Journal of Heuristics, 6, 107–130 (2000)

A Zoom-In Approach to Design SDH Mesh Restorable Networks

MARIO PICKAVET* AND PIET DEMEESTER
University of Gent-IMEC, Department of Information Technology, Sint-Pietersnieuwstraat 41, 9000 Gent (Belgium)
email: mario.pickavet@intec.rug.ac.be

Abstract

Mesh restorable networks based on SONET (Synchronous Optical Network, standard optical transmission technology widely accepted and implemented in North America) or SDH (Synchronous Digital Hierarchy, the European standard currently adopted by the major European telecom operators) are an economically attractive solution in areas where high demand and high connectivity are involved (Wu, 1995). In these networks, the reconfiguration capability of the digital cross connect systems (DCS) allows to reroute the demand affected by network failures. The degree of sharing of spare capacity in networks based on this architecture is high.

This paper presents a heuristic algorithm for solving the near-optimal design of SDH mesh-type link restorable networks, i.e. determining the network topology and assigning the capacity to transport the demand in normal situations and to allow full link restorability in case of single link failures. The algorithm is based on a Zoom-In technique, a novel approach which forms a compromise between sequential and integrated techniques. The different building blocks of the algorithm are tested extensively and compared with other results mentioned in literature. Comparison of the simulation results for the overall design problem with other solution techniques indicates that the Zoom-In method is a quite promising approach, able to combine the accuracy of integrated approaches with the calculation speed of sequential approaches.

Key Words: capacity design, Genetic Algorithm, link restoration, topology design, Zoom-In technique

1. Introduction

The problem studied in this paper is the design of SDH mesh restorable networks.[1] Given is a set of node locations and a VC-4 node-to-node demand (1 VC-4 corresponds to a bit rate of 155 Mbit/s). Asked is to determine between which nodes a link must be installed (topology design), how to route the demand (routing design) and which capacity should be assigned to each link (capacity design). The installed capacity must be sufficient to carry the demand, not only when all network components are functioning properly, but also in case of a single link failure. The mechanism used to reroute the demand affected by a link failure is link restoration: the affected connections are rerouted between the endpoints of the failing link (see (Wu, 1992; Sato, 1996)).

Installing a link with a certain VC-4 capacity between two nodes comprises the installation of a cable between those nodes and a sufficient number of fiber systems. A fiber system

*Research Assistant of the Fund for Scientific Research—Flanders (F.W.O.-V, Belgium).

consists of a line system and eventually regenerators at regular distances along the cable to upgrade the attenuated and distorted signal. Accordingly, three cost components were considered (including equipment cost as well as installation cost): the cable cost, the line system cost and the regenerator cost. The cable cost is roughly proportional to the distance between the nodes. With respect to the cost of the line systems and regenerators, we considered STM-1 (155 Mbit/s), STM-4 (622 Mbit/s) and STM-16 (2.5 Gbit/s) fiber systems. Economies of scale play a role in the selection of line systems. For instance, an STM-4 offers a capacity which is four times as high as that of an STM-1 at less than double the cost (a similar relationship holds between STM-16 and STM-4). As a result, when the VC-4 demand offered to the network is relatively high, one may expect a predominant use of STM-16 line systems, STM-4's and STM-1's only being installed where STM-16's would be severely underused. The cost of the regenerators depends on the type of used fiber system, but also shows a stepwise dependence on the length of the cable.

To measure the performance of a potential solution method for the complex network design problem described above, several aspects must be taken into account. The most important element is, of course, the quality of the found solutions (i.e., a low cost). However, also other features, that are related to the available computing resources such as the calculation time and the memory requirements, turn out to be indispensable to make a correct comparison between different algorithms. Despite the off-line nature of a network design process, the calculation speed is important because it restricts the size of problem instances that can be solved within a "reasonable" time. Also the memory requirements of an algorithm might pose a restriction on the size of problems that can be solved in practice. Depending on the particular algorithm and the available resources, either the calculation time or the memory usage can be the most limiting factor.

A number of related network design problems has been considered in literature. On one hand, the spare capacity assignment problem in mesh restorable networks has been treated by many authors, see for instance Grover, Bilodeau, and Venables (1991), Herzberg (1993), Herzberg, Bye, and Utano (1995), and Poppe and Demeester (1998). Starting from a given network topology and a given routing in the failureless state, these papers describe algorithms to allocate spare capacity for different failure types and restoration mechanisms. On the other hand, our problem is also related to the design of multi-commodity survivable networks, studied for instance in Minoux (1981), Gavish et al. (1989), and Stoer and Dahl (1994). These papers assume that there is no relationship between the routing of the demand in the failureless state and the routing of the demand in case of a failure. The formulations they studied impose the feasibility of a multi-commodity flow in each of the network states (the failureless state and each of the failure states), which is valid under the assumption that there is no restriction on the way the network may be reconfigured after a failure has occurred. This rerouting strategy contrasts with the link restoration situation, where a clear relationship is noticed between the routing in the failureless state and the routing in each of the failure states: the traffic which is not affected by the (link) failure cannot be rerouted and the affected traffic can only be rerouted between the endpoints of the failing link.

This paper presents a novel algorithm that combines the topology design and capacity assignment in case of a strict relationship between the routing in the failureless state and the failure states, based on the link restoration paradigm. In Poppe and Demeester (1997) a similar problem is described and an Integer Linear Programming based algorithm

is presented. Further on, we will compare the results of the two approaches whenever possible.

The paper is structured as follows. The two main building blocks of the algorithm are a type of Genetic Algorithm which solves a more abstract capacitated survivable network design problem and an algorithm to allocate the spare capacity, described in Sections 2 and 3, respectively. The overall algorithm (Section 4) is based on a Zoom-In strategy, a technique that forms a compromise between a sequential approach and an integrated approach. In Section 5 some numerical examples are described and attention is paid to the relative importance of the different phases of the algorithm. The major building blocks of the algorithm are tested in Section 6 and the main assumptions on which the Zoom-In algorithm is based are highlighted. The quality of the overall solution is compared with the results of other techniques. Section 7 concludes the paper.

2. Abstract Capacitated Survivable Network Design (ACSND)

One of the modules used in the overall algorithm (described in Section 4) is an algorithm to solve a more abstract network design problem ACSND. The original problem stated in Section 1 is therefore slightly adapted to speed up the algorithm.

Firstly, a linearized cost model is used. This leads to a representation of the cost of a network as a sum of link costs that are bilinear functions of the length and the capacity of the link:

$$c(N) = \sum_{e \in E(N)} [c_1 \cdot l(e) + c_2 \cdot u(e) + c_3 \cdot l(e) \cdot u(e)]$$

where N is a network, $c(N)$ is the cost of the network N, $E(N)$ is the set of links of the network N, $l(e)$ is the length of the link e, $u(e)$ is the capacity provided on link e, c_1, c_2 and c_3 represent cable, line system and regenerator cost, respectively.

Secondly, a more abstract routing strategy is used, which does only indirectly incorporate the spare capacity needed for link restoration. The link capacities are calculated by rescaling the demand matrix (e.g. multiplying all node-to-node demands by a factor $\eta = 1.2$) and dividing the traffic along the two shortest link-disjoint paths. The philosophy behind this bi-routing strategy is that in case of a single link failure at least a fraction $\eta/2$ will not be affected, while the remaining fraction $1 - \eta/2$ can be (partly or entirely) restored using some spare capacity available in the network if $\eta > 1$.

These assumptions lead us to a straightforward optimal capacity assignment procedure for a fixed topology:[2] using a minimum cost flow algorithm (Ahuja, 1993) each demand d is routed through the network with link costs $c_2 + c_3 \cdot l(e)$ (these are the only cost components which depend on the capacity assignment and hence on the routing of the traffic) and link capacities $d/2$ (to ensure bi-routing along link-disjoint paths). This allows a fast evaluation of the total network cost $c(N)$ for a certain topology, so that we can concentrate on the choice of a good topology.

