

# Topological Design of Survivable IP Networks Using Metaheuristic Approaches\*

Emilio C.G. Wille<sup>1</sup>, Marco Mellia<sup>2</sup>, Emilio Leonardi<sup>2</sup>, and Marco Ajmone Marsan<sup>2</sup>

<sup>1</sup> Dipartimento di Elettronica, Politécnico di Torino, Italy

<sup>2</sup> Departamento de Eletrônica, Centro Federal de Educação Tecnológica do Paraná, Brazil

**Abstract.** The topological design of distributed packet switched networks consists of finding a topology that minimizes the communication costs by taking into account a certain number of constraints such as the end-to-end quality of service (e2e QoS) and the reliability. Our approach is based on the exploration of the solution space using metaheuristic algorithms (GA and TS), where candidate solutions are evaluated by solving CFA problems. We propose a new CFA algorithm, called GWFD, that is capable of assign flow and capacities under e2e QoS constraints. Our proposed approach maps the end-user performance constraints into transport-layer performance constraints first, and then into network-layer performance constraints. A realistic representation of traffic patterns at the network layer is considered as well to design the IP network. Examples of application of the proposed design methodology show the effectiveness of our approach.

## 1 Introduction

Today, with the enormous success of the Internet, packet networks have reached their maturity, and all enterprises have become dependent upon networks or networked computations applications. In this context the loss of network services is a serious outage, often resulting in unacceptable delays, loss of revenue, or temporary disruption. In an usual network-design process, were networks are designed to optimize a performance measure (e.g., delay, throughput) without considering possible network failures, the network performance can degrade drastically at an element failure. A network-design process based only on reliability considerations (e.g., connectivity, successful communication probability) does not necessarily avoid network performance degradation, despite the connections guarantees during the failure of one or several stations. To avoid loss of network services, communications networks should be designed so that they remain operational and maintain as high performance level as feasible, even in presence of network component failures.

The packet network design methodology that we propose in this paper is quite different from the many proposals that appeared in the literature. Those focus almost invariably on the trade-off between total cost and average performance (expressed in terms of average network-wide packet delay, average packet loss probability, average

---

\* This work was supported by the Italian Ministry for Education, University and Research under project TANGO. The first author was supported by a CAPES Foundation scholarship from the Ministry of Education of Brazil.

link utilization, network reliability, etc.). This may lead to situations where the average performance is good, but, while some traffic relations obtain very good quality of service (QoS), some others suffer unacceptable performance levels. On the contrary, our packet network design methodology is based on user-layer QoS parameters, and explicitly accounts for each source/destination QoS constraint.

From the end user's point of view, QoS is driven by e2e performance parameters, such as data throughput, web page latency, transaction reliability, etc. Matching the user-layer QoS requirements to the network-layer performance parameters is not a straightforward task. For example, the QoS perceived by end users in their access to Internet services is mainly driven by TCP, whose congestion control algorithms dictate the latency of information transfer. Indeed, it is well known that TCP accounts for a great amount of the total traffic volume in the Internet [1, 2].

According to the Internet protocol architecture, at least two QoS mapping procedures should be considered; the first one translates the application-layer QoS constraints into transport-layer QoS constraints, and the second translates transport-layer QoS constraints – such as *file transfer latency* ( $L_t$ ), or *file transfer throughput* ( $T_h$ ) – into network-layer QoS constraints, such as *Round Trip Time* ( $RTT$ ) and *packet loss probability* ( $P_{loss}$ ). In [3, 4], the authors describe *QoS translators* that are able to do this mapping process. While the first procedure is an ad-hoc one, the second is based on the numerical inversion of analytical TCP models presented in the literature.

