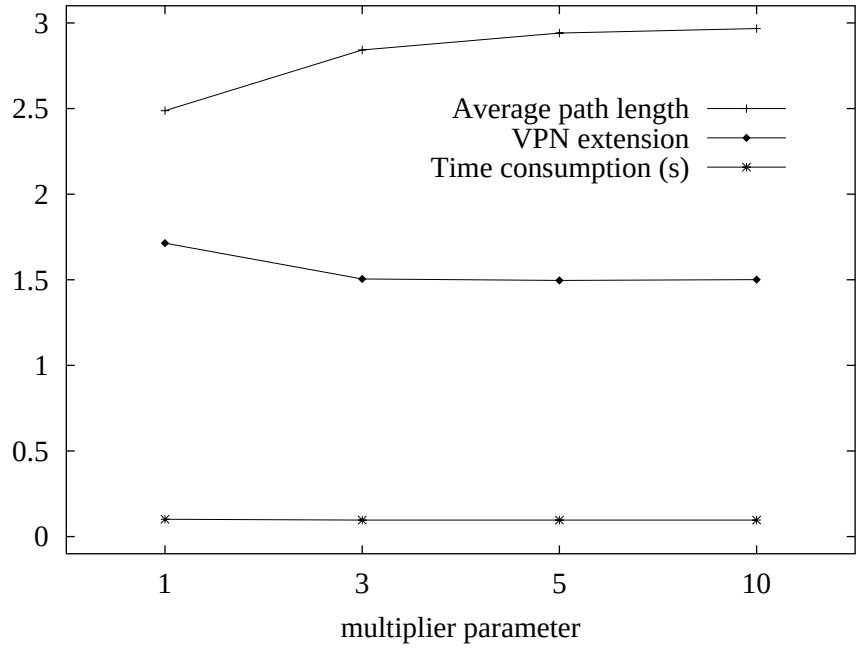


Figure 3.18: Effect of Weight Offset Multiplier – Metrics with Absolute Values

The Score Based Ordering significantly reduces the search space, therefore gives result in short time. However, to allow more options to test in the search space the following heuristics not only reserve, but also delete paths in a converging way. Therefore more variations are examined, which enhances the quality of the result but the running time is also longer.

Simulated Allocation Simulated allocation (SimAll) routes or deletes paths randomly, but in a converging way. It was proposed by Michal Pióro [PSG+00]. This version of the algorithm was adapted to VPN design. It is shown as pseudo-code in Algorithm 1. [**C9**, **J1**]

Algorithm 1 SimulatedAllocation()

$x =$ empty solution
$cost_{min} = INF$
$noimpr = 0$
repeat
  $noimpr = noimpr + 1$
  $p = random([0, 1])$
  if $p < q(x)$ **then**
    $Allocate(x)$
  else
    $Deallocate(x)$
  end if
  if $complete(x) \wedge (cost(x) < cost_{min})$ **then**
    $x_{min} = x$
    $cost_{min} = cost$
    $noimpr = 0$
  end if
  if $complete(x) \vee (cost(x) > cost_{min})$ **then**
    $Deallocate2(x)$
  end if
until $noimpr < M$
return x_{min}

SimAll starts from an empty solution, i.e. none of the traffic demands is routed. Then, in each step, it chooses between routing a demand or deleting some already routed ones. The function $q()$ determines the probability of the allocation (demand routing). This has to

be larger than 0.5 to ensure the convergence of the method. The parameter setting was: $q()$ returns 0.9 if less than 80% of the demands are routed, and 0.8 otherwise.

If allocation ($Allocate()$) is chosen, then one demand gets routed. The demand for routing is chosen in a probabilistic way from the set of the not yet routed demands. The probability of choosing a demand is proportional to its score. The score of the demand is calculated in the following way (the maximums are taken over all demands):

$$\text{Score} = \frac{1}{2}\left(1 - \frac{d}{d_{\max}}\right) + \frac{1}{2}\frac{b}{b_{\max}} \tag{3.2.6}$$

In equation 3.2.6 b denotes the bandwidth of the demand, and d denotes the distance of its endpoints. Thus, high bandwidth demands with near endpoints are preferred as previously at *Score based Ordering* (equation 3.2.1 with $\beta = 0.5$). The test runs have shown that with simulated allocation a balanced emphasis between the distance and bandwidth based scores performs better ($\beta = 0.5$), and it has also less importance because of the reserving–freeing steps.

If deallocation ($Deallocate()$) is chosen, then one or more demands get deleted. A probabilistic choice is made between deleting a single demand or deleting all the demands using a virtual or physical link. (A virtual link is a link of a VPN, i.e. a physical link consists of as many virtual links as many VPNs use that physical link.) The probabilities of these are respectively 0.85, 0.1 and 0.05 ($P_{demand} = 0.85, P_{vlink} = 0.1, P_{link} = 0.05$). The demand, the virtual link and the physical link for deletion is chosen using uniform distribution from the corresponding sets. The pseudo-code written in Algorithm 2 describes the deallocation.

Algorithm 2 Deallocate(x)

$p = random([0, 1])$
if $p < P_{link}$ **then**
  $DeallocateLink(x)$
else if $(P_{link} \leq p) \wedge (p < P_{link} + P_{vlink})$ **then**
  $DeallocateVirtualLink(x)$
else
  $DeallocateDemand(x)$
end if

When during the iterations a complete solution is found, i.e. all demands are routed, then it is compared to the best one found till now. If it is better, then the current best solution is updated. If the solution is complete or the cost of the partial solution is higher than the best cost obtained so far, then demands will be deleted until the number of routed demands decreases by 20% (N=0.8). The pseudo-code is shown in Algorithm 3.

Algorithm 3 Deallocate2(x)

$r = RoutedDemands(x)$
repeat
  $DeallocateVlink(x)$
until $RoutedDemands(x) > r * N$

If the solution could not be improved for a pre-specified number of iterations (e.g. M=1000), then the algorithm terminates and returns the best solution found so far (x_{min}).

This enhanced algorithm (SimAll) was compared with other variants. One of them is a slightly modified one (SimAllUni), where the demand was chosen for allocation with uniform distribution from the set of the not routed ones regarding only the reserved capacity. In another variant the demands were routed in randomly generated order and averaged by 50 random runs. As a reference in the fourth variant the demands were routed in descendent order of their score (ScoreOrder) (equation 3.2.6).

The results have shown that SimAll is better than SimAllUni, and ScoreOrder is better than the average of 50 random runs, but it is worse than the SimAlls [**J1**]. Thus the enhanced version of Simulated Allocation seems to be an effective demand ordering heuristic for this problem.

Results The first goal again is to achieve complete solution. This is realized by two parameters: the number of iterations and the degree of capacity relaxation. Increasing the number of iterations increases the running time, relaxing the capacity means additional costs. Table 3.8 shows the ratio of complete results and the time consumption by increasing the iteration count with the original capacity bounds.

Increasing the iteration count increases the running time linearly, however, even with

Iteration count	Ratio of complete solutions	Average time consumption (sec)
1,000	61.67%	2.7
10,000	81.67%	24
100,000	91.67%	257
1,000,000	94.17%	2436

Table 3.8: Increasing the Iteration Count in Simulated Allocation

1,000,000 iteration the ratio of complete solutions is still not 100%. Therefore the capacity bounds were relaxed. The tests has shown that only 5% of additional capacity and 10,000 iteration is sufficient to achieve 100% of complete solutions (see Table 3.9).

Additional capacity (%)	Iteration count	Ratio of complete solutions (%)
10	1,000	98.33
10	10,000	100.00
5	10,000	100.00
3	10,000	96.67

Table 3.9: Relaxing the Capacity Bounds for Simulated Allocation

With the combination of the parameters of Dijkstra's algorithm and the simulated allocation algorithm a similar balancing can be set up between the capacity and topology objectives like by the integer linear programming with the α parameter. Table 3.10 shows these parameter settings. If the base weight is the capacity to reserve, then the capacity minimization is included to some extent in the objective. If the offset is set to zero, then topology objective is excluded from the objective. If SimAll evaluates the actual state by the reserved capacity, then the capacity objective is emphasized. If SimAll evaluates the actual state by the number of virtual links, then it emphasizes the topology objective. By the last variant in the table the capacity is not explicitly included in the weighting, the 10:1 ratio between the not used and the already used edges drives the compacting of the topology.

In Figures 3.19 and 3.20 the results from these four objectives are depicted for the three test networks. The figures are similar to the results of the ILP (Figures 3.6 and 3.7).

	Base weight	Offset weight	SimAll evaluation objective
Capacity minimization	capacity to reserve	0	capacity
Combined minimization 1	capacity to reserve	10 * average bandwidth	capacity
Combined minimization 2	capacity to reserve	10 * average bandwidth	number of virtual links
Topology minimization	average bandwidth	10 * average bandwidth	number of virtual links

Table 3.10: Parameter Setting Variants for Simulated Allocation

The capacity reservation shows increasing tendency, the number of tree VPNs also and the VPN node coverage is decreasing. The first values in each group are the results for the capacity minimization where the topology objective is excluded. That is why the capacity reservation is lower compared to the result of ILP, because in ILP formulation the topology objective is always included to some extent (by $\alpha = 0.9$ with 0.1 weight). In Figure 3.20 the VPN extension is clearly decreasing with the emphasizing of the topology objective. However, the average path length is increasing, but with topology minimization for some networks it is less than combined objective 2. Comparing it with the ILP solution it can be seen that the change in the ILP solutions is significant with $\alpha \leq 0.01$, however with Simulated Allocation the results are already changing with the combined objectives, which indicates that they are closer to the topology minimization.

Table 3.11 shows how good is the Simulated Allocation with the topology minimization parameter settings compared to the ILP solution regarding the topology minimization. The

α		0.9		0.1		0.01		0.001	
tree VPNs (%)		8.56		10.53		30.77	41.37	66.35	
VPN extension		2.20		2.14		1.84	1.71	1.47	
VPN node coverage (%)		66.92		66.00	61.02	60.50		55.98	
Time consumption (sec)		32.11		33.76	75.28	156.56		1993.79	
Capacity reservation (%)		48.21		48.21		49.56	54.01	54.11	
Average path length		2.39		2.39		2.45		2.72	2.81

Table 3.11: Simulated Allocation Compared to ILP

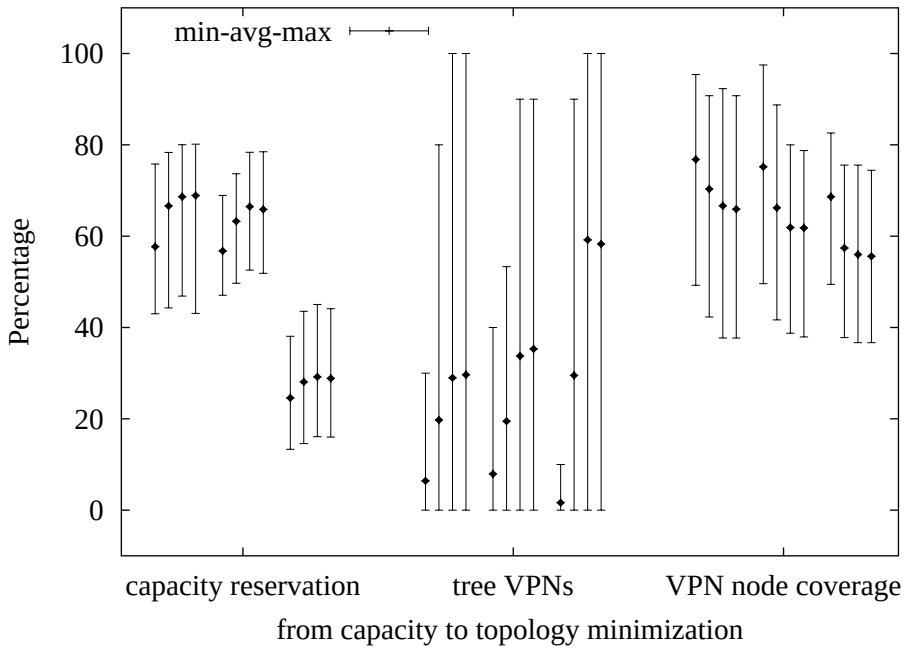


Figure 3.19: Simulated Allocation with Different Objectives – Metrics in Percentage

values are the average for all test cases on the three test networks. Most of the values are between α 0.01 and 0.001, that means the topology objective is emphasized, but not as much as with the ILP by $\alpha = 0.001$. The table shows that in shorter time (75 sec vs. 156 sec) better values can be reached regarding the ratio of tree VPNs and VPN extension than with ILP by $\alpha = 0.01$. The VPN node coverage is slightly worse than the ILP with $\alpha = 0.01$. The capacity reservation and the average path length is higher, but not significantly.