To solve the ACSND problem, a Genetic Algorithm enhanced with some deterministic optimization routines is used. We refer to Pickavet et al. (1997) for a detailed description of the algorithm. The main building blocks of the ACSND algorithm are shown in figure 1.

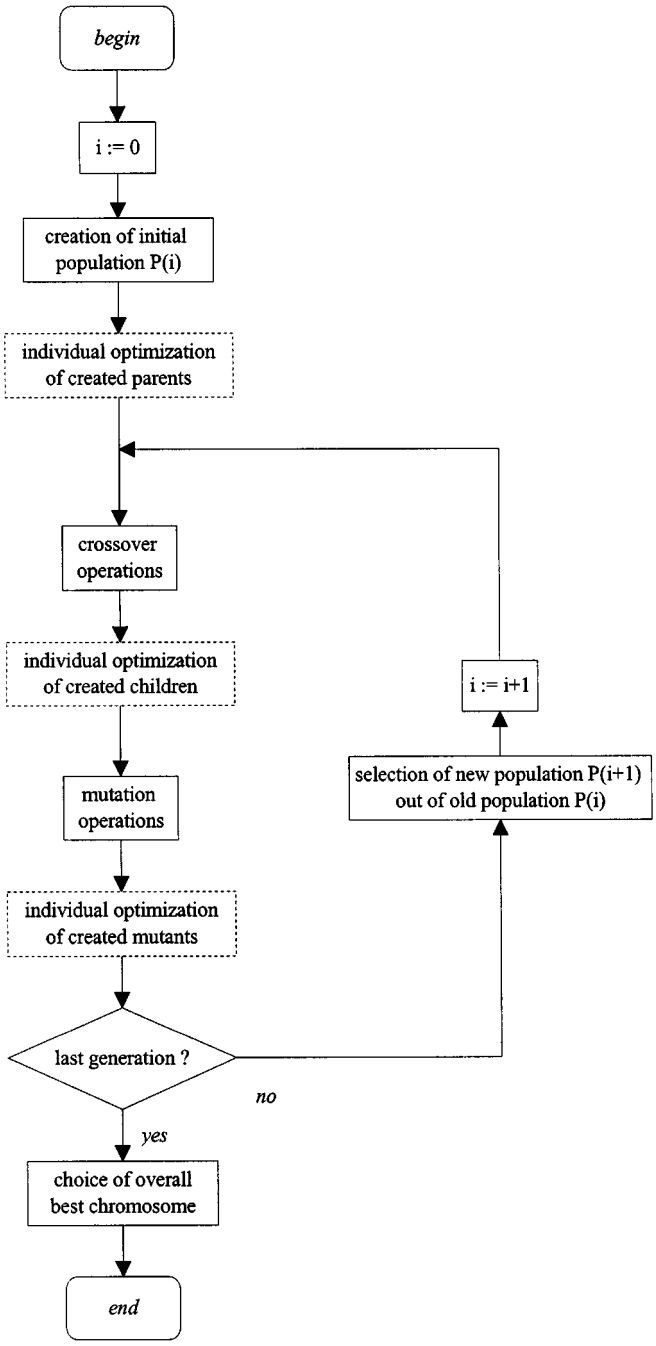

Figure 1. Abstract capacitated survivable network design (ACSND) algorithm.

A Genetic Algorithm (Holland, 1975) is a general optimization technique which mimics the genetic evolution of species. The main difference with other A. I. approaches, as for instance Simulated Annealing or Tabu Search, is that a Genetic Algorithm deals with a 'population' of solutions (a solution is called a 'chromosome') rather than with a single solution. In case of the ACSND problem, a chromosome is in fact a network topology. The objective is to create topologies with a minimal overall network cost $c(N)$ (and with at least link-connectivity 2).

The network topologies in the initial population are created by randomly selecting some links. Some chromosomes are added to this population by applying some crossover and mutation operations. A crossover operation combines two existing chromosomes to create a new (and hopefully better) one, whereas a mutation operation creates a new chromosome by modifying an existing one. Out of this extended population, consisting of so-called parents, children and mutants, a new population is selected according to the 'survival of the fittest' principle: high quality chromosomes (i.e., topologies with at least link-connectivity 2 and a low network cost $c(N)$) have a higher probability of being chosen for the next generation. This sequence of crossover operations, mutation operations and selection of a new generation is repeated until no better solutions are found during a number of generations. Out of all the generations, the best solution is retained as the final solution of the ACSND problem.

It is important to recognize the building blocks (Holland, 1975) of a high quality solution, in order to define a crossover operation that combines the good characteristics of the parents and removes the bad characteristics. We finally opted for a sort of cut-and-paste operation on the network topologies: we replace a 'bad' part[3] of one topology by the corresponding 'good' part of another topology. This is illustrated on a 8-node example problem in figure 2. The 'weakness' of the basic parent network is the single link between nodes v_4 and v_5 leading to a low reliability: if that link fails, the network will be divided in two not-connected parts. By replacing a part of the network around that weakness (all the links that are incident to the nodes v_4 and/or v_5, dashed lines in basic parent network) by the corresponding part of another network (all the links that are incident to the nodes v_4 and/or v_5, full lines in auxiliary parent network), we create a child network which is hopefully more reliable than both of its parents. To decide which crossovers will take place in a population, we first check which parent chromosomes are complementary, i.e., the bad parts of the two topologies should be non-overlapping. From these candidate pairs of complementary parents, a fixed number of pairs is randomly selected, for each pair a basic parent network is randomly assigned and one child chromosome is created per pair.

When defining a suitable mutation operation, it is important to introduce some drastic adaptations to the parent network that allow the Genetic Algorithm to escape from any local minima. The quality of the mutant network is hereby only of secondary importance. We opted for a type of rotation operation. First, the centre of gravity of all nodes is calculated and the nodes of the network are ordered (in a circular way) according to their polar angle from the centre of gravity. The links in the mutant network are then obtained by rotating the node numbering over one position and redrawing the parent network links between the same node numbers in the mutant network. This is illustrated in figure 3 on a 8-node problem.

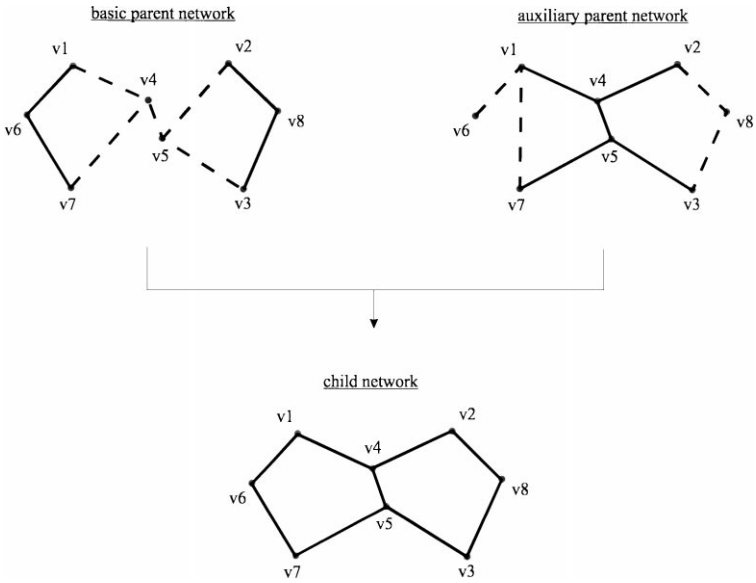

Figure 2. Example of crossover operation.

Figure 3. Example of mutation operation.

The parent networks that will be used for a mutation operation are randomly chosen. Usually the mutant chromosomes in a population form only a small fraction of the total population.

The Genetic Algorithm described above enables us to explore the ACSND search space in a thorough and efficient way, but lacks some strong directives towards link-connectivity 2 of the created topologies. For instance, the link-connectivity of children networks created with the crossover operation is often lower than 2. Therefore, in contrast with the standard Genetic Algorithm paradigm, some deterministic routines were introduced to optimize the chromosomes individually (see figure 1). The most important optimization routines are:

1. On one hand, to ensure a link-connectivity of at least 2, each node of the network should have at least degree 2. Hence, links will be added to interconnect nearby nodes with

Table 1. Typical values for the Genetic Algorithm parameters.