The representation of traffic patterns inside the Internet is a particularly delicate issue, since it is well known that IP packets do not arrive at router buffers following a Poisson process [5], but a higher degree of correlation exists, which can be partly due to the TCP control mechanisms. This means that the usual approach of modeling packet networks as networks of M/M/1 queues [6–8] is not appropriate. In this paper we adopt a refined IP traffic modeling technique, already presented in [9], that provides an accurate description of the traffic dynamics in multi-bottleneck IP networks loaded with TCP mice and elephants. The resulting analytical model is capable of producing accurate performance estimates for general topology IP networks loaded by realistic TCP traffic patterns, while still being analytically tractable.

In addition, given that the network reliability depends on its components, it is necessary to consider the failure of one or several components in the design of such networks. In general, two disjoint physical paths  $p_1$  and  $p_2$  are associated with each existing path. In normal conditions, only path  $p_1$  is used to information exchange; however, enough capacity is reserved on  $p_2$  to be able to restore communication when one of the physical links (or nodes) belonging to  $p_1$  fails. Protection and/or restoration schemes normally guarantee the resilience to the failure of a single physical link (or node); thus, the capacity to be reserved in the network must be sufficient to successfully restore all the paths disrupted by a link (or node) failure. In this paper, the concept of 2-connectivity is used as a reliability constraint.

The challenge in the area of network design is to determine the less expensive solution to interconnect nodes, and assign flow and capacities, while satisfying the reliability and end-to-end (e2e) QoS constraints. This problem is called the TCFA (Topological, Capacity and Flow Assignment) problem. This is a combinatorial optimization problem classified as NP-complete. Polynomial algorithms which can find the optimal solution

for this problem are not known. Therefore, heuristic algorithms are applied, searching for solutions. In this paper we suggest two metaheuristic approaches: the *Genetic Algorithm* (GA) [10], and the *Tabu Search* (TS) algorithm [11] to address the topological design problem, where traffic is mostly due to TCP connections. GAs are heuristic search procedures which applies natural genetic ideas such as natural selection, mutations and survival of the fittest. The TS algorithm is an evolution of the classical Steepest Descent method, however thanks to an interior mechanism that provides to accept also worse solutions than the best solution found so far, it is less subject to local optima entrapments.

The evaluation of each candidate inside each metaheuristic algorithm is done considering the cost of the network obtained from the solution of its related Capacity and Flow Assignment (CFA) problem. In this paper, we present a nonlinear mixed-integer programming formulation for the CFA problem and propose an efficient heuristic procedure called Greedy Weight Flow Deviation (GWFD) to obtain CFA solutions.

When explicitly considering TCP traffic it is also necessary to tackle the Buffer Assignment (BA) problem, for which we propose an algorithm that computes solutions to the droptail case BA problem.

The rest of the paper is organized as follows. Section 2 outlines the problem and the solution approach adopted in this paper, In particular, the CFA and BA formulations are presented and the GWFD method is described. Section 3 describes fundamentals of GAs, their operating techniques, and synthesizes the adaptation and implementation details of the GAs applied to the topological design problem; it also presents the TS algorithm. Section 4 presents and analyzes computational results. Finally, Section 5 summarizes the main results obtained in this research.

## 2 Problem Statement

The topological design of packet networks can be formulated as a TCFA problem as follows: given the geographical location of the network nodes on the territory, the traffic matrix, the capacity costs; minimize the total link cost, by choosing the network topology and selecting link flows and capacities, subject to QoS and reliability constraints. As reliability constraint we consider that all traffic must be exchanged even if a single node fails (2-connectivity). There is a trade off between reliability and network cost; we note that more links between nodes imply more routes between each node pair, and consequently the network is more reliable; on the other hand, the network is more expensive. Finally, the QoS constrains correspond to maintain the e2e packet delay for each network source/destination pair below a maximum tolerable value.

Our solution approach is based on the exploration of the solution space (i.e., 2-connected topologies) using the metaheuristic algorithms (GA and TS). As the goal is to synthesize a network that remains connected despite one node failure, for each topology evaluation, actually, we construct  $n$  different topologies, that are obtained from the topology under evaluation by the failure of a node each time, and then for each topology we solve the CFA problem. Link capacities are set to the maximum capacity value found so far considering the set of topologies. Using these capacity values the objective function (network cost) is evaluated.