Comparing this table with Table 3.6 it can be seen that for a higher time consumption all other metrics are considerably better with Simulated Allocation than with Score Based Ordering.

To achieve better topology minimization SimAll requires some more capacity. Table 3.12 shows that regarding the ratio of tree VPNs and VPN extension with 25% capacity extension better results can be reached than with ILP with $\alpha = 0.001$. For VPN node coverage the values are decreasing and the difference is at 25% extension within 2%. The price for these are that the average path length are increasing, the difference is 0.15.

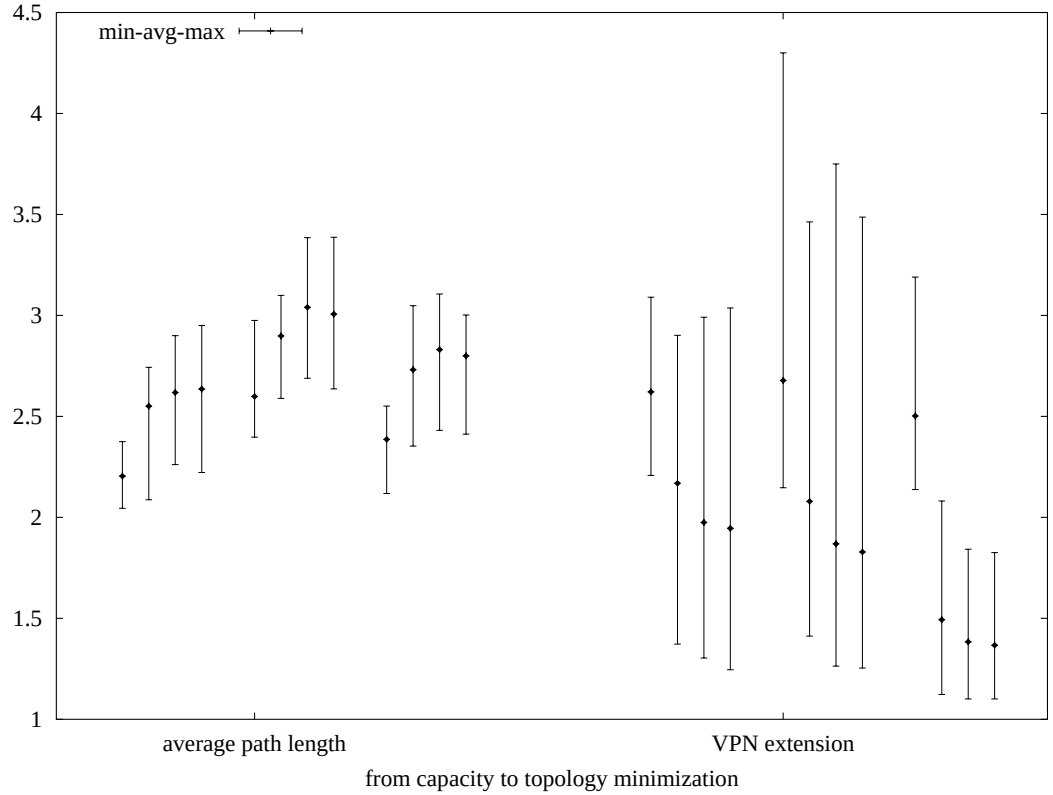


Figure 3.20: Simulated Allocation with Different Objectives – Metrics with Absolute Values

Capacity extension (%)	Ratio of tree VPNs (%)	VPN extension (%)	VPN node coverage	Average path length
5	41.37	1.71	61.02	2.81
10	49.89	1.61	59.05	2.84
25	70.92	1.45	56.00	2.87
ILP with $\alpha = 0.001$	66.35	1.47	54.11	2.72

Table 3.12: Simulated Allocation – Improving Topology Minimization with Capacity Relaxation

Chapter 4

Bandwidth Guaranteed VPN Design with Protection

To ensure reliability – which is an important factor for a VPN service – the design must be prepared for failures. Therefore, the VPNs will have some kind of redundancy.

To guarantee reliability two main method is applicable: protection and/or restoration. Protection reserves resources in advance to use them in case of failure. Restoration does not reserve resources in advance, it uses the available free resources at the time of failure. There are also combined solutions of course. Protection is pre-planned, therefore it is mainly used with static traffic patterns, like VPNs. Restoration is a good solution in changing traffic environments. In case of restoration no in advance planning is done.

Therefore, the adequate method is the *protection* for the VPN design. Because the demands are known in advance the protection can be planned in advance as well.

The protection methods have three main types: path, segment and link protection. Path protection reserves disjoint backup paths for primary paths. The backup path can be link or node disjoint with the primary path. If the backup path is node disjoint, then it involves that they are link disjoint as well.

Link protection defines a protection path for each link in the primary path. In case of failure the traffic between the terminating nodes of the failed link gets rerouted to the protection path.

Segment protection is an intermediate solution between path and link protection. The primary path is divided to segments, and the protection paths defined for these segments. In general, path protection is more capacity efficient and simpler to determine than link protection, which is a strong motive to deal only with path protection for VPNs.

If the primary path is split, it also delivers survivability to some extent: if a link failure is on one of the branches, then the other branch(es) can transmit part of the traffic. Therefore, path protection assumes that the primary path is unsplit, and calculates also an unsplit backup path, which is typical in telecommunication networks.

There are two types of protection schemes regarding the bandwidth reservation. The *dedicated* protection reserves the full capacity of the primary path for the backup path. This scheme protects the network against all failures affecting the primary paths, however, it requires significant additional bandwidth. Dedicated protection has two typical modes. In 1+1 protection mode the traffic is sent on both primary and backup paths. In 1:1 protection mode the traffic is only sent on primary path, and the backup is only used in case of failure.

It is not typical that many elements fail in the network at the same time. In case of a failure only those backup paths gets used for which the primary path failed, which is only a relative small part of total capacity reserved for dedicated protection. Primary capacity cannot be shared of course, but protection capacity can be shared as long as a single link failure does not affect more than one protection element. This is called *shared* protection scheme. The assumption is that in a given time only one failure is present is the network. This is a viable assumption because the time between failures is larger than the time required to repair a failure.

Therefore, the protection paths are planned in such a way that they can handle one arbitrary failure in the network. This allows that backup paths can share the reserved capacity when the failure is not on a common link of primary paths. This means, that if two primary paths do not have a common link, it is enough to reserve the maximum of the capacity of the two primary paths for backup path, because only one can fail in a given time. However, if two primary paths do have common links, the capacity reserved for backup path is the sum of the two primary paths, because if this common link fails, both primary paths fail.

This scheme effectively reduces the reserved backup capacity.

The goals are the capacity and topology objectives as well as by the unprotected VPN design. However, because there are primary paths and backup paths that are disjoint, the topology objective cannot result in tree topology, therefore this metric is omitted in the analysis of the results. On the other hand, compacting the protected topology still reduces the expansion of the VPNs.

4.1 Linear Programming Problem

The integer linear programming problem gets more complex in case of protected VPN design, because additional variables are presented for the protection paths.

As mentioned in the previous section, the primary and backup paths are not split, therefore the flow variables are binary in all of the following integer linear programming problems.

The integer linear programming is well suited to formulate the dedicated protection. Shared protection can be formulated as well, however, it gets even more computation demanding. It adds $|E||E-1|$ binary variables and constraints (see [BM03]).

Thus, I formulated only the dedicated protection problem to solve with pure integer linear programming. [**C5**]

4.1.1 Link Disjoint Path Level Dedicated Protection

The following variables are the solution variables associated with the problem: $X1_e^{d^p_{(i,j)}}$ indicates that edge e is in the first path that carries demand $d^p_{(i,j)}$, i.e. the traffic in VPN p between endpoints i and j. The short notation is: $X1_e^d$. $X2_e^{d^p_{(i,j)}}$ indicates that edge e is in the second path that carries demand $d^p_{(i,j)}$, i.e. the traffic in VPN p between endpoints i and j. The short notation is: $X2_e^d$. The $X1_e^d$ and $X2_e^d$ variables are interchangeable, therefore, from the first and the second path the not longer will be chosen for the *primary*, and the other for the *backup* path.

Y_e^p indicates whether VPN p has traffic on edge e, i.e. e is part of the virtual topology of VPN p, independently whether it is part of primary path, backup path or both.

The cost of the solution is proportional with the reserved total bandwidth and the number of virtually used edges. The balancing parameter α is also very important because values for the reserved bandwidth and the number of virtually used edges differ significantly. The amount of reserved bandwidth is typically greater even by orders of magnitude than the number of virtual links.

All the variables $X1_e^d$, $X2_e^d$ and Y_e^p are binary.

The ILP formulation is the following:

Variables:

$$X1_e^d \in \{0,1\} \tag{4.1.1}$$

$$X2_e^d \in \{0,1\} \tag{4.1.2}$$

$$Y_e^p \in \{0,1\} \tag{4.1.3}$$

Objective function:

$$\min \left\{ \alpha \sum_{\forall d \in D, \forall e \in E} (X1_e^d + X2_e^d) b_d + (1-\alpha) \sum_{\forall p \in P, \forall e \in E} Y_e^p \right\} \tag{4.1.4}$$

subject to

$$\sum_{\forall u:e=(u,v)} X1_e^d - \sum_{\forall w:e=(v,w)} X1_e^d = \begin{cases} -1 & \text{if } v \text{ is the source of } d, \\ 1 & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{4.1.5}$$

$$\sum_{\forall u:e=(u,v)} X2_e^d - \sum_{\forall w:e=(v,w)} X2_e^d = \begin{cases} -1 & \text{if } v \text{ is the source of } d, \\ 1 & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{4.1.6}$$

$$X1_e^d + X2_e^d \leq Y_e^p \quad \forall (i,j) : d_{(i,j)}^p \in D, \forall e \in E, \forall p \in P \tag{4.1.7}$$

$$\sum_{\forall d \in D} (X1_e^d + X2_e^d) b_d \leq b_e \quad \forall e \in E \tag{4.1.8}$$

In equation 4.1.4 if $\alpha = 0$, then only the number of virtual edges are minimized, while if $\alpha \neq 0$, then the bandwidth reservation is also considered. Because $X1_e^d$ and $X2_e^d$ are binary, the sum of them is multiplied by the bandwidth of the demand b_d.

Equation 4.1.5 and 4.1.6 are flow conservation constraints. They ensure that traffic coming in and going out is the same at each node, when it is not the source or the destination of the flow, both for the primary and backup paths. Because the variables are binary, the right side of the equation is $-1, 0, 1$ instead of the amount of bandwidth.

Equation 4.1.7 is a diversity constraint which expresses that when edge e is part of VPN p, i.e. $Y_e^p = 1$, then exclusively the primary or the backup path (or neither) can carry traffic on edge e. Thus, at most one out of $X1_e^d$ and $X2_e^d$ can be 1. However, if edge e is not part of VPN p, i.e. $Y_e^p = 0$, then it cannot carry traffic for demand d, thus $X1_e^d = 0$ and $X2_e^d = 0$. In the constraint for a given Y_e^p only those demands are selected that belong to VPN p, denoted as $\forall (i,j) : d_{(i,j)}^p$.

The constraint in equation 4.1.8 ensures that the bandwidth utilization together for primary and backup paths does not exceed the physical bounds.

Complexity measured by ILP formulation:

Number of binary variables:	$\|E\|(2\|D\| + \|P\|)$
Number of constraints:	$2\|V\|\|D\| + \|E\|(\|D\| + 1)$

Results

To guarantee bandwidth for the backup paths the capacity bounds need to be relaxed. The question is how much extra capacity is required to achieve, that all primary paths have a backup path with the same bandwidth guarantees. If the capacity bounds were doubled only 40% of the test cases was feasible. I examined what kind of test configurations require more capacity for protection. The following traffic characteristics caused the most unfeasible

cases:

- many VPNs (15 or 10)
- all sort of VPN sizes
- constant demand sizes

Because the traffic volume was approximately the same in all traffic files, many VPNs means also more demands with smaller bandwidth requirement. Thus, this kind of traffic configurations require more capacity for protection.

To be able to solve almost all test cases (more than 97%) the capacity bounds were increased to 2.5 times of the initial.