Number of parents per population	20–40
Number of children per population	20–40
Number of mutants per population	5–10
Number of generations	5–20

a low degree. On the other hand very high node degrees usually lead to high network costs, so (long) links between two nodes with a high node degree may be removed.

2. Topologies containing traversing links—i.e. links that are crossing each other—usually lead to a higher network cost than topologies without traversing links, without providing extra connectivity. Hence, two traversing links are omitted and replaced by non-traversing links (if these links are not present yet).
3. If the line system costs (c_2) are high when compared to the cable cost (c_1), introducing additional links in a network with link-connectivity 2 may decrease the overall network cost (due to shorter routing paths).
4. After the calculation of the link-connectivity matrix of the topology, links may be added between nodes with a mutual connectivity below 2. On the other hand, long links that interconnect nodes with a very high mutual connectivity may be removed to reduce the network cost. If needed, this process of evaluating the connectivity of the network and adding/removing some links can be repeated.

The routines 1, 2 and 3 are rather rough heuristics to correct some major shortcomings of a chromosome quickly, while routine 4 is a refined method using more CPU time. The influence of these 4 routines on the quality of the final solution was tested extensively and turned out to be quite significant. The importance of including problem specific methods in a Genetic Algorithm to increase the search efficiency has also been observed for a wide range of other optimization problems (Michalewicz, 1997).

Some typical values for the most important Genetic Algorithm parameters are shown in table 1. In general, the algorithm turns out to be very robust and not very sensitive to the exact parameter values, so fine tuning of the values is not very useful.

3. Spare Capacity Allocation (SCA)

The SCA algorithm starts from a fixed topology and a fixed working capacity assignment. The objective is to allocate enough spare capacity to allow full link restorability of single link failures at minimum cost (using one modularity m, e.g. $m = 16$ if we use STM-16 fiber systems). Moreover, to be suited for use in the Zoom-In approach (see Section 4), the algorithm should be able to provide rather good solutions quickly and very good solutions during a long run.

The algorithm consists of three phases: a minimal spare capacity allocation procedure, a rough spare capacity addition and a tightening phase (see figure 4).

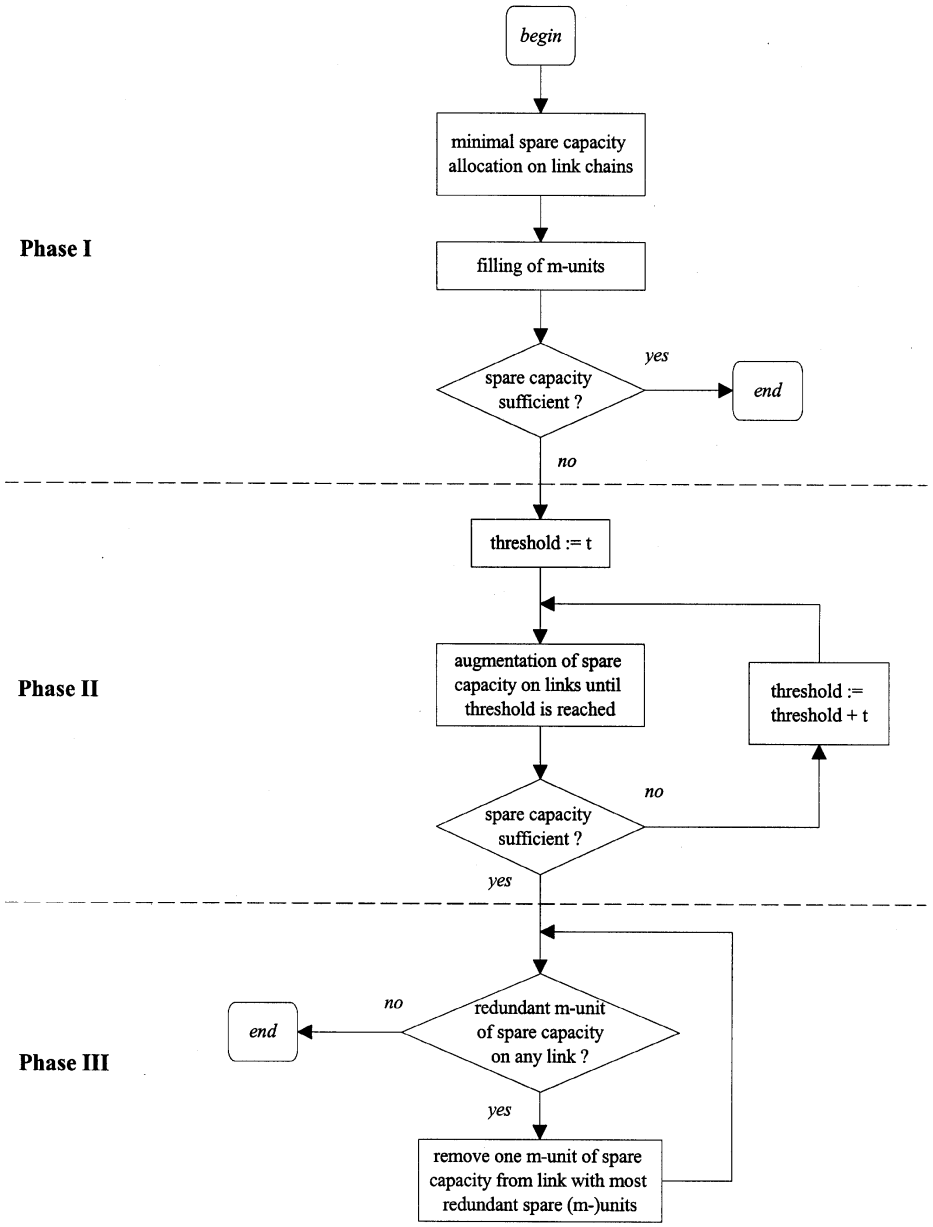

Figure 4. Spare capacity allocation (SCA) algorithm.

3.1. *Phase I: minimal spare capacity allocation*

Starting from the working capacity assignment on each link, a simple and extremely fast method can be used to assign some initial spare capacity on the links. Indeed, if several links $l_1, l_2, \ldots, l_i$ form a chain,[4] the spare capacity $s(l_x)$, $1 \leq x \leq i$, on the link l_x must be at least as big as the working capacity $w(l_y)$ on any other link l_y of the chain (due to the principle of link restoration):

$$s(l_x) \geq \max_{1 \leq y \leq i, y \neq x} w(l_y) \quad 1 \leq x \leq i$$

Furthermore the spare capacity on each link l_x can be augmented without any additional cost as long as no additional line system is needed:

$$s(l_x) := m \left\lceil \frac{s(l_x) + w(l_x)}{m} \right\rceil - w(l_x)$$

After these spare capacity additions, we check whether all single link failures are fully link restorable (using a maximum flow algorithm (Goldberg, 1988)). If so, the found spare capacity allocation is optimal. If not, some spare capacity will be added in phase II.

3.2. *Phase II: spare capacity addition*

If the spare capacity allocation found in phase I turns out to be insufficient, we define an initial threshold t on the spare capacity $s(l_x)$ on each link l_x. If $s(l_x) \leq t$, we add one or more m-units[5] of spare capacity on each link until the threshold is reached:

$$s(l_x) := s(l_x) + m \cdot \max\left(\left\lceil \frac{t - s(l_x)}{m} \right\rceil, 0 \right)$$

If not all single link failures are fully link restorable, we define new thresholds $2t, 3t, \ldots$ until full restorability is reached. Usually the initial threshold t is chosen rather large (e.g. 20% of the average working capacity on a link), resulting in a second phase which is quite fast (only a few threshold adaptations are needed) but inaccurate (inefficient usage of spare capacity).