## 2.1 Traffic and Queuing Models

The representation of traffic patterns inside the Internet is a particularly delicate issue, since it is well known that IP packets do not arrive at router buffers following a Poisson process, see [5], but a higher degree of correlation exists, which can be partly due to the TCP control mechanisms. This means that the usual approach of modeling packet networks as networks of M/M/1 queues as discussed in [6–8, 12, 13] is not appropriate. In this paper we adopt a refined IP traffic modeling technique, already presented in [9], that provides an accurate description of the traffic dynamics in multi-bottleneck IP networks loaded with TCP mice and elephants. The resulting analytical model is capable of producing accurate performance estimates for general topology IP networks loaded by realistic TCP traffic patterns, while still being analytically tractable.

We choose to model the increased traffic burstiness induced by TCP by means of batch arrivals, hence using  $M_{[X]}/M/1/B$  queues. A wide range of investigations performed in [9] certify the accurate network layer performance estimates by considering  $M_{[X]}/M/1/B$  models. The batch size varies between 1 and  $W$  with distribution  $[X]$ , where  $W$  is the maximum TCP window size expressed in segments. The distribution  $[X]$  is obtained considering the number of segments that TCP sources send in one  $RTT$  [9]. Additionally, our choice of using batch arrivals following a Poisson process has the advantage of combining the nice characteristics of Poisson processes (analytical tractability in the first place) with the possibility of capturing the burstiness of the TCP traffic [9].

Considering the  $M_{[X]}/M/1/\infty$  queue, the average packet delay (queueing and transmission time) is given by:

$$E[T] = \frac{K}{\lambda} \frac{\rho}{1-\rho} = \frac{K}{\mu} \frac{1}{C-f} \quad (1)$$

with  $K$  given by:

$$K = \frac{m'_{[X]} + m''_{[X]}}{2m'_{[X]}} \quad (2)$$

where  $f$  is the average data flow on the link (bps),  $C$  is the link capacity (bps), the utilization factor is given by  $\rho = f/C$ , the packet length is assumed to be exponentially distributed with mean  $1/\mu$  (bits/packet),  $\lambda = \mu f$  is the arrival rate (packets/s),  $m'_{[X]}$  and  $m''_{[X]}$  are the first and second moments of the batch size distribution  $[X]$ .

## 2.2 Network Model

In the mathematical model, the IP network infrastructure is represented by a graph  $G = (V, E)$  in which  $V$  is a set of nodes (with cardinality  $n$ ) and  $E$  is a set of edges (with cardinality  $m$ ). A node represents a network router and an edge represents a physical link connecting one router to another. The output interfaces of each router is modeled by a queue with finite buffer.

Each network link is characterized by a set of attributes which principally are the flow, the capacity and the buffer size. For a given link  $(i, j)$ , the flow  $f_{ij}$  is defined as the effective quantity of information transported by this link, while its capacity  $C_{ij}$  is a

measure of the maximal quantity of information that it can transmit. Flow and capacity are both expressed in bits per second (bps). Each buffer can accommodate a maximum of  $B_{ij}$  packets, and  $d_{ij}$  is the link physical length.

The average traffic requirements between nodes are represented by a traffic matrix  $\hat{T} = \{\hat{\gamma}_{sd}\}$ , where the traffic  $\hat{\gamma}_{sd}$  between a node pair  $(s, d)$  represents the average number of bps sent from source  $s$  to destination  $d$ . We consider as traffic offered to the network  $\gamma_{sd} = \hat{\gamma}_{sd}/(1 - P_{loss})$ , thus accounting for the retransmissions due to the losses that flows experience along their path to the destination. The flow of each link that composes the topological configuration depends on the traffic matrix. We consider that for each source/destination pair  $(s, d)$ , the traffic is transmitted over exactly one directed path in the network (a minimum-cost routing algorithm is used). The routing and the traffic uniquely determine the vector  $\bar{f} = (f_1, f_2, \dots, f_m)$  where  $\bar{f}$  is a multicommodity flow for the traffic matrix; it must obey the law of flow conservation. A multicommodity flow results from the sum of single commodity flows  $f_{sd}$  with source node  $s$  and destination node  $d$ .