Figures 4.1 and 4.2 show the results with this capacity bound setting. Because protected VPN configurations could not form tree topology, therefore this metric is omitted. In figure 4.1 the primary and protection capacity reservation is depicted separately, and also the total capacity reservation. It can be seen that pushing the topology objective (decreasing α) affects mainly the backup capacity reservation, the primary capacity reservation is almost constant. The total capacity reservation is similar to the capacity reservation for the not protected VPNs with the original capacity bounds (see figure 3.6). However, the VPN node coverage is in the 65–90% interval instead of 55–75%. This indicates, that to realize two link disjoint paths between each VPN endpoints the paths have to go through 10–15% more nodes in the network. More emphasis on the topology objective is realized with $\alpha = 0.001$, smaller α values cannot achieve significant decrease. The price for the VPN node coverage minimization is the increased capacity reservation. The figure shows that a good compromise is the $\alpha = 0.01$ setting, where the capacity reservation increase is minimal while the VPN node coverage is definitely reduced.

The result for the paths are similar to the capacity reservation results. The path length increase is more significant for the backup paths than for the primary paths by enforcing the topology objective, however, there is an increase also in primary path length. The average total path length is higher (approximately 3–4) compared to the result of the not protected configuration (approximately 2.25–3). The VPN extension is about twice as much compared to the not protected VPN configuration (2.5–1.25 vs. 3.5–2.25), and it can be

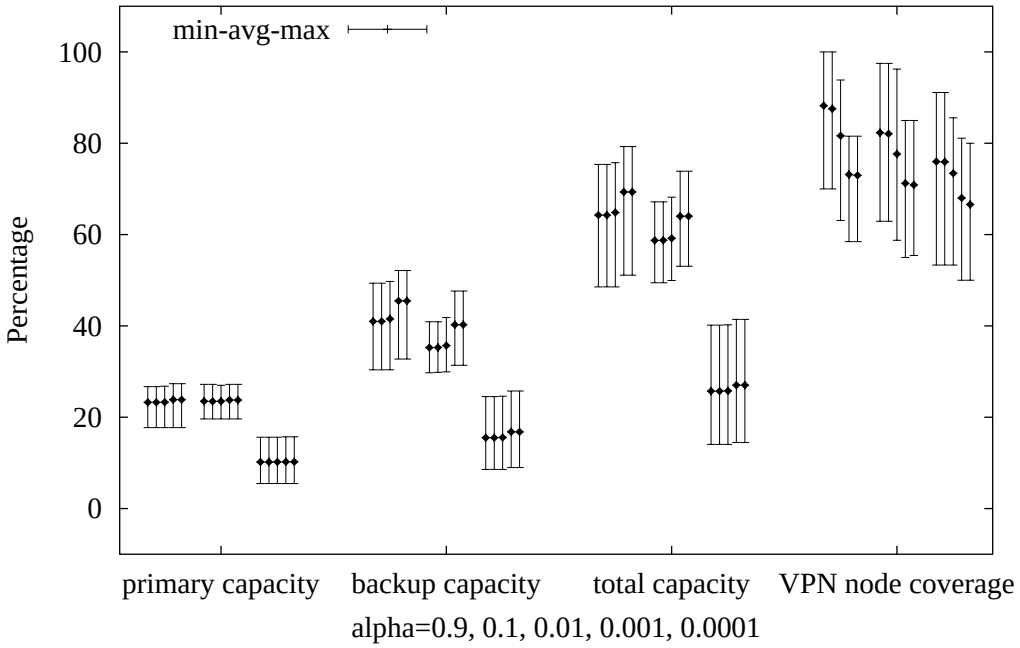


Figure 4.1: Link Disjoint Dedicated Protection with ILP – Metrics in Percentage

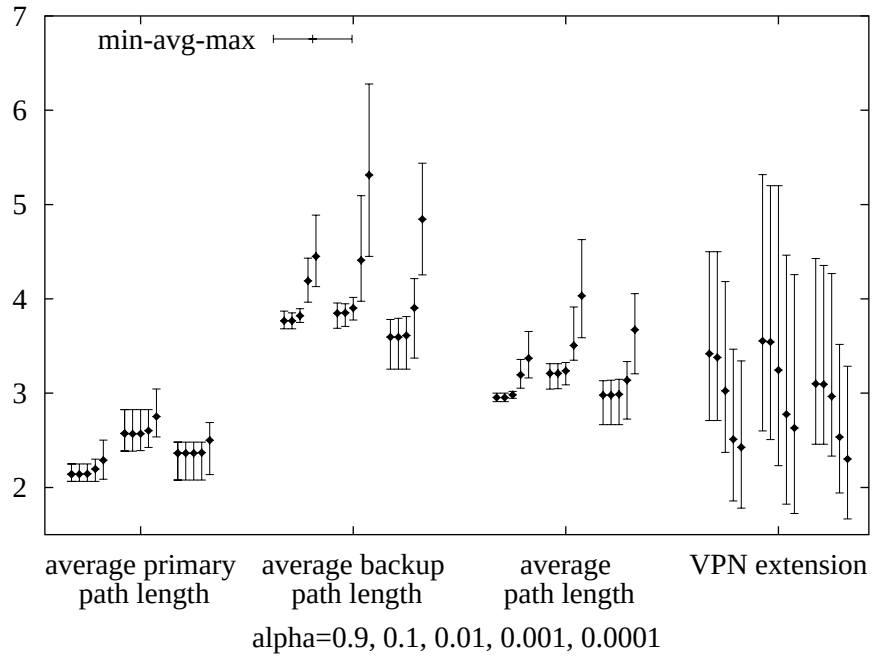


Figure 4.2: Link Disjoint Dedicated Protection with ILP – Metrics with Absolute Values

reduced as in the case of not protected VPNs (see Figure 3.4 and Figure 4.2). This figure also illustrates that the $\alpha = 0.01$ setting is a good compromise.

Including dedicated protection to the VPN configuration requires 2.5 times more capacity. The topology objective metrics causes increase in the backup capacity reservation and average backup length, while the primary capacity reservation and average primary path length is almost constant.

The average time consumption was dependent on α parameter. For $\alpha \geq 0.01$ the average time consumption was 1.5–2 minutes, for $\alpha = 0.001$ around 14 minutes, and for $\alpha = 0.0001$ it was around 47 minutes. In the last two cases there were significant number of test cases that has reached the one hour time limit, which was set to be able to evaluate over 100 test cases with each α setting. This indicates again that enforcing the topology objective requires more computation, at least one order of magnitude higher.

4.1.2 Node Disjoint Path Level Dedicated Protection

This protection mode adds only one more constraint (equation 4.1.9) to the ILP formulation of Link Disjoint Path Level Protection. This additional constraint expresses that primary and backup paths must not have common nodes except the source and the destination node.

$$\sum_{\forall v: e=(u,v) | \{u \neq i\}} (X1_e^d + X2_e^d) \leq 1 \quad \forall u \in V, \forall d_{(i,j)}^p \in D \tag{4.1.9}$$

This increases the number of constraints with $|D|$.

Results

Each node in the test networks have at least two connecting edges, i.e. two link disjoint paths can always be found. However, this condition is not sufficient for finding two node disjoint paths. The third, GÉANT test network has a “triangle”, which triangle is connected only with one node to the rest of the network. (5 Norway – 6 Denmark – 10 Sweden: see in the Appendix) The other two nodes in the triangle cannot be connected to any other nodes

with two node disjoint paths, because these paths would cross the only one connecting node, therefore cannot be node disjoint. This means that for node disjoint protection the structure of the network needs to be analyzed, too.

If the network structure permits the node disjoint paths, the results can be compared to the link disjoint case. The test runs showed that there are only very small differences (maximum 0.7% by metrics in percentage and maximum 0.05 by absolute metrics) between the results of link and node disjoint configurations, therefore the numerical results are omitted.

4.1.3 VPN Level Dedicated Protection

VPN level protection handles the protection from the point of view of the VPN customer. The VPN customer does not know the internal topology of the physical network, only the connectivity among the endpoints of his VPN. If the VPN customer wants to handle the protection on its own, it may request a primary and a backup VPN. If the connectivity fails in the primary VPN, then the traffic is switched to the backup VPN without service interruption.

The service provider gets a notice about the failure. Then, after the service provider identifies and fixes the failure, it informs the VPN user that he/she can use the primary VPN again.

By VPN level protection the primary VPN (the links that form the VPN, the virtual links) will be protected and not each traffic demand separately. The primary and backup VPNs are disjoint, the common elements are only the VPN endpoints. I investigates the link disjoint case for the VPN level protection. This protection concerns the whole VPN and not the individual paths. [**C5**]

The following variables are the solution variables associated with the problem: $X1_e^{d^p_{(i,j)}}$ indicates that edge e is in the *primary* path that carries demand $d^p_{(i,j)}$, i.e. the traffic in VPN p between endpoints i and j. The short notation is: $X1^d_e$. $X2_e^{d^p_{(i,j)}}$ indicates that edge e is in the *backup* path that carries demand $d^p_{(i,j)}$, i.e. the traffic in VPN p between endpoints i and j. The short notation is: $X2^d_e$.

$Y1^p_e$ indicates whether primary VPN p has traffic on edge e, i.e. e is part of the virtual

topology of primary VPN p. $Y2_e^p$ indicates whether backup VPN p has traffic on edge e, i.e. e is part of the virtual topology of backup VPN p.

The cost of the solution is proportional to the reserved total bandwidth and the number of virtually used edges. The balancing parameter α is also very important because values for the reserved bandwidth and the number of virtually used edges differ significantly. The amount of reserved bandwidth is typically greater even by orders of magnitude than the number of virtual links.

All the variables $X1_e^d$, $X2_e^d$, $Y1_e^p$ and $Y2_e^p$ are binary.

The ILP formulation is the following:

Variables:

$$X1_e^d \in \{0,1\} \tag{4.1.10}$$

$$X2_e^d \in \{0,1\} \tag{4.1.11}$$

$$Y1_e^p \in \{0,1\} \tag{4.1.12}$$

$$Y2_e^p \in \{0,1\} \tag{4.1.13}$$

Objective function:

$$\min \left\{ \alpha \sum_{\forall d \in D, \forall e \in E} (X1_e^d + X2_e^d) b_d + (1-\alpha) \sum_{\forall p \in P, \forall e \in E} (Y1_e^p + Y2_e^p) \right\} \tag{4.1.14}$$

subject to

$$\sum_{\forall u: e=(u,v)} X1_e^d - \sum_{\forall w: e=(v,w)} X1_e^d = \begin{cases} -1 & \text{if } v \text{ is the source of } d, \\ 1 & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{4.1.15}$$

$$\sum_{\forall u: e=(u,v)} X2_e^d - \sum_{\forall w: e=(v,w)} X2_e^d = \begin{cases} -1 & \text{if } v \text{ is the source of } d, \\ 1 & \text{if } v \text{ is the destination of } d, \\ 0 & \text{otherwise.} \end{cases} \quad \forall v \in V, \forall d \in D \tag{4.1.16}$$

$$\sum_{\forall d \in D} (X1_e^d + X2_e^d) b_d \leq b_e \quad \forall e \in E \tag{4.1.17}$$

$$X1_e^d \leq Y1_e^p \quad \forall (i,j) : d^p_{(i,j)} \in D, \forall p \in P, \forall e \in E \tag{4.1.18}$$

$$X2_e^d \leq Y2_e^p \quad \forall (i,j) : d^p_{(i,j)} \in D, \forall p \in P, \forall e \in E \tag{4.1.19}$$

$$Y1_e^p + Y2_e^p \leq 1 \quad \forall p \in P, \forall e \in E \tag{4.1.20}$$

In equation 4.1.14 if $\alpha = 0$, then only the number of virtual edges are minimized, while if $\alpha \neq 0$, then the bandwidth reservation is also considered. Because $X1_e^d$ and $X2_e^d$ is binary the sum of them is multiplied by the bandwidth of the demand b_d.

Equations 4.1.15 and 4.1.16 are the flow conservation constraints. They ensure that traffic coming in and going out is the same at each node, when it is not the source or the destination of the flow, both for the primary and backup paths. Because the variables are binary the right side of the equation is $-1, 0, 1$ instead of the amount of bandwidth.

The constraint in equation 4.1.17 ensures that the bandwidth utilization together for primary and backup paths does not exceed the physical bounds.

Equation 4.1.20 is a diversity constraint which expresses that when edge e is part of primary VPN p, i.e. $Y1_e^p = 1$, then it cannot be part of backup VPN and vice versa. Thus, at most one out of $Y1_e^p$ and $Y2_e^p$ can be 1.