3.3. *Phase III: spare capacity tightening*

This last and most time-consuming phase starts from the solution found in the second phase and tries to remove some m-units of spare capacity from the links. We repeatedly search for the link (or one of the links) with most redundant units of spare capacity, by changing the rerouting patterns in such a way that the considered link is avoided as much as possible, while honoring the spare capacity restrictions on the other links. We then remove one m-unit of spare capacity from the link with most redundant units. This procedure is repeated until no redundant m-units of spare capacity are left. The philosophy behind removing only one m-unit on a link at a time instead of removing as much m-units as possible is based on the intuitive notion that, by acting in a less greedy fashion, the final result will generally

show a more uniform distribution of the spare resources throughout the network, leading to a more efficient sharing of spare resources.

To speed up the tightening phase, we allow a rough, approximate search for the link with most redundant spare units. For instance, in order to decide from which link an m-unit of spare capacity should be removed, we can compare the links with respect to the number of spare m-units or $m/2$-units on a link that are completely redundant (granularity m or $m/2$ instead of 1). This rough SCA algorithm is much faster and does usually not effect the solution quality significantly.[6]

Remark: The extension of the algorithm towards multiple facility types is straightforward. For instance, let us assume that the three facility types STM-16, STM-4 and STM-1 are considered and that the cost of an STM-16 is twice the cost of an STM-4 and four times the cost of an STM-1 (economy of scale principle). Then, since the STM-16 fiber systems are the most cost-effective option (lowest cost per capacity unit), we first execute the three phases of the SCA algorithm for $m = 16$, resulting in a capacity allocation with only STM-16 line systems. Then the tightening phase III is executed again, to replace some STM-16 systems by STM-4 and/or STM-1 systems, by adapting the rerouting pattern if necessary. Since the removal of an STM-16 by one STM-1 provides now the highest cost saving per removed capacity unit, we start by repeatedly searching for a link with 15 redundant units of capacity and replacing one of the STM-16s on this link by an STM-1, leading to a cost saving corresponding to 0.75 STM-16s. If no links with 15 redundant units are left, we repeatedly search for links with 12 redundant units (replacing an STM-16 by an STM-4, cost saving 0.5 STM-16s) and finally we check the links with redundancy 11 (replacing an STM-16 by an STM-4 and an STM-1, cost saving 0.25 STM-16s). It must be stressed that this multi-facility extension of the SCA algorithm is not a simple link-by-link optimization of the single-facility STM-16 solution, since an adaptation of the rerouting pattern is possible at each facility replacement.

4. Zoom-In algorithm

A sequential three-phase approach to handle the SDH mesh restorable network design problem (defined in Section 1) seems natural. One could start by designing a suitable link topology, connecting all the nodes while satisfying certain connectivity constraints to ensure some reliability (see for instance Monma and Shallcross (1989)). Once the topology is fixed, one could design the routing pattern and assign sufficient working capacity on each link. In the third phase, one starts from a given topology and working capacity assignment to calculate the needed spare capacity on the links to ensure full link restorability of single link failures (see for instance Grover, Bilodeau, and Venables (1991)). The advantage of such a sequential approach is its calculation speed: by dividing the overall design problem in three subproblems that are solved independently, the final solution is found relatively quickly (or, stated otherwise, we can design bigger SDH networks within a reasonable calculation time). However the interdependence of the three subproblems is neglected, e.g. the topology choice has an influence on the capacity costs and the routing of the working traffic has a certain impact on the spare capacity cost. As a consequence, the separation of the three subproblems might lead to low quality solutions (i.e., high costs).

To overcome this disadvantage, one could tackle the design problem as a whole: the relations between the three subproblems described above are fully taken into account. Such an integrated approach makes it the possible to create high quality solutions. However, the required resources, calculation time and memory requirements, are usually much higher than for the sequential approach. As a consequence, only smaller problem instances can be solved in practice. To design bigger networks simplifying assumptions or premature termination of the algorithm must be applied to reduce the calculation time and/or memory requirements. These measures however affect the solution quality.

The problems encountered with these two fundamentally different approaches give rise to a new approach which forms a compromise between the sequential and the integrated approach: a Zoom-In strategy. The 'zooming-in' philosophy is applied with respect to both the accuracy of the solution and the search effort. At the first stage(s) of the algorithm a rough solution is constructed, based on simplifying assumptions. As the algorithm proceeds, more accurate models are used and the search effort is increased. The final stage of the algorithm takes all problem details into account and performs a thorough and detailed search.

The Zoom-In strategy is also reflected in the evolution of the optimization techniques that are used in the algorithm. Global construction heuristics are used in the beginning, eventually combined with some random elements to get a more global view of the solution space. Afterwards, the emphasis shifts towards local improvement techniques where the nature of searching becomes more deterministic.

The Zoom-In approach tries to combine the advantages of both the sequential approach—its calculation speed—and the integrated approach—its high quality solutions. To achieve this, a careful choice has to be made of the problem details to be considered in each phase of the algorithm. This decision is typically based on a trade-off: considering an additional aspect of the problem improves the solution quality, but also leads to an augmentation of the required time and memory.

An additional advantage of the Zoom-In strategy is its flexibility. Depending on the time or memory available, some problem details can be ignored during one or more phases. On one hand, for small problem instances, we can incorporate a large number of problem details from the beginning of the algorithm, leading to high quality solutions. On the other hand, the Zoom-In approach is also useful for much bigger problem instances, since we can ignore some problem details in the first phases of the algorithm to limit the calculation time without losing too much on solution quality.

To apply the Zoom-In strategy to the SDH network design problem, the interactions between topology design, working capacity assignment and spare capacity assignment must be studied carefully. We finally opted for a four-phase algorithm (see table 2).

Table 2. Zoom-In approach applied to SDH network design problem.

	Function	Cost model	Capacity assignment
I	Creation of basic topology	Simplified	Approximation
II	Local optimization of topology	Simplified	Approximation
III	Local optimization of topology	Exact	Correct
IV	Refinement of capacity assignment	Exact	Correct, refined

The first and usually most important phase is a Genetic Algorithm which provides a good basic topology, based on a simplified cost model and an approximate capacity assignment procedure. The quality of this basic topology is checked and eventually improved in a second phase. The third phase considers the correct cost model and capacity assignment procedure. Again, the influence on the topology is checked. The working and spare capacity assignment is refined in the last phase, using a fixed topology.

4.1. Phase I: creation of basic topology

In a first phase, a starting topology is built and an estimate of the needed link capacities is made. The cost of the line systems is approximated by a function which is proportional to the capacity on the link—based on the STM-16 fiber system costs—and the regenerator cost is assumed to be proportional to both the capacity and the length of the link. Furthermore, several simulations have shown that applying a bi-routing strategy represents a rather good approximation for the exact spare capacity assignment procedure based on link restoration, as long as the total amount of needed capacity remains roughly the same (i.e., if the factor η, see Section 2, is properly chosen). A typical example is shown in figure 5.