### 2.3 The Capacity and Flow Assignment Problem

Different formulations of the CFA problem result by selecting i) the cost functions, ii) the routing model, and iii) the capacity constraints; different methodologies must be applied to solve them. In this paper we focus on Virtual Private Networks (VPNs), in which common assumptions are i) linear costs, ii) non-bifurcated routing, and iii) continuous capacities.

The CFA problem requires the joint optimization of routing and link capacities. Our goal is to determine a route for the traffic that flows on each source/destination pair and the link capacities in order to minimize the network cost subject to the maximum allowable e2e packet delay. Let  $\delta_{ij}^{sd}$  be a decision variable which is one if link  $(i, j)$  is in path  $(s, d)$  and zero otherwise. Thus the CFA problem is formulated as the following optimization problem:

$$Z_{CFA} = \min \sum_{i,j} g(d_{ij}, C_{ij}) \quad (3)$$

subject to:

$$\sum_j \delta_{ij}^{sd} - \sum_j \delta_{ji}^{sd} = \begin{cases} 1 & \text{if } s = i \\ -1 & \text{if } t = i \\ 0 & \text{otherwise} \end{cases} \quad \forall (i, s, d) \quad (4)$$

$$K_1 \sum_{i,j} \frac{\delta_{ij}^{sd}}{C_{ij} - f_{ij}} \leq RTT_{sd} - K_2 \sum_{i,j} \delta_{ij}^{sd} d_{ij} \quad \forall (s, d) \quad (5)$$

$$f_{ij} = \sum_{s,d} \delta_{ij}^{sd} \gamma_{sd} \quad \forall (i, j) \quad (6)$$

$$C_{ij} \geq f_{ij} \geq 0 \quad \forall (i, j) \quad (7)$$

$$\delta_{ij}^{sd} \in \{0, 1\} \quad \forall (i, j, s, d) \quad (8)$$

The objective function (3) represents total link cost, which is a linear function of both the capacity  $C_{ij}$  and the physical length  $d_{ij}$  of link  $(i, j)$ , i.e.,  $g(d_{ij}, C_{ij}) = d_{ij}C_{ij}$ .

Constraint set (4) enforce flow conservation, defining a route for the traffic from a source  $s$  to a destination  $d$ . Non-bifurcated routing model is used where the traffic will follow the same path from source to destination. Equation (5) is the e2e packet delay constraint for each source/destination pair. It says that the total amount of delay experienced by all the flows routed on a path should not exceed the maximum  $RTT$ <sup>1</sup> minus the propagation delay of the route. Equation (6) defines the average data flow on the link. Constraints (7) and (8) are non-negativity and integrity constraints. Finally,  $K_1 = K/\mu$  and  $K_2$  is a constant to convert distance in time.

We notice that this problem is a nonlinear non convex mixed-integer programming problem. Other than the nonlinear constraint (5), it is basically a multicommodity flow problem. Multicommodity flow belongs to the class of NP-hard problems for which no known polynomial time algorithms exist. In addition, thanks to its non convex property there are in general a large number of local minima solutions. Therefore, in this paper we only discuss CFA sub-optimal solutions.

## 2.4 The Flow Deviation Method

Before presenting a new, fast CFA algorithm, we briefly review the classical Flow Deviation (FD) method [14], which allows to determine the routing of all the flows entering the network at different source/destination pairs; thus it solves the Flow Assignment (FA) problem.