If an edge is part of the primary path for a demand d in VPN p, i.e. $X1_e^d = 1$, then that edge belongs to primary VPN p, i.e. $Y1_e^p = 1$. However, if edge e is not part of primary VPN p, i.e. $Y1_e^p = 0$, then it cannot carry traffic for demand d, thus $X1_e^d = 0$. The same is valid for backup VPNs. This is expressed in equations 4.1.18 and 4.1.19. In the constraints for a given $Y1_e^p$ (or $Y2_e^p$) only those demands are selected that belong to VPN p, denoted as $\forall (i,j) : d^p_{(i,j)}$.

Complexity measured by ILP formulation:

Number of binary variables:	$2\|E\|(\|D\| + \|P\|)$
Number of constraints:	$2\|V\|\|D\| + \|E\| + 3\|P\|\|E\|$

Results

The VPN level protection is more dependent on the network and VPN structure than the node disjoint protection scheme. Only about 15% of the test cases were feasible. The same ratio remained feasible even if the capacity bounds were relaxed to infinite. This indicates that VPN level protection can be applied only in certain cases (a few number of small size VPNs), therefore it is not a general kind of protection. It can be used only if the network and the VPN traffic structure allows it.

4.2 Heuristics

Because the numerical results of link and node disjoint design modes were similar with the ILP design method, in the following only the link disjoint design modes are evaluated with the heuristic design methods.

4.2.1 VPN Based Heuristic

As in section 3.2.1 is detailed, the integer linear programming problems can be divided to VPNs, thus reducing the complexity. By protected VPN design the same principle is used. By solving the integer linear programming problem for one VPN both the primary and backup paths will be determined. The order in which the VPNs are calculated is determined by the scoring system. The preference function of the scoring system was analyzed again. Table 4.1 shows that the differences between the preference modes are small regarding the ratio of partial results depending on different α settings. Preferring high bandwidth demands with near endpoints instead of low bandwidth demands with far endpoints gives slightly less partial solutions. Therefore, in the results the demands with high bandwidth and near endpoints will be preferred by the scoring system. [**C7**]

α	Prefer high bandwidth with near endpoints	Prefer low bandwidth with far endpoints
0.9	34.1%	35.5%
0.1	34.1%	34.1%
0.01	34.8%	34.8%
0.001	63.6%	64.3%

Table 4.1: Ratio of Partial Results Depending on Preference Modes with Unsplittable Flows for Protected VPN Design

Results

The VPN based heuristic does not provide a full solution in all cases as by the not protected VPN configuration.

Figure 4.3 shows the ratio of partial solutions and also the ratio of not planned VPNs depending on α parameter. If the topology objective is emphasized ($\alpha \leq 0.001$) the superposition of locally optimized VPN design hinders more the accommodation of lower ranked VPNs, but higher α values do not influence significantly the ratio of partial solutions.

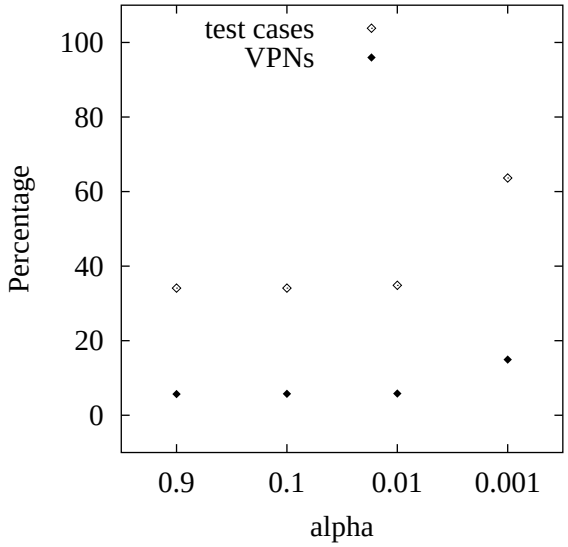


Figure 4.3: VPN Based Heuristic for Protected VPNs – Percentage of Partial Solutions

To reduce the ratio of partial solutions the capacity bounds were relaxed again. Figure 4.4 shows that a small capacity extension significantly decreases the ratio of partial solution, and 50% extension (from 250% to 375%) makes all solutions complete.

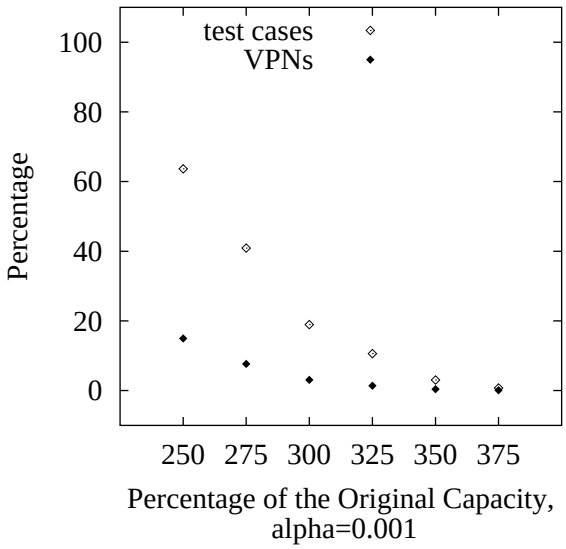


Figure 4.4: VPN Based Heuristic for Protected VPNs – Relaxing the Capacity Bounds

For fair comparison with the results of the Integer Linear Programming, I have chosen such an α parameter value, that produces enough complete solution with the heuristics. The comparison of the results can be seen in Figures 4.5 and 4.6 by $\alpha = 0.01$ with original (for protected configurations 250% of the unprotected capacity) capacity bounds. Only those results are included in the statistics where the VPN based heuristic could achieve complete solutions (approximately 65%) for fair comparison.

It can be seen that the difference is by the metrics in percentage within 2% and by the metrics with absolute values within 0.1, i.e. if the VPN based heuristic is able to provide complete solution, it is very close to the global optimum and the time consumption is significantly lower: approximately 1.5–3 minutes for the examined α values instead of starting form 1.5 and increasing to 47 minutes with decreasing α by the global ILP (see end of subsection 4.1.1).

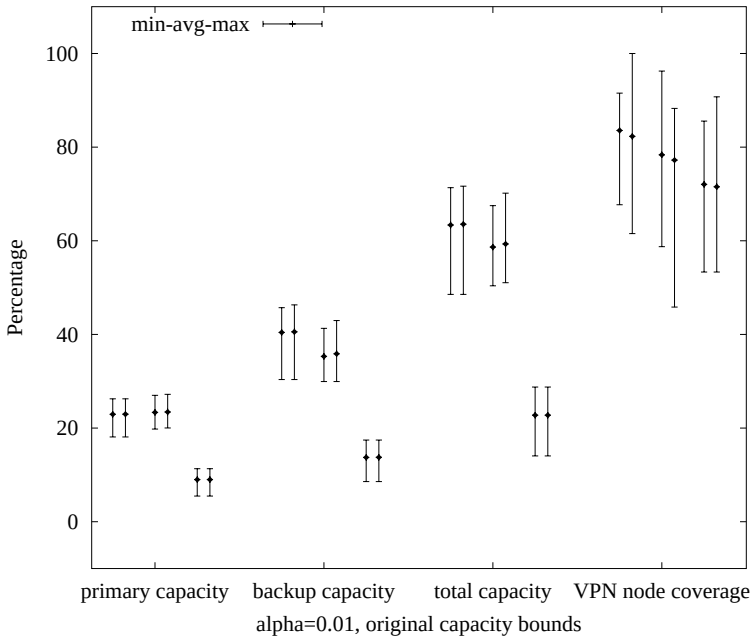


Figure 4.5: Comparison of VPN Based Heuristic and ILP Solution for Protected VPNs – Metrics in Percentage

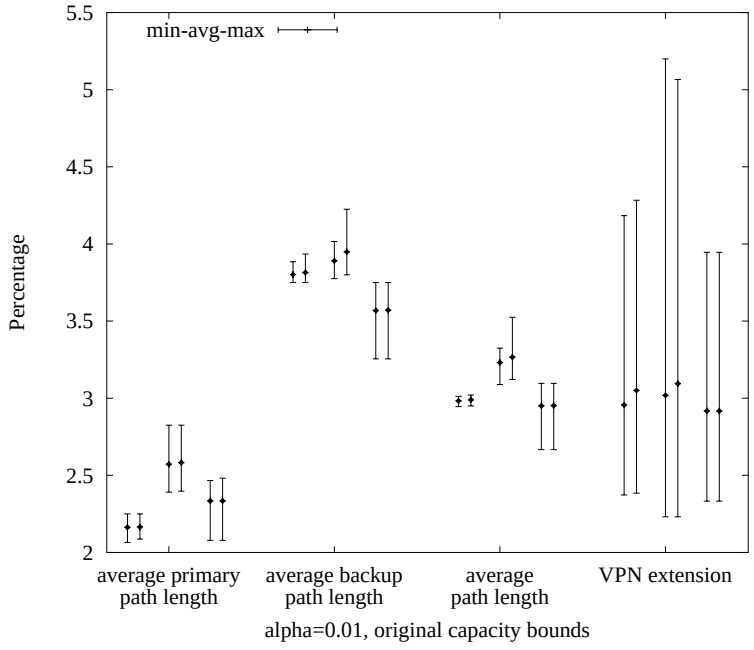


Figure 4.6: Comparison of VPN Based Heuristic and ILP Solution for Protected VPNs – Metrics with Absolute Values

4.2.2 Path Based Heuristics

In path protection two edge (or node) disjoint paths must be determined for a demand. The path based heuristics search primary and backup path with shortest path algorithms.

Shortest Path Algorithms

Double Dijkstra's Algorithm Dijkstra's algorithm is used to route not only the primary paths but the backup paths too. At first, the primary path is calculated with Dijkstra's shortest path algorithm. Then, the links of the primary path are temporarily deleted from the network to ensure link disjointness and Dijkstra's algorithm is applied again. [**C7**, **C8**, **W1**]

This solution gives the shortest path for the primary path. The backup path will be also the shortest path, however, since the edges of the primary path are deleted, the backup path can be much longer than the primary path.

A drawback of this method is that deleting the edges of the primary path can prevent finding the backup path.

Suurballe's Algorithm Suurballe's algorithm [ST84] searches the shortest pair of edge disjoint paths simultaneously in polynomial time. Therefore, the result is the shortest regarding together the primary and backup paths, thus it differs from the result of Double Dijkstra's algorithm. From the two path the not shorter will be the primary, and the other the backup. [**C7**]

Weight Assignment

Dedicated Protection In case of dedicated protection, because the backup paths are similar to primary paths – i.e. they connect the same endpoints, and the reserved bandwidth along the path is also equal –, the same combined capacity and topology objective based weight assignment scheme is used as by the VPN design without protection described in section 3.2.2 both for primary and backup paths. [**C7**, **J1**]

Shared Protection In case of shared protection the calculation of backup paths differs from the primary paths. However, the weight assignment used with dedicated protection is also applicable for primary and backup paths too. For primary path calculation the edge weight may have two different possible values: it is either the bandwidth of the demand or the bandwidth of the demand plus the offset. However, for backup path calculation the weights can have more and different values, because the capacity to be reserved depends on the actual shared protection paths. It can be even zero, if the primary path can be protected with no additional capacity reservation on that particular edge, because of sharing the backup path with an already established one. [**J1**, **C8**, **C9**]

On a particular edge the capacity reserved for protection is shared among multiple primary paths. The more primary path shares the protection capacity the more effective is the capacity reduction.

Demand Ordering

The same scoring scheme is used as by the VPN design without protection described in section 3.2.2. The only difference is that both the primary and the backup paths are calculated for a selected demand. [**C7**] The algorithm takes the demands one-by-one according to their scores, and for each demand the primary and backup paths are determined with the double Dijkstra's or with the Suurballe's algorithm.

Results

With the combinations of the

- shortest path algorithms: Dijkstra's or Suurballe's
- protection type: dedicated or shared
- path calculation: with score based ordering or with Simulated Allocation

different design variants can be built. The following four variants were evaluated. In the first comparison Suurballe's and the double Dijkstra's algorithm were chosen, because Suurballe's algorithm finds the shortest pair of paths, thus differs from the result of the double

Dijkstra's algorithm.

Suurballe's algorithm simultaneously searches the primary and backup paths. With the double Dijkstra's algorithm I calculated both the primary and the backup paths for a selected demand. Another possible way is to calculate all primary paths at first, and then all backup paths. This was examined in [**C8**] and was found to perform worse than calculating the backup path right after the primary path for each demand.