Based on a randomly chosen topology, the factor η was set to match the total network costs for the bi-routing and link restoration case. For this fixed value of η, the total network

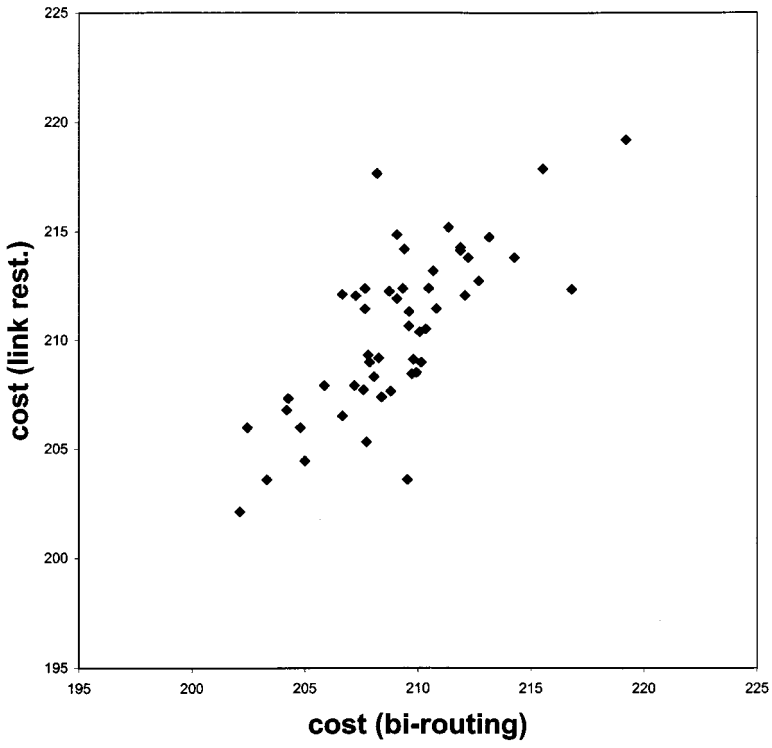

Figure 5. Network cost for a random set of topologies: bi-routing case vs. link restoration case.

costs were then compared for a number of topologies, obtained by randomly applying some minor modifications on the initial topology. Most scattered points in the figure are located close to the first bisector, which illustrates the high correlation of the network cost $c(N)$ between the bi-routing case and the link restoration case. This relationship allows us to reduce the calculation time by appling the bi-routing strategy in the first phase of the algorithm. The Genetic Algorithm described in Section 2 is used to solve the resulting ACSND problem.

4.2. Phase II: local optimization of topology (optional)

In a second phase, we still consider the ACSND problem. We try to improve the topology found in the first phase using local improvement techniques. Several possibilities were implemented: removing a link, adding a link or replacing a link by a new link. To restrict the number of possible link replacements several criteria could be used, e.g., only substituting a link l_x by a link l'_x if l_x and l'_x have a common node, introducing an upper bound on the length of the link l'_x, introducing an upper bound on the angle between the links l_x and l'_x.

The decision which local optimization techniques will be used is based on a trade-off between the desired quality of the final solution and the calculation time. This second phase may be omitted if the Genetic Algorithm already performed a thorough search through the solution space, which usually yields a solution that cannot be improved anymore by the above mentioned local optimization techniques. Especially if the relative importance of cable installation cost is high when compared with the line system cost, the Genetic Algorithm outperforms the local optimization technique by far.

4.3. Phase III: local optimization of topology using more detailed description

The third phase considers the correct cost model (as described in Section 1) which incorporates the discrete nature of capacity (STM-1, STM-4 and STM-16) and length (regenerators placed at regular distances). Furthermore, the exact capacity allocation procedure is applied: the working capacity suffices to route the demand through the network during normal operation and the spare capacity allows to reroute the traffic affected by a single link failure using a link restoration scheme. This phase is meant to examine the influence of the changes with respect to the cost model and capacity assignment on the topology. We again try to improve the topology found in the second phase by removing, adding or replacing a link.

Extremely long calculation times to evaluate the overall network cost for a certain topology are avoided by fixing the routing strategy for the demand (working capacity along shortest path) and by applying the rough SCA algorithm (see Subsection 3.3). Nevertheless, the evaluation time is usually much higher than for the bi-routing based capacity assignment procedure used in the first two phases. This indicates the benefit of running the second phase before starting the third phase: if necessary, some serious shortcomings on the topology can be solved quickly in the second phase and only the 'finishing touch' on the topology is performed by the slower third phase.

4.4. Phase IV: refinement of capacity assignment

This final phase leaves the topology unchanged and refines the working and spare capacity assignment, respectively.

For the working capacity assignment, we consider two possible paths for each node-to-node demand: the shortest path and the shortest path which is link-disjoint from this shortest path.[7] By dividing each demand between its two possible paths, we try to obtain a cheaper overall (working and spare) capacity allocation. The refinement of the working capacity is performed in two steps. First we try to reduce the overall cost by switching a node-to-node demand completely from its present path to its alternative path and recalculating the spare capacity based on this altered working capacity allocation, using the rough SCA algorithm. Since the switching of a node-to-node demand from its shortest path to its alternative path tends to be more effective if the alternative path is as short as possible, the node-to-node demands are enumerated according to increasing length difference between shortest and alternative path. This procedure is repeated until any switching of a node-to-node demand yields a solution which is at least as expensive as before: a local optimum is reached. In the second step, we examine what happens when we switch one unit from the present path to the alternative path or vice versa, until no improvements can be found anymore. Since no fractional division of demands is allowed, the working capacity refinement stops here. Note that this working capacity refinement is in fact also based on the Zoom-In strategy: we first examine some drastic changes on the working capacity and only if the local optimum is reached, the influence of small changes is checked.

Once the working capacity is fixed, we try to refine the spare capacity allocation by examining the influence of the granularity of the redundant spare units (see Subsection 3.3, refined tightening phase). The tightest spare capacity allocation is retained.

5. Numerical examples

The Zoom-In algorithm was tested on several quite different problem situations: we studied situations where no cables are present yet (so-called ‘green-field’ or ‘desert’ problem instances) as well as situations with some cables already installed and we varied the scale of the network (international, national and regional networks), the size of the network (8 to 50 nodes) and the density of the demand matrix. The calculation times ranged from a few seconds to about 24 hours,[8] depending mainly on the size of the network, but also on the number of commodities and the relative importance of cable cost versus line system cost. For all these problem instances, the memory requirements stayed largely below the available resources.

As an example, two 20-node green-field problem instances are discussed in detail below: a national and a regional network. The problems are identical with respect to the node locations and demand matrix (39 commodities, ranging from 1 to 20 VC-4’s), but the scale is different: the network diameters are 300 km and 30 km, respectively. The cable cost is 2 MBF/km. Three types of fiber systems were considered, the corresponding costs are mentioned in table 3. The regenerator interleaving distance is 50 km, independent from the fiber system.

Table 3. Cost of line systems and regenerators for different fiber systems.

	Line systems (MBF)	Regenerators (MBF)
STM-1	1.0	0.025
STM-4	1.8	0.050
STM-16	3.5	0.100

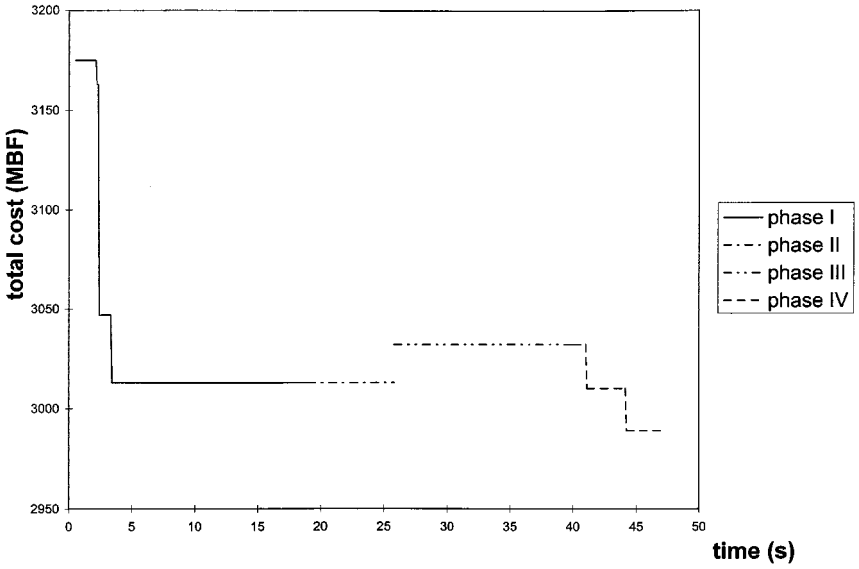

Figure 6. Evolution of the overall network cost (national problem instance).