It is well known that optimal routing directs traffic exclusively along paths which are shortest with respect to some *link weights* that depend on the flows carried by the links. This suggests that suboptimal routing can be improved by shifting the flow, for each source/destination pair, from a path to another with lower link weights values. This is the key idea in the FD method. The method starts from a given feasible path flow vector  $\bar{f}$  and, using a set of link weights, finds a new path for each source/destination pair. The weight  $L_{ij}$  of the link  $(i, j)$  is obtained as the partial derivative of the average total network delay  $T$  (based on  $M/M/1$  formulations) with respect to the flow rate  $f_{ij}$  which is traversing the link  $(i, j)$  evaluated at the current flow assignment  $f_{ij}$ . With this metric, the weight of a link represents the change in the objective function value if an incremental amount of the traffic is routed on that link. Then, the new flow assignment is determined by using the shortest path algorithm with the set of link weights  $L$ . The flow assignment is incrementally changed along the descent direction of the objective function. By incrementally changing the previous flow assignment into the new one, the optimal flow assignment is determined.

The FD method can be used, in a properly modified form, to solve the CFA problem; however it leads to local optima [14].

## 2.5 The Greedy Weight Flow Deviation Method

In this section we present a greedy heuristic to obtain solutions to the CFA problem presented in subsection 2.3. The key idea is to use a modified flow deviation method to find the flow distributions that minimize the network cost. The natural approach would

<sup>1</sup>  $RTT$  values are obtained by using the QoS translators given in [3, 4].

be to obtain the optimal values of the capacities  $C_{ij}$  as a function of the link flows  $f_{ij}$ , to substitute them in the cost function and finally (using partial derivatives) to obtain the link weights  $L_{ij}$ . Unfortunately, no closed form expression for the optimal capacities can be derived from our CFA formulation. However, we use a modified FD method in the following way.

First, it is straightforward to show that the link weights in the Kleinrock's method are given by:

$$L_{ij} = \frac{d_{ij}C_{ij}}{f_{ij}} \quad (9)$$

Second, in order to enforce e2e QoS delay performance constraints, the link capacities  $C_{ij}$  must be obtained using an e2e QoS CA problem solver (in this paper we use the approximate CA solution presented in [3]). As our new method relies on the greedy nature of the CA solver algorithm to direct computations toward a local optima, we called it the Greedy Weight Flow Deviation (GWFD) method.

As noted before, the CFA problem has several local optima. A way to obtain a more accurate estimate of the global optima is restart the procedure using random initial flows. However, we obtained very good results setting as initial trail  $L_{ij} = d_{ij}$ .

## 2.6 The Buffer Assignment Problem

As final step in our methodology, we need to dimension buffer sizes, i.e., to solve the following problem:

$$Z_{BA} = \min \sum_{i,j} B_{ij} \quad (10)$$

Subject to:

$$\sum_{ij} \delta_{ij}^{sd} p(B_{ij}, C_{ij}, f_{ij}, [X]) \leq P_{loss}, \quad \forall (s, d) \quad (11)$$

$$B_{ij} \geq 0, \quad \forall (i, j) \quad (12)$$

The objective function (10) represents the total buffer cost, which is the sum of the buffer values. Equation (11) is the loss probability constraint for each source/destination node pair. Where  $p(B_{ij}, C_{ij}, f_{ij}, [X])$  is the average loss probability for  $M_{[X]}/M/1/B$  queue, which is evaluated by solving its Continuous Time Markov Chain (CTMC). Constraints (12) are non-negativity constraints.

In the previous formulation we have considered the following upper bound on the value of loss probability for path  $(s, d)$  (constraint (11)).

$$\begin{aligned} \hat{P}_{loss} &= 1 - \prod_{i,j} (1 - \delta_{ij}^{sd} p(B_{ij}, C_{ij}, f_{ij}, [X])) \\ &\leq \sum_{ij} \delta_{ij}^{sd} p(B_{ij}, C_{ij}, f_{ij}, [X]) \end{aligned} \quad (13)$$

Consequently, the solution of the BA problem is a conservative solution.