The double Dijkstra's and Suurballe's algorithm was combined with the score based ordering and with dedicated protection.

By the not protected VPNs the most promising heuristic was the Simulated Allocation, therefore it was analyzed, combined with dedicated and also with shared protection, incorporating the double Dijkstra's algorithm. The Simulated Allocation is used in the same way as by the not protected VPN design. The demands are routed and deleted randomly, in a convergent way, but both the primary and the backup path will be determined or erased in one allocation/deallocation step. The demand scoring system is the same as by the not protected design, only the demands get scores. The primary and backup paths are determined in one allocation step, i.e. this score is used for both primary and backup paths.

Double Dijkstra's and Suurballe's Algorithm with Score Based Ordering I examined the scoring variants again according to the distance to bandwidth ratio and the preference modes. Tables 4.2 and 4.3 show the results for the Suurballe's algorithm (the results of double Dijkstra's algorithm have the same tendencies). In Table 4.2 it can be seen, that the ratio of not routed demands is the lowest, when only the distance of endpoints is taken into consideration and the near endpoints are preferred (1.18), however, the differences are within 1–2% compared to the other variants. In Table 4.3 the ratio of the complete test cases is shown, where the most complete solution (64.44%) is achieved with distance to bandwidth ratio 0.75 by preferring the demands with high bandwidth and near endpoints. The difference is more significant, around 4%, compared to the next lowest. Therefore, the chosen parameters used in the evaluations for Score Based Ordering were the same as by the unprotected case, i.e. the distance to bandwidth ratio (β) is 0.75, and the high bandwidth demands with near endpoints were preferred (see section 3.2.2).

Preference mode		Distance to bandwidth ratio				
Endpoints	Bandwidth	0	0.25	0.5	0.75	1
far	high	2.35	2.58	2.59	2.46	2.34
far	low	1.43	1.76	1.95	2.27	2.34
near	low	1.43	1.28	1.19	1.19	1.18
near	high	2.35	2.11	1.76	1.49	1.18

Table 4.2: Effects of the Scoring Variants on the Not Routed Demands with Suurballe's Algorithm (%)

Preference mode		Distance to bandwidth ratio				
Endpoints	Bandwidth	0	0.25	0.5	0.75	1
far	high	54.81	53.33	55.56	52.59	54.81
far	low	51.11	51.11	51.11	54.07	54.81
near	low	51.11	55.56	57.04	58.52	60.74
near	high	54.81	57.04	60.74	64.44	60.74

Table 4.3: Effects of the Scoring Variants on the Complete Solutions with Suurballe's Algorithm (%)

The initial capacity for the protected configurations (250% of the not protected capacity) was not enough to solve all configurations completely by the heuristic. The ratio of complete solutions is depicted in Figures 4.7 and 4.8 depending on the multiplier parameter (m) and the additional capacity extension. Two values for the multiplier parameter was examined, in case of 1 the topology objective is not emphasized, in case of 10 the topology objective is emphasized. The number of not routed demands is also shown: even if the ratio of complete solutions is only 25–30%, the ratio of not routed demands is under 5–7%. The curves indicate that the results of the two methods is similar: only a few percent of the demands was failed to route, which diminishes near to zero as the capacity bounds are relaxed. The ratio of complete solutions is increasing with the capacity bound relaxation, however it could reach 100% only with emphasis on the capacity objective ($m = 1$) by 150% capacity relaxation. Comparing the two methods Suurballe's algorithm yields somewhat better results, i.e. more complete solutions and less not routed demands, but the difference is not significant.

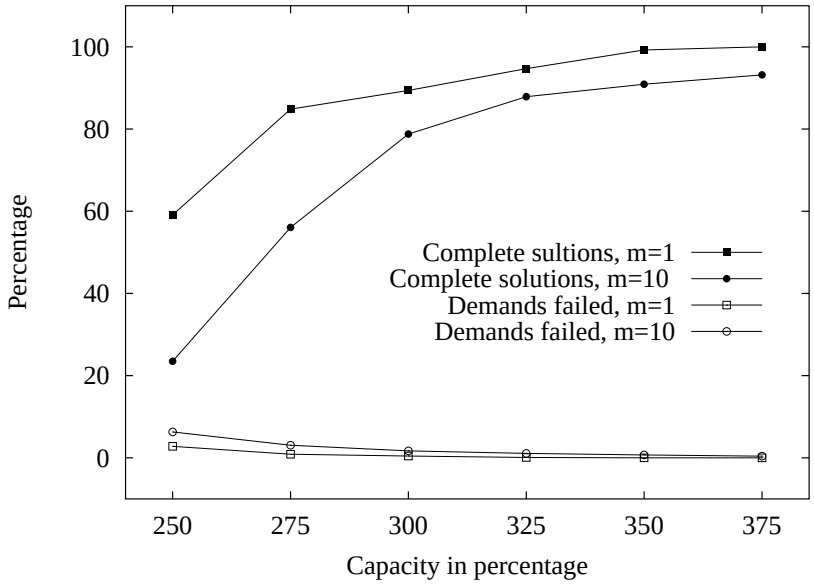


Figure 4.7: Ratio of Complete Solutions and Not Routed Demands – Double Dijkstra's Algorithm

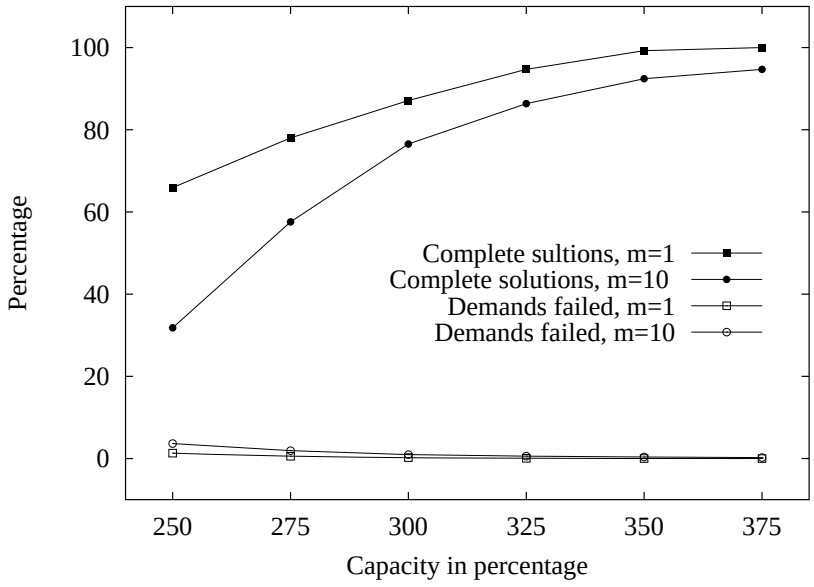


Figure 4.8: Ratio of Complete Solutions and Not Routed Demands – Suurballe's Algorithm

Tables 4.4 and 4.5 show the results of the heuristics compared to the results of the Integer Linear Programming solution based on the α capacity–topology balancing parameter. The results of the heuristics are in bold frames. The results are with $m = 10$ parameter setting, i.e. the topology was tried to be minimized. As it can be seen from the tables, the topology related metrics, VPN extension, and VPN node coverage realized only a modest degree falling close to $\alpha = 0.1$, i.e. the heuristics with the initial capacity bounds are not able to realize the compacting of the topology. This comparison also shows that Suurballe's algorithm is slightly better, since the VPN extension is lower. Even when the topology is not minimized, the capacity reservation and the path length is high, as by the ILP solution with emphasized topology objective. However, it must be noted that the difference between the ILP solution with emphasizing the capacity objective ($\alpha = 0.9$) and results of the heuristics is maximum around 4% at the capacity reservation and 0.3 at the path length. Altogether, the heuristics are not able to achieve the topology minimization, but the capacity objective is fairly approximated in at least one order of magnitude shorter time (seconds instead of minutes).

α		0.9		0.1		0.01		0.001		0.0001	
Primary capacity		18.88		18.88		18.90		19.18	19.98	20.14	
Backup capacity		30.34		30.36		30.69	32.74	33.92		38.86	
Total capacity		49.23		49.24		49.59	52.73	53.10		59.00	
Primary path length		2.36		2.36		2.36		2.39		2.52	2.55
Backup path length		3.74		3.74		3.78		4.17	4.19	4.88	
Path length		3.05		3.05		3.07		3.28	3.37	3.70	
VPN extension		3.36		3.34	3.25	3.08		2.63		2.45	
VPN node coverage		82.04		81.72	80.11	77.47		70.75		70.09	
Time consumption	1 sec	1.5-2 min		1.5-2 min		1.5-2 min		14 min		47 min	

Table 4.4: Double Dijkstra's Algorithm Compared to ILP Solution

α		0.9		0.1		0.01		0.001		0.0001	
Primary capacity		18.88		18.88		18.90		19.18		20.14	20.60
Backup capacity		30.34		30.36		30.69	32.75	33.92		38.86	
Total capacity		49.23		49.24		49.59		53.10	53.35	59.00	
Primary path length		2.36		2.36		2.36		2.39		2.52	2.54
Backup path length		3.74		3.74		3.78		4.17	4.20	4.88	
Path length		3.05		3.05		3.07		3.28	3.35	3.70	
VPN extension		3.36		3.34	3.21	3.08		2.63		2.45	
VPN node coverage		82.04		81.72	80.03	77.47		70.75		70.09	
Time consumption	0.25 sec	1.5-2 min		1.5-2 min		1.5-2 min		14 min		47 min	

Table 4.5: Suurballe's Algorithm Compared to ILP Solution

As with the not protected configuration, it was assumed that relaxing the capacity bounds could improve the topology related metrics. Table 4.6 shows the results when the capacity bounds were relaxed to 150% of the initial values. The topology metrics, VPN extension, and VPN node coverage get closer to the ILP solution, but the difference is still significant 0.2–0.4 and approximately 5%, not as by the not protected case (Table 3.7: 0.04 and 0.63%). However, the values for capacity reservation and path lengths are close to the ILP solution as by the unprotected case. Because Suurballe's algorithm searches the shortest pair of edge disjoint paths the total backup capacity and total path length are slightly lower than the result of Dijkstra's algorithm, however as a side effect of this, the primary capacity and path length are higher and the backup capacity and backup path length are lower for Dijkstra's algorithm results than for Suurballe's algorithm results. Thus, these results are in correspondence with the theoretical expectations coming form the definitions of the two algorithms.

Metrics	ILP solution Initial capacity (250%) $\alpha = 0.001$	Heuristic solution Double Dijkstra 150% of initial capacity (375%)	Heuristic solution Suurballe 150% of initial capacity (375%)
VPN extension	2.63	3.03	2.84
VPN node coverage (%)	70.75	76.23	75.03
Primary capacity (%)	19.18	20.97*	20.35*
Backup capacity (%)	33.92	33.84*	34.08*
Total capacity (%)	53.10	54.81*	54.43*
Primary path length	2.39	2.58	2.50
Backup path length	4.17	4.16	4.20
Total path length	3.28	3.37	3.35

*the actually reserved capacity is multiplied by 1.5 for comparison

Table 4.6: Topology Minimization with Protected Path Based Heuristics

Dedicated Protection with Simulated Allocation Simulated Allocation required only 5% capacity extension and iteration limit set to 10,000 to successfully route all demands by the not protected VPN configurations. Unfortunately, in case of VPNs with dedicated

protection, where two disjoint paths has to be found with the same capacity, this is not enough.

Relaxing the capacity bounds improves the results, however, Table 4.7 shows the decrease in the ratio of partial solutions by 10,000 iteration count with topology minimization objective. The table shows that 40% of extra capacity extension is required to completely solve all test cases. Although the ratio of not completely solved test cases is significant by the original capacity bounds, the ratio of not routed demands is under 1%.