5.1. Case study I: national network

The evolution of the overall cost of the temporary solution is shown in figure 6. The Genetic Algorithm (phase I) founds a first feasible solution with a cost of 3175 after 0.5 seconds. Gradually, better solutions are found until an excellent solution with a cost of 3013 is encountered after 3.4 seconds. No better solutions are found until the end of the Genetic Algorithm and the found solution turns out to be optimal when compared with 'neighbor' networks with one link more, one link less or one replaced link (phase II). After 25.8 seconds the first two phases of the Zoom-In algorithm are finished. From the third phase on, the correct cost model and capacity assignment procedure are introduced. This transition of cost model and capacity assignment procedure leads to a gap in the cost evolution from 3013 to 3032. Again, the algorithm searches for better neighboring networks (phase III), in vain. In the last phase, the capacity assignment is refined, leading to a first improvement by switching the working path of a node-to-node demand completely to its alternative path

Table 4. Components of national network.

Component	Number	Cost (MBF)
Cable	1200.4 km	2400.8
STM-1 line systems	1	1
STM-1 regenerators	1	0.025
STM-4 line systems	0	0
STM-4 regenerators	0	0
STM-16 line systems	165	577.5
STM-16 regenerators	109	10.9
Total		2990.225

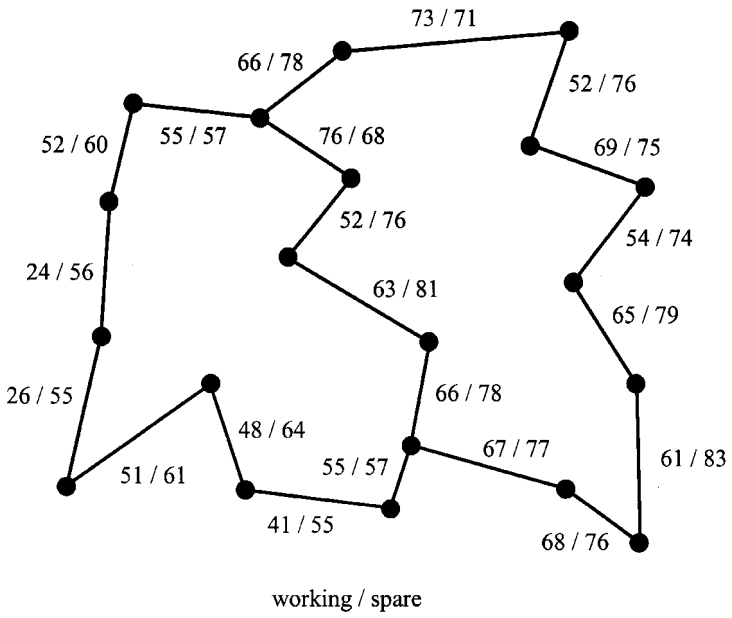

Figure 7. National network design (topology, working and spare capacity).

(cost 3010) and a second improvement is performed by switching 3 units from a working path to an alternative path (cost 2990). Varying the granularity of the redundant spare units in the SCA algorithm does not yield better solutions and the program terminates after 47.2 seconds. The final topology and capacity allocation are shown in figure 7, the different components of the overall cost are mentioned in table 4.

When examining the evolution of the overall network cost, we notice that the topology found by the Genetic Algorithm in phase I is not improved by the second and third phase.

This is caused by several reasons. Firstly, several experiments showed that the Genetic Algorithm performs best in situations where the relative importance of the cable cost versus the line system cost is high—as is the case here: ratio 80%/19%, see table 4—, leading to sparse networks. Furthermore, a local optimization technique based on adding, removing or replacing a link is more powerful in the opposite situation of dense networks.[9] This examination of the strengths and weaknesses of the first two phases explains the lack of improving moves in phase II. Since we are dealing with a cost dominated by the cable installation here, the adaptation of the capacity assignment procedure from a bi-routing strategy towards a working and spare capacity allocation based on link restoration has no big influence on the cost of a solution. As a consequence, the topology found in phase I turns out to be locally optimal with respect to the correct capacity assignment procedure too (phase III).

When considering the type of line systems used to build the network (see table 4), a clear predominance of STM-16 line systems is noticed. This is due to the high demand requirements on one hand and the economy of scale principle on the other hand: two STM-4 line systems (providing 8 VC-4 channels) are more expensive than 1 STM-16 line system (providing 16 VC-4 channels). As a consequence, the working paths and certainly the spare paths strive towards a distribution throughout the network that fills the available STM-16 fiber systems as good as possible and uses as few STM-4 and STM-1 line systems as possible.

The cost of the regenerators is only 0.4% of the total cost, hence the influence on the topology choice and capacity allocation is almost negligible. Regenerators are only important when international networks with high demands are considered (long distances combined with big capacity requirements).

5.2. Case study II: regional network

When tackling the 30 km diameter problem instance, the overall cost of the temporary solution changes as shown in figure 8.

The final topology and capacity allocation are shown in figure 9. Table 5 mentions the different components of the overall cost (no regenerators are needed: short distances).

Now the relative importance of cable cost versus line system cost is much smaller (due to the smaller distances): the final solution shows a ratio of 52%/48%. As a consequence the topology is much denser (see figure 9).

When we compare the evolution of the algorithm for the regional network (figure 8) with the national network (figure 6), some striking differences are noticed. Firstly, the total calculation time is much bigger for the regional network: a problem instance with about a fifty-fifty ratio of cable cost versus line system cost is usually much harder to solve than a problem instance dominated by the cable cost. Secondly, the relative importance of the four phases has changed. Whereas the second and the third phase were in fact useless for the national problem instance, these phases clearly show their use for this example. Indeed, local optimization techniques based on adding, removing or replacing a link become more powerful when we are dealing with dense networks.

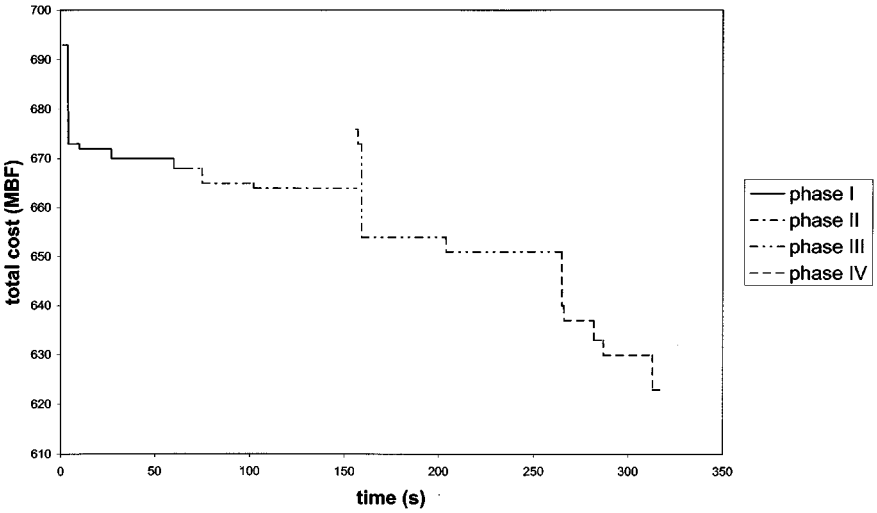

Figure 8. Evolution of the overall network cost (regional problem instance).

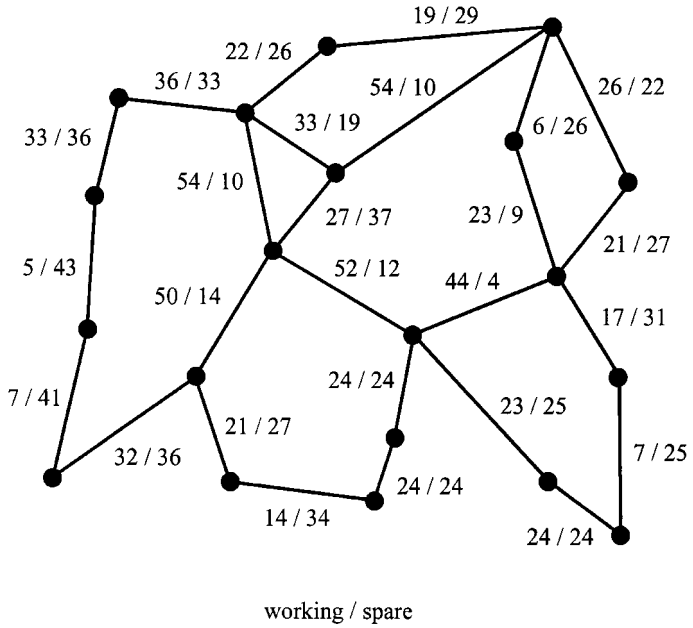

Figure 9. Regional network design (topology, working and spare capacity).