Capacity bounds	Ratio of not completely solved test cases	Ratio of not routed demands
original (250%)	15%	0.8%
+10% (275%)	9%	0.2%
+20% (300%)	5%	0.05%
+30% (325%)	1.5%	0.01%
+40% (350%)	0%	0%

Table 4.7: Simulated Allocation for Protected VPNs – Relaxing the Capacity Bounds

Since the number of not routed demands remains under 1% at the original capacity bounds (250%) in the following comparisons the results are computed with this capacity bound. Table 4.8 positions the Simulated Allocation with the topology minimizing objective in the framework of ILP solution based on α parameter value.

a		0.9		0.1		0.01		0.001		0.0001
Primary capacity		18.88		18.88		18.90		19.18	19.58	20.14
Backup capacity		30.34		30.36		30.69		33.92	34.68	38.86
Total capacity		49.23		49.24		49.59		53.10	54.26	59.00
Primary path length		2.36		2.36		2.36		2.39	2.45	2.52
Backup path length		3.74		3.74		3.78		4.17	4.31	4.88
Path length		3.05		3.05		3.07		3.28	3.38	3.70
VPN extension		3.36		3.34		3.08	2.91	2.63		2.45
VPN node coverage		82.04		81.72		77.47	76.31	70.75		70.09
Time consumption	0.1-1.5 min	1.5-2 min		1.5-2 min		1.5-2 min		14 min		47 min

Table 4.8: Simulated Allocation for Protected VPNs Compared to ILP Solution

As it can bee seen in the table, the results of Simulated Allocation requires more capacity and yields longer paths than the ILP solution with $\alpha = 0.001$, while the topology

minimization is still moderate corresponding to approximately $\alpha = 0.01$. The time consumption remains lower than the time consumption of the ILP solver, but it must be noted that the topology minimization is not realized even, only to certain extent.

The same approach has been followed as by the not protected Simulated Allocation, i.e. 4 variants of objectives has been evaluated (see Table 3.10) from capacity minimization to topology minimization. Figures 4.9 and 4.10 depict the results of these 4 objective variants. The capacity reservation is very similar to the results of the ILP solution (see figure 4.1), and the same tendency can be observed, namely the topology minimization increases the backup capacity more than the primary. The capacity reservation is slightly higher than the ILP solution values, they are within 1.5%. The difference is more significant regarding the VPN node coverage, namely Simulated Allocation is able to get only near by 6%.

The average path length results are similar to the capacity reservation results. The values are slightly higher (within 0.15), and the topology minimization affects the backup paths more than the primary paths. The VPN extension is greater (approximately with 0.3) than the result of ILP solution.

The VPN node coverage and the VPN extension is lower by the first test network with the type 2 combined minimization than the topology minimization, which indicates that the network topology can influence the heuristic, however, the difference is within 1.5% and 0.1 respectively.

To achieve better topology metric values I tried to extend the capacity bounds. Table 4.9 shows the changing of these metrics that are the average of all test cases on all the three test networks. The protected VPN configuration by Simulated Allocation can properly approximate the results of the ILP solution, but only with 40% (or more) capacity extension, instead of the not protected case where 25% extension was enough (see Table 3.12).

Shared Protection with Simulated Allocation Shared protection effectively reduces the capacity reserved for backup paths due to the sharing. The test runs have been done on the initial network for the protected cases, i.e. the one with 250% capacity bounds, to be able to compare the results with the dedicated protection. This capacity was enough to route almost all test cases (over 97%).

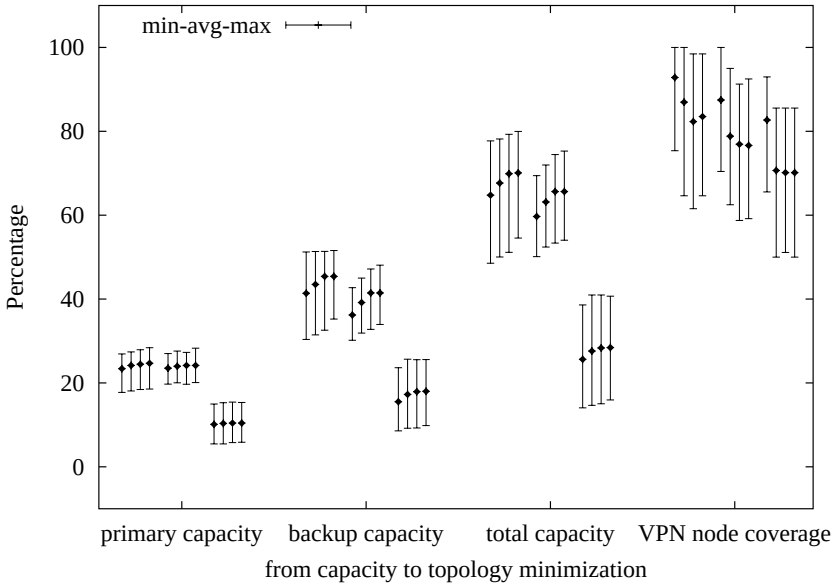


Figure 4.9: Simulated Allocation for Dedicated Protection with Different Objectives – Metrics in Percentage

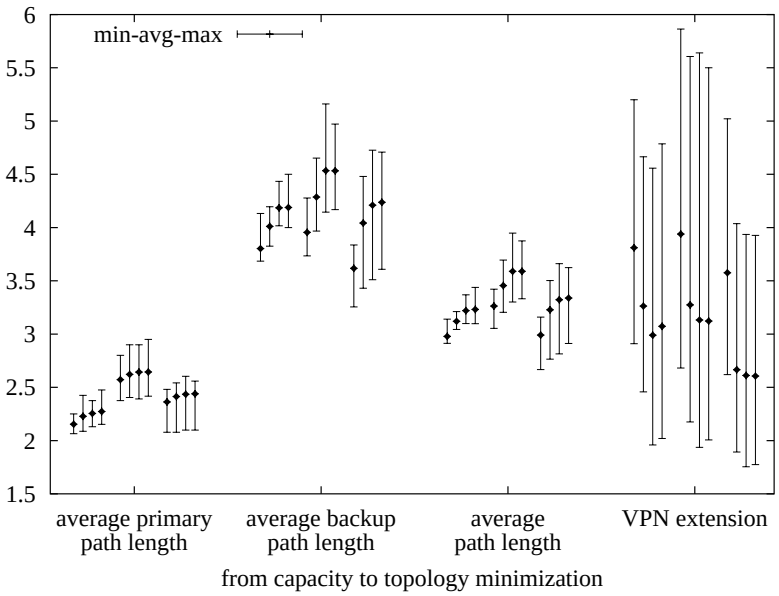


Figure 4.10: Simulated Allocation for Dedicated Protection with Different Objectives – Metrics with Absolute Values

Capacity extension	VPN extension	VPN node coverage (%)
original (250%)	2.93	76.60
+10% (275%)	2.78	73.98
+20% (300%)	2.70	72.46
+30% (325%)	2.66	71.83
+40% (350%)	2.64	71.46
ILP with $\alpha = 0.001$	2.63	70.75

Table 4.9: Simulated Allocation for Protected VPNs – Improving Topology Minimization with Capacity Relaxation

The 4 objective variants were investigated again, from the capacity minimization to the topology minimization. Figures 4.11 and 4.12 show these results. As shared protection affects only the backup paths, the primary capacity reservation is the same as by the dedicated protection. However, the backup capacity reservation is significantly reduced, especially with the capacity minimization objective. Thence, the total reserved capacity is also reduced. The VPN node coverage values are lower than by the dedicated protection, except with the capacity minimization objective. As the capacity reservation for the primary paths, the average primary path length is similar to results of the dedicated protection. However, the backup path lengths are somewhat higher, and they are also varying among the objective types, and this effect is occurring also in the total path length. The VPN extension values are in accordance with the VPN node coverage, i.e. the results are lower than the result of dedicated protection, except with the capacity minimization objective.

Table 4.10 summarizes the numerical values for comparing the dedicated and shared protection methods by the capacity and topology minimizing objectives. The values are the averages on all test cases.

By capacity minimization the reduction in backup capacity reservation is around 18% with shared protection compared to dedicated protection. It is around 9% by the topology minimization. This is also true for the total capacity reservation, because the primary capacity reservation is approximately the same. The backup path lengths are definitely longer by capacity minimization (with 1.59), but only moderately longer in case of topology minimization (with 0.24). The differences in total path lengths are half of them, because again

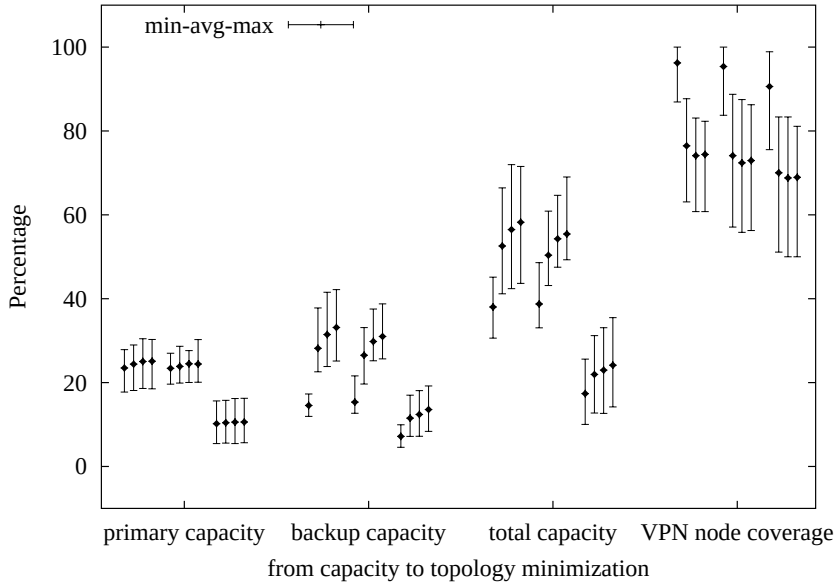


Figure 4.11: Simulated Allocation for Shared Protection with Different Objectives – Metrics in Percentage

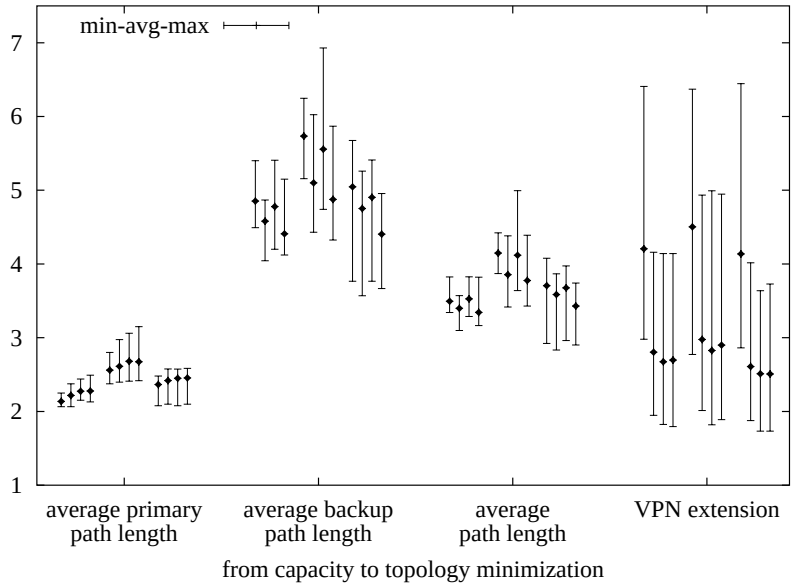


Figure 4.12: Simulated Allocation for Shared Protection with Different Objectives – Metrics with Absolute Values

Metric	Capacity minimization		Topology minimization	
	Dedicated Protection	Shared Protection	Dedicated Protection	Shared Protection
Primary capacity (%)	18.91	19.03	19.65	20.04
Backup capacity (%)	30.80	12.38	34.71	25.90
Total capacity (%)	49.70	31.41	54.36	45.94
Primary path length	2.37	2.35	2.46	2.47
Backup path length	3.79	5.38	4.32	4.56
Total path length	3.08	3.86	3.39	3.52
VPN extension	3.77	4.31	2.93	2.70
VPN node coverage (%)	87.54	94.56	76.60	72.09

Table 4.10: Comparison of Simulated Allocation for Dedicated and Shared Protection of VPNs

the primary path lengths are similar. The VPN extension is lower with capacity minimization by the dedicated protection, however, it is higher with topology minimization and the same is true for the VPN node coverage. Thus, shared protection can significantly reduce the capacity reservation even with topology minimization objectives, where the topology metrics are also better than the results of dedicated protection with SimAll.

Comparing the results of shared protection to the results of ILP solution the VPN related metrics are in between the results of dedicated ILP solution and the dedicated protection with Simulated Allocation (see Table 4.11). Thus, SimAll gives a quite good solution for the shared protection VPN design, and in reasonable time (1–5 minutes).