Table 5. Components of regional network.

Component	Number	Cost (MBF)
Cable	161.5 km	323.0
STM-1 line systems	2	2
STM-4 line systems	4	7.2
STM-16 line systems	83	290.5
Total		622.7

6. Comparison with other techniques

In this section the results of the main building blocks of the algorithm and the overall Zoom-In algorithm are compared with some results found in literature.

6.1. ACSND problem

One possible way to trace the performance of a Genetic Algorithm is to compare the obtained results with optimal solutions found with Integer Linear Programming techniques (see for instance also Karunanithi and Carpenter (1997) for such a comparison). However, due to the intrinsic complexity of the ACSND problem, only for some special cases (e.g. the Traveling Salesman Problem) we were able to find guaranteed optimal solutions using an integer programming code based on the work done in Stoer (1992) and Grötschel, Monma, and Stoer (1995). In these simple cases, the Genetic Algorithm reached the global optimum rather quickly. In more general cases an optimality check was not possible. However, an examination of the evolution of chromosomes in the Genetic Algorithm shows that the best solution is not found 'by accident': a lot of chromosomes, mostly obtained after several crossover operations, are nearly as good as the best solution.[10] Furthermore, the strength of the Genetic Algorithm for sparse networks, eventually combined with the strength of the local optimization search in phase II for dense networks, gives rise to an algorithm which is suited for a broad spectrum of ACSND problems.

6.2. Spare capacity assignment

The SCA algorithm allocates the spare capacity in a network with link modularity (e.g. 16 for STM-16 fiber systems) to allow full link restorability of single link failures. This problem was also tackled in Sakauchi, Nishimura, and Hasegawa (1990) and Grover, Bilodeau, and Venables (1991), describing a LP approach and a heuristic algorithm SLP for the near-optimal assignment of spare capacity, respectively. These papers mention a problem instance with 11 nodes and 23 links based on OC-12 link modularity. The solution based on a LP approach uses 164 OC-12 fiber systems (see figures 3 and 4 in (Sakauchi, 1990)), the SLP algorithm yields a slightly better solution with 163 fiber systems (see figure 3b in

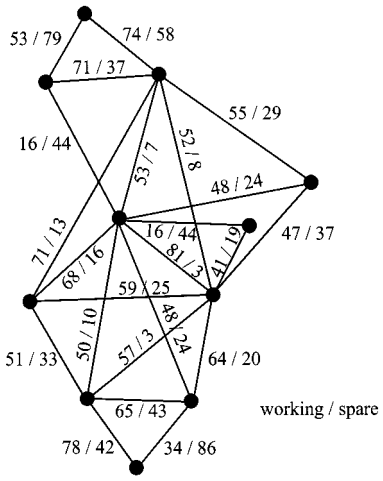

Figure 10. Spare capacity allocation (mod. 12) by SCA algorithm.

(Grover, 1991)). We tested the refined[11] SCA algorithm on this problem instance, yielding a capacity assignment that uses 163 OC-12 fiber systems (see figure 10). This solution is capacity-equivalent but not identical to the SLP solution and slightly better than the LP based solution, using very few calculation time resources (1.1 seconds).

6.3. Working and spare capacity assignment

The overall capacity assignment procedure, as described in Subsection 4.4, is based on the assumption that considering only two disjoint paths for the working traffic suffices to create a cost-effective capacity distribution. Stated otherwise, considering a much higher degree of bifurcation of the working traffic (i.e. considering a higher number of possible working paths) usually won't lead to a big additional reduction of the overall capacity cost. In fact, the philosophy behind this approach is the flexible nature of spare capacity, which allows to make efficient use of the available capacity due to the imperfect filling of the last *m*-unit on a link by the working capacity and moreover allows to compensate for local accumulations of working capacity.

To verify this assumption, some experiments were carried out on a set of random problem instances. We compared the results with an ILP method which constructs the optimal solution of a related problem: the cheapest capacity allocation is constructed which allows a routing of the traffic under normal conditions and provides enough spare capacity to ensure full link restorability of single link failures, when *fractional* flow variables are allowed (Poppe, 1997). The set of possible solutions of the latter problem includes all possible solutions of the SDH capacity problem—which requires integer flow variables—, hence the solution found with the ILP technique provides a lower bound for the SDH problem. The average gap between the solutions found with the method of Subsection 4.4 and the ILP solutions was 1.1%. The ILP solutions usually contained a lot of non-integer

Table 6. Comparison of (working + spare) capacity assignment with ILP lower bounds.

	Lower bound (ILP)	Found solution (phase IV)	Gap (%)
Case study I	166	166	0.0
Case study II	86	87	1.2

flow variables, so the actual gap between the found solution and the optimal solution might even be smaller. From these experiments, we can conclude that considering only two disjoint paths for the working traffic usually has little or no influence on the solution quality.

As an example, the number of STM-16 line systems obtained in the (slightly adapted)[12] case studies I and II (see Section 5) were also compared with the lower bounds obtained by the ILP method, see table 6.

6.4. Overall problem: design of SDH mesh restorable network

The Zoom-In approach can be described as an integrated approach with some adaptations to avoid excessive calculation time. The algorithm described in Section 4 is in fact based on two main assumptions:

1. The exact type of capacity assignment (based on bi-routing in phases I and II, exact working and spare assignment in phase III and IV) has no big influence on the choice of the topology or at least, the drawbacks of a topology choice based on an approximate capacity assignment can be eliminated almost completely by local improvement of the topology in phase III.
2. The fast capacity allocation procedure used in phase III (working capacity along shortest path, rough SCA algorithm to assign spare capacity) is sufficiently accurate to measure the influence of the capacity on the topology choice. As a consequence, the refined capacity allocation procedure in phase IV can be carried out on a fixed topology.

To give an impression of the impact of these assumptions on the calculation time, we estimated the time needed to design a SDH mesh restorable network for case study I (see Subsection 5.1) when all capacity allocation refinements were considered from the beginning of the algorithm: 2300 seconds instead of 47 seconds! It is clear that this integrated approach does not leave us any hope of tackling large problem instances (e.g. 50 nodes) within a reasonable calculation time.

To check whether the assumptions mentioned above are justified, some experiments were carried out to get an indication of the quality of the overall solution. Due to the intrinsic complexity of the overall problem, verifying the optimality of the solutions of the Zoom-In algorithm is only possible for small problem instances. The algorithm was tested on a problem instance with 8 nodes, 13 candidate links and 13 commodities, taken from Gavish et al (1989). This problem was solved to optimality in Poppe and Demeester (1997), using

Table 7. Comparison of solution quality of ILP algorithm and Zoom-In algorithm (same calculation times).

	ILP solution (MBF)	Zoom-In solution (MBF)	Gap (%)
Case study I	3038	2981	1.2
Case study II	649	633	2.5

a sophisticated branch-and-cut-and-price algorithm, based on an integrated ILP approach. The result obtained with the Zoom-In algorithm was 562.2 (calculation time: 9.1 seconds), the global optimum for this problem instance.

The Zoom-In algorithm and the branch-and-cut-and-price were also compared for a set of bigger random problem instances. Due to time and memory restrictions it was not possible to search the whole branching tree for these bigger problems: the ILP algorithm was ended after examining a number of branching nodes. To be able to compare the quality of the results, the calculation time of the Zoom-In algorithm was also restricted (by decreasing the number of possible moves in local optimization procedures) not to exceed the time needed by the ILP algorithm. On the average, the solutions found with the Zoom-In algorithm were 1.3% cheaper than the ILP solutions (standard deviation: 0.85%). The simulation results show the intrinsic advantage of a Zoom-In approach when compared to an integrated approach: by carefully balancing the impact of introducing a simplifying assumption on the solution quality on one hand and the calculation time on the other hand in each phase of the algorithm, a method can be developed which produces better results than an integrated approach within the same time. Furthermore, the memory requirements of the Zoom-In algorithm are very modest, even for large problem sizes (e.g., 50 nodes). In realistic situations, the calculation time will be the limiting factor.