Design method	Protection type	VPN extension	VPN node coverage (%)
Global ILP, $\alpha = 0.001$	dedicated	2.63	70.75
Simulated Allocation, Topology min.	shared	2.70	72.09
Simulated Allocation, Topology min.	dedicated	2.93	76.60

Table 4.11: Simulated Allocation for Shared Protection – Topology Minimazition Results

Chapter 5

Applications

The optimal resource partitioning problem for virtual networks is present in many areas of telecommunication networks. VPNs can be applied to different network architectures, e.g. to ATM, MPLS or DWDM networks. The output of the optimization gives the logical design of the VPNs.

5.1 Link Partitioning

ATM networks provide quality of service (QoS) including bandwidth guarantees. ATM VPNs are created using (semi-)permanent ATM virtual paths or circuits rented from the public network. These point-to-point links form the virtual backbone for a VPN. VPN service providers offer "managed router" services, where the service provider designs and operates the virtual backbone for each VPN. In this service the virtual backbones could be determined by the result of the VPN design process.

In MPLS networks LSPs can be established with reserved resources, e.g. with the help of RSVP [BZB+97, ABG+01]. Such an LSP is called "guaranteed bandwidth LSP". These guaranteed bandwidth LSPs can be set up in the network according to the result of the optimization by inserting Explicit Route Object into the RSVP PATH message. This ensures that the LSP will be established along the selected path [DR00]. Basically the optimization process realizes the constraint-based routing, i.e. it selects paths that satisfy the capacity and topology objectives.

While VPNs offer security and privacy, optical networks provide high-speed transfer and QoS guarantees for them. VPNs have great importance in case of optical networks, because VPNs can be formed using wavelengths-paths. Therefore introducing Wavelength Division Multiplexing (WDM) into transport networks makes VPN configuration an increasingly important topic. Optical VPN design over wavelength-routing DWDM networks with wavelength routing switches enabling the dynamic establishment of light-paths between VPN endpoints, while no wavelength-conversion capability is assumed is presented in [**C7**]. The results provide the virtual light-path configuration over the optically switched network, which accomplishes the bandwidth requirements and route the demands between the VPN endpoints.

5.2 Node Partitioning

The result of the design process gives the VPN topologies and the paths with the reserved bandwidth inside the VPN topology. The bandwidth reservation values at each node can be summed by the VPNs that have traffic flowing trough that node. The bandwidth utilization ratio by the different VPNs at a given node yields the partitioning of the resources of that node.

The Multiservice Switching Forum [msf] architecture enables the sharing of common physical infrastructure for a multitude of services, e.g. different types of VPNs, such as voice, video or data [**J2**]. Therefore the control logic and the switching or forwarding functionality is divided.

In such multiservice networks the different VPNs over the same hardware platform require the partitioning of the node resources. These partitions are the virtual switches or routers [NSS+04], that see only the “virtualized” resources, i.e. the subset of the physical resources of the hardware. This way strictly independent VPNs are created that do not influence each other. The number of partitions on a particular switch needs to be also minimized, to reduce the overhead of the partitioning. The VPN node coverage topology objective strives for this goal.

This idea is also applied to IP routers [KOBG04], where virtual routers are created with separated routing tables in a similar form. The routing protocol between the virtual routers can be independent of the routing used in the physical backbone. It also enables to add new functions to a virtual router, by implementing proprietary control software [**C3**] even specialized for VPNs [Isa00, IL01]. However, the performance of virtualized routers require thorough evaluation [**C6**, PE00].

The important units of voice VPNs are the media gateways, that connect different voice networks [**J4**]. Media gateways must support a variety of signaling and media type conversion. While several Digital Signal Processors (DSPs) handle the media type conversion, one general purpose processor is enough to handle the signaling. From this arised the concept of decomposition, i.e. to separate the *media gateway* and the *gateway controller*, which also yields scalability. The control/signaling functions are often implemented in software, called “softswitch”, that can easily be upgraded or changed. Moreover, in one softswitch there can be multiple controller softwares running in parallel, controlling several media gateways. Thus, the resources of the softswitch needs to be partitioned and these partitions can be considered as separate VPNs.

Acronyms

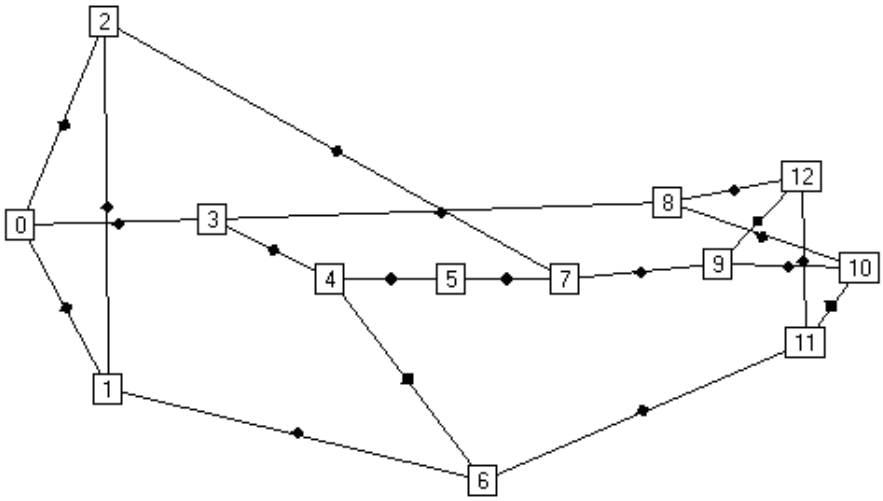
ATM	Asynchronous Transfer Mode
COST	European Co-Operation in the field of Scientific and Technical Research
DSP	Digital Signal Processor
DWDM	Dense Wavelength Division Multiplexing
GHz	GigaHertz
GMPLS	Generalized Multi Protocol Label Switching
GRE	Generic Routing Encapsulation
ILP	Integer Linear Programming
IP	Internet Protocol
IPSec	Secure Internet Protocol
ISO	International Standards Organizatio
ISP	Internet Service Provider
L2TP	Layer 2 Tunneling Protocol
LAN	Local Area Network
LP	Linear Programming
LSP	Label Switched Path
MPLS	Multi Protocol Label Switching
NP	Non Polynomial
NSF	National Science Foundation
OSI	Open System Interconnection
PC	Personal Computer
PPTP	Point to Point Tunneling Protocol

QoS	Quality of Service
RSVP	Resource Reservation Protocol
SimAll	Simulated Allocation
SMT	Steiner Minimum Tree
TDM	Time Division Multiplexing
VPN	Virtual Private Network
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing

Appendix A

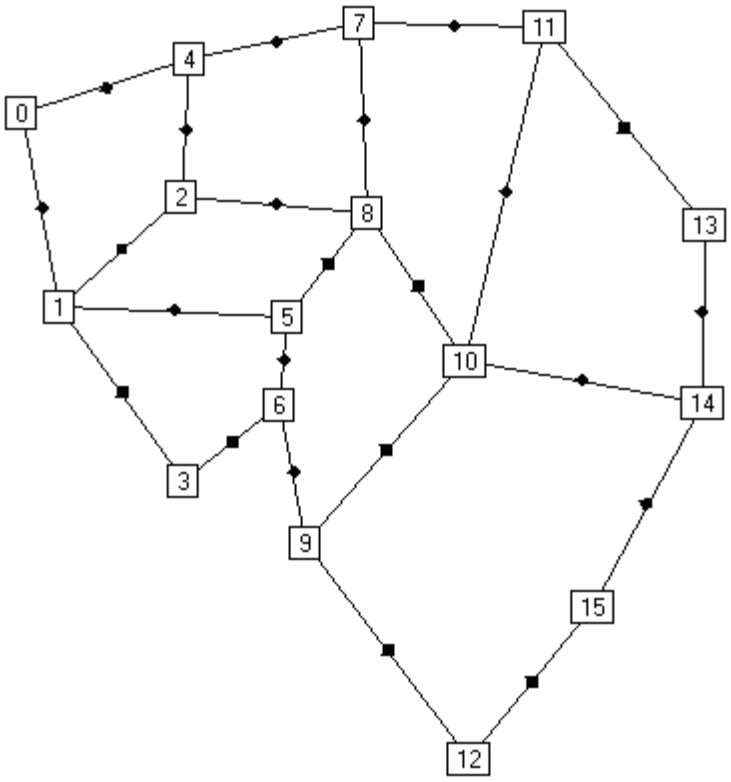
Test Network Topologies

The topologies were drawn with the help of Interactive Graph Drawing [gra].



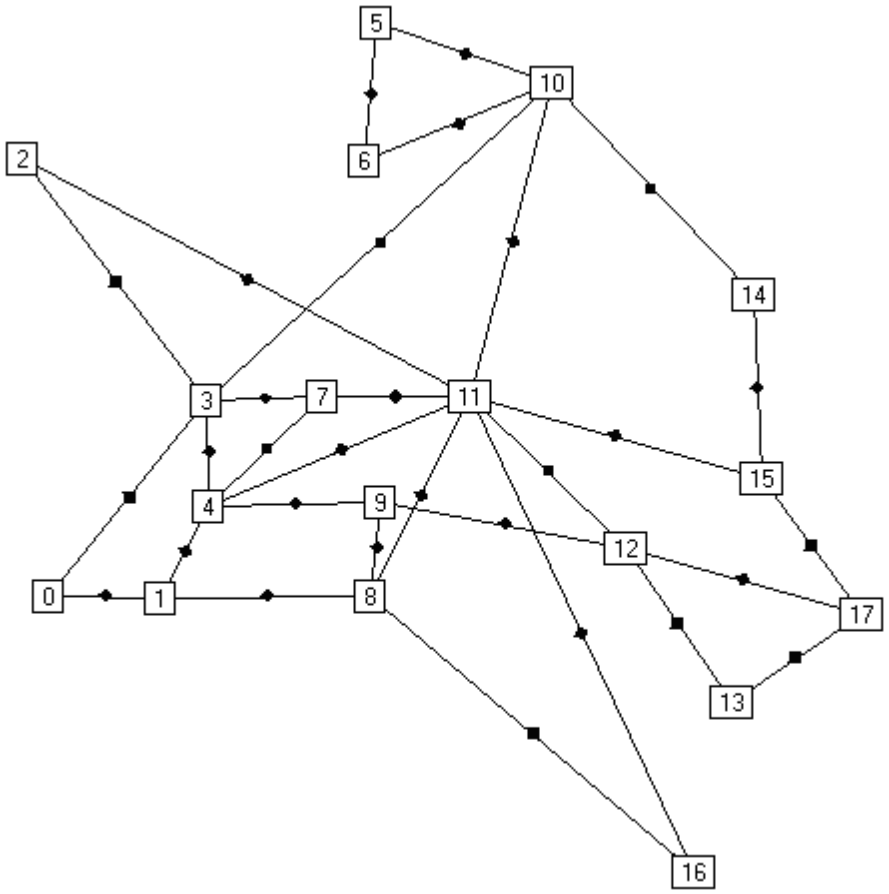
0 Palo Alto	5 Lincoln	10 Princeton
1 San Diego	6 Houston	11 Washington
2 Seattle	7 Urbana	12 Ithaca
3 Salt Lake City	8 Ann Arbor	
4 Boulder	9 Pittsburgh	

Figure A.1: NSFNet Topology



0 London	4 Amsterdam	8 Frankfurt	12 Rome
1 Paris	5 Strasbourg	9 Milan	13 Prague
2 Brussels	6 Zurich	10 Munich	14 Vienna
3 Lyon	7 Hamburg	11 Berlin	15 Zagreb

Figure A.2: COST 266 Core Network Topology



0 Portugal	6 Denmark	12 Austria
1 Spain	7 Netherlands	13 Croatia
2 Ireland	8 Italy	14 Poland
3 UK	9 Switzerland	15 Czech Republic
4 France	10 Sweden	16 Greece
5 Norway	11 Germany	17 Hungary

Figure A.3: Simplified GÉANT Topology

Bibliography

[ABG+01] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, December 2001.

[AQ00] V. Anand and C. Qiao. Dynamic Establishment of Protection Paths in WDM Network, Part I. In *Proceedings of the* 9^{th} *International Conference on Computer Communications*, October 2000.

[AR04] Loa Andersson and Eric C. Rosen. Framework for Layer 2 Virtual Private Networks (L2VPNs). Internet Draft, June 2004.

[BM03] Pietro Belotti and Federico Malucelli. A lagrangian relaxation approach for the design of networks with shared protection. In *Proceedings of the International Network Optimization Conference*, page 72, Evry/Paris, October 2003.

[BZB+97] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP). RFC 2205, September 1997.