As an example, the results for the two (slightly adapted)[13] case studies in Section 5 are compared in table 7.

7. Conclusions

An algorithm has been developed for the near-optimal design of SDH mesh restorable networks. The technique is based on several building blocks such as a type of Genetic Algorithm to design an initial topology, local optimization procedures to adapt the topology to a specific capacity assignment procedure and a SCA algorithm to assign spare capacity to ensure full link restorability of single link failures. To combine these building blocks into an overall method, a Zoom-In strategy is applied: the algorithm gradually moves from a rough solution based on an approximate problem description towards a refined solution taking all problem details into account. This novel approach, which forms a compromise between a sequential and an integrated method, aims at combining the advantages of both approaches.

The algorithm was tested extensively on several different problem instances and the results were compared with other techniques mentioned in literature. These experiments show that the Zoom-In algorithm is a quite promising technique which is able to deliver high quality solutions using modest time and memory requirements. Furthermore, the used

approach provides a flexible framework with respect to changing problem formulations: usually only some minor modifications are needed to tackle related problems. Ongoing work is studying the introduction of other restoration mechanisms and modeling more complex failure scenarios.

Acknowledgment

The authors like to thank Fabrice Poppe for providing Integer Linear Programming solutions to compare and test the performance of the Zoom-In algorithm. Part of this work has been supported by the Flemish Government through the IWT project ITA/950214/INTEC and by the European Commission through the ACTS project AC205/PANEL.

Notes

1. The described algorithm can also be used to design similar networks, e.g. based on the SONET technology.
2. With at least link-connectivity 2, to allow bi-routing along link-disjoint paths.
3. A bad part of a topology is a part that typically leads to solutions with link-connectivity <2 or a high network cost.
4. A chain $l_1 - l_2 -, \ldots, -l_i$ is an ordered list of links where l_x and l_{x+1} have a common node and the degree of this node is exactly 2, $\forall x \in \{1, 2, \ldots, i-1\}$.
5. An m-unit of capacity is a block of m capacity units (corresponding to one fiber system).
6. Another possibility to speed up the tightening phase is to add an intermediate phase between phase II and III, where we remove all m-units of spare capacity that are not at all used by the rerouting determined at the end of phase II (for any link failure). However, such an intermediate phase usually does not significantly reduce the spare capacity of phase II, since the considered rerouting pattern at the end of phase II is typically using up most of the spare capacity in the close neighborhood of the failing link and little or no spare capacity far from the failing link. This inefficient usage of the spare resources can not be corrected by the intermediate phase.
7. The requirement of the second path to be link-disjoint with respect to the first path is based on the intuitive notion that this choice allows a thorough redistribution of the working capacity allocation, if needed.
8. All calculation times mentioned in this paper are measured on a Pentium Pro 200 Mhz PC with 32Mbyte internal memory.
9. As an example, we also studied the situation where phase I was omitted: phase II starts from a topology without links and moves towards better networks using local optimization. This way, the first feasible result was obtained after 7.5 seconds (cost 4101) and the final result of phase II was 3047 after 130 seconds, i.e. 1.1% worse than the result based on phase I using up much more calculation time. Several tests on other random problem instances confirmed this behavior, clearly indicating that the Genetic Algorithm generally outperforms the local optimization technique by far in the case of sparse networks.
10. We refer to Pickavet et al. (1997) for a more detailed discussion of the solution quality obtained with the Genetic Algorithm.
11. I.e. in phase III of the SCA algorithm, we repeatedly remove one 12-unit of spare capacity from the link with most redundant spare *units*.
12. We considered only STM-16 fiber systems and the regenerator cost was neglected. As can be seen from the results in section 5 (Tables 3 and 4), these restrictions have no big influence on the final result.
13. Again, only STM-16 fiber systems were considered and the regenerator cost was neglected.

References

Ahuja, R.K., T.L. Magnanti, and J.L. Orlin. (1993). *Network Flows: Theory, Algorithms and Applications*. Englewood Cliffs, New Jersey: Prentice Hall.

Gavish, B., T. Trudeau, M. Dror, M. Gondreau, and L. Mason. (1989). "Fiberoptic Circuit Network Design Under Reliability Constraints," *IEEE Journal on Selected Areas in Communications* 7(8), 1181–1187.

Goldberg, A.V. and R.E. Tarjan. (1988). "A New Approach to the Maximum Flow Problem," *Journal of the ACM* 35, 921–940.

Grötschel, M., C.L. Monma, and M. Stoer. (1995). "Design of Survivable Networks." In M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser (eds.), *Handbooks in Operations Research and Management Science, Vol. 7*, Amsterdam: North-Holland, pp. 617–672.

Grover, W.D., T.D. Bilodeau, and B.D. Venables. (1991). "Near Optimal Spare Capacity Planning in a Mesh Restorable Network." In *Proc. of IEEE Globecom*, pp. 2007–2012.

Herzberg, M. (1993). "A Decomposition Approach to Assign Spare Channels in Self-Healing Networks." *Proc. of IEEE Globecom*, pp. 1601–1605.

Herzberg, M., S. Bye, and A. Utano. (1995). "The Hop-Limit Approach for Spare Capacity Assignment in Survivable Networks," *IEEE/ACM Transactions on Networking* 3(6), 775–784.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Karunanithi, N. and T. Carpenter. (1997). "SONET Ring Sizing with Genetic Algorithms," *International Journal on Computing and Operations Research* 24(6), 581–591.

Michalewicz, Z. (1997). *Genetic Algorithms + Data Structures = Evolution Programs*. London: Springer-Verlag.

Minoux, M. (1981). "Optimum Synthesis of a Network with Non-Simultaneous Multi-Commodity Flow Requirements." In P. Hansen (ed.), *Studies on Graphs and Discrete Programming*. North-Holland, pp. 269–277.

Monma, C.L. and D.F. Shallcross. (1989). "Methods for Designing Communications Networks with Certain Two-Connected Survivability Constraints," *Operations Research* 37(4), 531–541.

Pickavet, M., F. Poppe, J. Luystrermans, and P. Demeester. (1997). "A Genetic Algorithm for Solving the Capacitated Survivable Network Design Problem." In *Proc. of Fifth International Conference on Telecommunication Systems*, pp. 71–76.

Poppe, F. and P. Demeester. (1997). "The Design of SDH Mesh Restorable Networks: Problem Formulation and Algorithm." In *Proc. of Fifth International Conference on Telecommunication Systems*, pp. 88–97.

Poppe, F. and P. Demeester. (1998). "Economic Allocation of Spare Capacity in Mesh Restorable Networks: Model and Algorithm." In *Proc. of Sixth International Conference on Telecommunication Systems*, pp. 77–86.

Sakauchi, H., Y. Nishimura, and S. Hasegawa. (1990). "A Self-Healing Network with an Economical Spare-Channel Assignment." In *Proc. of IEEE Globecom*, pp. 438–443.

Sato, K.-I. (1996). *Advances in Transport Network Technologies, Photonic Networks, ATM and SDH*, Norwood: Artech House.

Stoer, M. (1992). *Design of Survivable Networks*. New York: Springer-Verlag.

Stoer, M. and G. Dahl. (1994). "A Polyhedral Approach to Multi-commodity Survivable Network Design," *Numerische Mathematik* 68, 149–167.

Wu, T.H. (1992). *Fiber Network Service Survivability*, London: Artech House.

Wu, T.H. (1995). "Emerging Technologies for Fiber Network Survivability," *IEEE Commun. Mag*. 33(2), 58–74.