[CK00] Reuven Cohen and Gideon Kaempfer. On the Cost of Virtual Private Networks. *IEEE/ACM TRANSACTIONS ON NETWORKING*, 8(6):775–784, December 2000.

[CLL04] I-Wei Chen, Ying-Dar Lin, and Yi-Neng Lin. Tunnel Minimization and Relay for Managing Virtual Private Networks. In *Proceedings of the Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, volume 4, pages 2128–2133, November-December 2004.

[cpl] ILOG CPLEX. http://www.ilog.com/products/cplex/.

[CS05] R. Callon and M. Suzuki. A Framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs). RFC 4110, July 2005.

[DG01] J. Doucette and W. Grover. Comparison of Mesh Protection and Restoration Schemes and the Dependency on Graph Connectivity. In *Proceedings of the* 3^{rd} *International Workshop on Design of Reliable Communication Networks*, 2001.

[Dij59] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[DR00] Bruce S. Davie and Yakov Rekhter. *MPLS: Technology and Applications*. Morgan Kaufmann, 2000.

[FLH+00] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina. Generic Routing Encapsulation (GRE). RFC 2784, March 2000.

[fS94] International Organization for Standardization. Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model, Second edition. International Standard, November 1994.

[gea] GÉANT Website. http://www.geant.net/.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, San Francisco, 1979.

[GLH+00] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. A framework for ip based virtual private networks. RFC 2764, February 2000.

[gra] Interactive graph drawing. http://www.cs.rpi.edu/research/groups/pb/graphdraw/.

[HH04] A. Haque and Pin-Han Ho. Design of Survivable Optical Virtual Private Networks (O-VPNs). In *Proceedings of the* 1^{st} *IEEE International Workshop*

on Provisioning and Transport for Hybrid Networks, San, Jose, CA, USA, Oct 2004.

[HPV+99] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, and G. Zorn. Point-to-Point Tunneling Protocol (PPTP). RFC 2637, July 1999.

[Hu63] T. C. Hu. Multicommodity Network Flows. *Operations Research*, 11:344–360, 1963.

[IKM03] Editors: Robert Inkret, Anton Kuchar, and Branko Mikac. Advanced Infrastructure for Photonic Networks – COST 266 Extended Final Report. Technical report, Faculty of Electrical Engineering and Computing, University of Zagreb, 2003.

[IL01] Rebecca Isaacs and Ian Leslie. Support for resource-assured and dynamic virtual private networks. *IEEE Journal on Selected Areas in Communications*, 19(3), March 2001.

[Isa00] R. Isaacs. Lightweight, Dynamic and Programmable Virtual Private Networks. In *Proceedings of the IEEE Third Conference on Open Architectures and Network Programming*, pages 3–12, March 2000.

[KA98] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. RFC 2401, November 1998.

[KARR90] P. Klein, A. Agrawal, R. Ravi, and S. Rao. Approximation through multicommodity flow. In *Proceedings of the* 31^{st} *IEEE Symp. on Foundations of Computer Science (FOCS'90)*, pages 726–739. IEEE Press, 1990.

[KEKAP02] E. Karasan, O. Ekin-Karasan, N. Akar, and M.C. Pinar. Mesh topology design in overlay virtual private networks. *Electronics Letters*, 38(16):939–941, 2002.

[KL03] Murali Kodialam and T. V. Lakshman. Dynamic Routing of Restorable Bandwidth-Guaranteed Tunnels Using Aggregated Network Resource Usage Information. *IEEE/ACM TRANSACTIONS ON NETWORKING*, 11(3):399–410, June 2003.

[KL04] P. Knight and C. Lewis. Layer 2 and 3 Virtual Private Networks: Taxonomy, Technology, and Standardization Efforts. *IEEE Communications Magazine*, 42(6):124–131, June 2004.

[KOBG04] Paul Knight, Hamid Ould-Brahim, and Bryan Gleeson. Network based IP VPN Architecture using Virtual Routers. Internet Draft, April 2004.

[LGN+01] L. K. Lim, J. Gao, T. S. E. Ng, P. Chandra, P. Steenkiste, and H. Zhang. Customizable Virtual Private Network Service with QoS. *Computer Networks*, 36(2-3):137–151, July 2001.

[LMP+91] T. Leighton, F. Makedon, S. Plotkin, É. Tardos C. Stein, and S.Tragoudas. Fast approximation algorithms for multicommodity flow problems. In *Proceedings of the* 25^{th} *Annual ACM Symposium on the Theory of Computing*, pages 101–111, 1991.

[LMSL90] C.-L. Li, S.T. McCormick, and D. Simchi-Levi. The complexity of finding two disjoint paths with min-max objective function. *Discrete Applied Mathematics*, 26(1):105–115, 1990.

[LR88] F. T. Leighton and S. Rao. An Approximate Max-Flow Min-Cut Theorem for Uniform Multicommodity Flow Problems with Application to Approximation Algorithms. In *Proceedings of the* 29^{th} *IEEE Symposium on Foundations of Computer Science*, pages 422–431, 1988.

[Man04] Editor: E. Mannie. Generalized Multi-Protocol Label Switching (GMPLS) Architecture. RFC 3945, October 2004.

[Min89] M. Minoux. Network Synthesis and Optimum Network Design Problems: Models, Solution methods and Applications. *NETWORK*, 19:313–360, 1989.

[msf] MultiService Forum. http://www.msforum.org/.

[nsf] NSFNET – The National Science Foundation Network. http://moat.nlanr.net/INFRA/NSFNET.html.

[NSS+04] Ananth Nagarajan, Junichi Sumimoto, Muneyoshi Suzuki, Paul Knight, and Benson Schliesser. Applicability Statement for Virtual Router-based Layer 3 PPVPN Approaches. Internet Draft, February 2004.

[oNUL] Optimization Technology Center of Northwestern University and Argonne National Laboratory. Linear programming frequently asked questions. http://www-unix.mcs.anl.gov/otc/Guide/faq/linear-programming-faq.html.

[PE00] C.J.C. Pena and J. Evans. Performance Evaluation of Software Virtual Private Networks (VPN. In *Proceedings of the* 25^{th} *Annual IEEE Conference on Local Computer Networks*, November 2000.

[PSG+00] M. Pióro, T. J. K. Stidsen, A. J. Glenstrup, C. Fenger, and H. Christiansen. Design problems in robust optical networks. In *Proceedings of the Networks 2000*, Toronto, Canada, 10–15 September 2000.

[QLLY02] Yang Qin, Bo Li, Wee Liang Lim, and Soon Hoe Yeo. A design for online virtual private networks (VPN) over optical WDM networks. In *Proceedings of the IEEE Global Telecommunications Conference, GLOBECOM*, volume 3, pages 2782–2786, November 2002.

[Rad95] T. Radzik. Fast deterministic approximation for the multicommodity flow problem. In *Proceedings of the* 6^{th} *ACM-SIAM symposium on Discrete algorithms*, pages 486–492, San Francisco, 1995.

[RR99] E. Rosen and Y. Rekhter. BGP/MPLS VPNs. RFC 2547, March 1999.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, January 2001.

[SOL] CUSTOM BUILT SOLUTIONS. http://www.cbscompany.com/services/vpn_main.htm.

[ST84] J. W. Suurballe and R. E. Tarjan. A quick method for finding shortest pairs of disjoint paths. *Networks 14*, pages 325–336, 1984.

[ST02] A. Srikitja and D. Tipper. Topological design of multiple VPNs over MPLS network. In *Proceedings of the IEEE Global Telecommunications Conference*, volume 3, pages 2195–2199, November 2002.

[TP05] Tomonori Takeda and Dimitri Papadimitriou. Layer 1 Virtual Private Networks: Driving Forces and Realization by GMPLS. *IEEE Communications Magazine*, pages 60–67, July 2005.

[TVR+99] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter. Layer Two Tunneling Protocol "L2TP". RFC 2661, August 1999.

[XCX+04] D. Xu, Y. Chen, Y. Xiong, C. Qiao, and X. He. On finding disjoint paths in single and dual link cost networks. In *Proceedings of the IEEE Infocom*, Hong Kong, March 2004.

[ZOM03] Hui Zang, Canhui (Sam) Ou, and Biswanath Mukherjee. Path-Protection Routing and Wavelength Assignment (RWA) in WDM Mesh Networks Under Duct-Layer Constraints. *IEEE/ACM Transactions on Networking*, 11(2):248–258, April 2003.

Publications

[J] **JOURNALS**

[**J1**] P. Hegyi, M. Maliosz, Á. Ladányi, T. Cinkler. Virtual Private/Overlay Network Design with Traffic Concentration and Shared Protection. *Journal of Network and Systems Management, Special Issue on Designing and Managing Optical Networks and Services for High Reliability*, Vol. 13, No. 1, March 2005. LR

[**J2**] T. Lundberg, N. Björkman, M. Maliosz and Y. Jiang. Realizing component based networking. *Journal of Communications and Networks, Special Issue on Programmable Switches and Routers*, pp. 56–63, March 2001. LR

[**J3**] M. Maliosz, K. Farkas. ATM hálózatra épülõ interaktív televízió szolgáltatás platformfüggetlen kezelõi felülettel. *Magyar Távközlés folyóirat*, pp. 21–24, 1998/II. Szám L

[**J4**] M. Maliosz, M. Nagy, M. Sebõk, Sz. Daróczy. Dekompozíció - Az átjárók új generációja. *Híradástechnika folyóirat* pp. 27–30, 2001. szeptember L

[C] **CONFERENCES**

[**C1**] M. Maliosz, K. Farkas, I. Cselényi. Agent Based Interactive Television Service For An Experimental Multimedia System. In *Proceedings of the International Conference on Communication Systems*, Singapore, 1998 November L

[**C2**] D. Ahlard, J. Bergkvist, I. Cselényi, T. Engborg, K. Németh, G. Fehér, M. Maliosz. Boomerang - A Simple Protocol for Resource Reservation in IP Networks. In *Proceedings of the IEEE Workshop on QoS Support for Real-Time Internet Applications*, pp. 93–99, Vancouver, Canada, 1999 June L

[**C3**] M. Maliosz, N. Végh. Resource Reservation with Boomerang in Connection Oriented Networks. In *Proceedings of the European Conference on Universal Multiservice Networks*, Colmar, France, 2000 October

[**C4**] M. Maliosz, T. Cinkler. Optimizing Configuration of Virtual Private Networks. In *Proceedings of the 2001 Polish-Czech-Hungarian Workshop*, pp. 241–247, Budapest, Hungary, 2001 September L

[**C5**] T. Cinkler, M. Maliosz. Configuration of Protected Virtual Private Networks. In *Proceedings of the Third International Workshop on Design of Reliable Communication Networks*, Budapest, Hungary, 2001 October L

[**C6**] M. Maliosz, T. Lundberg, N. Végh. Performance Evaluation of an Open Control Testbed. In *Proceedings of the International Conference on Software, Telecommunications and Computer Networks*, pp. 593–601, Split, Croatia, 2001 October L

[**C7**] M. Maliosz, T. Cinkler. Methods for Optical VPN Design over Multifiber Wavelength Routing Networks. In *Proceedings of the* 7^{th} *IFIP Working Conference on Optical Network Design & Modelling*, Budapest, Hungary, 2003 February L

[**C8**] P. Hegyi, M. Maliosz, Á. Ladányi, T. Cinkler. Shared Protection of Virtual Private Networks with Heuristic Methods. In *Proceedings of the 2003 Polish-Hungarian-Czech Workshop on Circuit Theory, Signal Processing, and Applications*, Prague, Czech Republic, 2003 September

[**C9**] P. Hegyi, M. Maliosz, Á. Ladányi, T. Cinkler. Shared Protection of Virtual Private Networks with Traffic Concentration. In *Proceedings of the* 4^{th} *International Workshop on Design of Reliable Communication Networks*, Alberta, Canada, 2003 October, L

[**C10**] J. Tapolcai, P. Fodor, G. Rétvári, M. Maliosz, T. Cinkler. Class-based Minimum Interference Routing for Traffic Engineering in Optical Networks. In *Proceedings of the First Conference on Next Generation Internet Networks Traffic Engineering*, Rome, Italy, 2005 April 18-20, L

[W] **WORKSHOPS AND OTHERS**

[**W1**] P. Hegyi, M. Maliosz, Á. Ladányi, T. Cinkler. Heuristic Algorithms for Shared Protection of Virtual Private Networks. 9^{th} *EUNICE Open European Summer School and IFIP WG6.3 Workshop on Next Generation Networks*, Balatonfüred, Hungary, September 2003.