The proof that the BA problem is a convex optimization problem is not a straightforward task. The difficulty derives from the need of showing that  $p(B, C, f, [X])$  is convex. Since, to the best of our knowledge, no closed form expression for the stationary distribution is known, no closed form expression for  $p(B, C, f, [X])$  can be derived. However, we conjecture that the BA problem is a convex optimization problem by considering that: (i) for an  $M/M/1/B$  queue,  $p(B, C, f)$  is a convex function; and (ii) approximating  $p(B, C, f, [X]) = \sum_{i=B}^{\infty} \pi_i$ , where  $\pi_i$  is the stationary distribution of an  $M_{[X]}/M/1/\infty$  queue, the loss probability is a convex function of  $B$ .

We can thus classify the BA problem as a multi-variable constrained convex minimization problem; therefore, the global minimum can be found using convex programming techniques. We solve the minimization problem applying first a constraints reduction procedure which reduces the set of constraints by eliminating redundancies. Then, the solution of the BA problem is obtained via the *logarithm barrier method* [15]

## 2.7 Numerical Examples: 40-Node Networks

To evaluate the performance of the new GWFD method, we run a large number of numerical experiments and computer simulations. We present results obtained considering several fixed topologies (40–node, 160–link each), which have been generated using the BRITE topology generator [16] with the router level option. Our goal is to obtain routing and link capacities. For each topology, we solved both the CFA and BA problems using the approaches described in previous sections.

Link propagation delays are uniformly distributed between 0.5 and 1.5ms, i.e., link lengths vary between 100 and 300 Km. Random traffic matrices were generated by picking the traffic intensity of each source/destination pair from a uniform distribution. The average source/destination traffic requirement was set to  $\gamma = 5$  Mbps. The file size follows a distribution, which is derived from one-week long measurements [2] (we recall that  $K$  and  $p(B, C, f, [X])$  are related to the file size distribution).

For all source/destination pairs, the target QoS constraints are: i) latency  $L_t \leq 0.2s$  for TCP flows shorter than 20 segments, ii) throughput  $T_h \geq 512$  kbps for TCP flows longer than 20 segments, iii)  $P_{loss} = 0.001$ . Using the transport-layer QoS translator (presented in [3, 4]), we obtain the equivalent network-layer performance constraint  $RTT \leq 0.032s$  for all source/destinations node pairs.

In Fig. 1, we compare network costs, considering 10 random topologies, obtained with four different techniques: i) Lagrangian relaxation (LB); ii) primal heuristic with logarithmic barrier CA solution (PH); iii) logarithmic barrier CA solution with minimum-hop routing (MinHop + CA); and iv) Greedy Weight Flow Deviation method (GWFD). A lower bound to the network cost is obtained by using the Lagrangian relaxation; using the second technique, feasible solutions are obtained from a sub-gradient optimization technique; the third technique just ignores the routing optimization when solving the CA problem (see [3, 4]).

We can observe that the GWFD solutions, for all considered topologies, always fall rather close to the lower bound (LB). The gap between GWFD and LB is about 13%. In addition, the GWFD algorithm is faster than the sub-gradient approach presented in [3, 4] (only 5 seconds of CPU time are needed to solve an instance with 40 nodes), while it obtains very similar results.

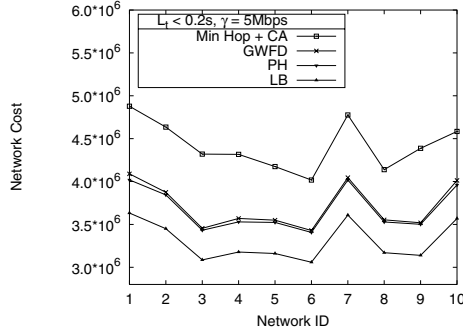


Fig. 1. Network Cost for 40–node Network Random Topologies.

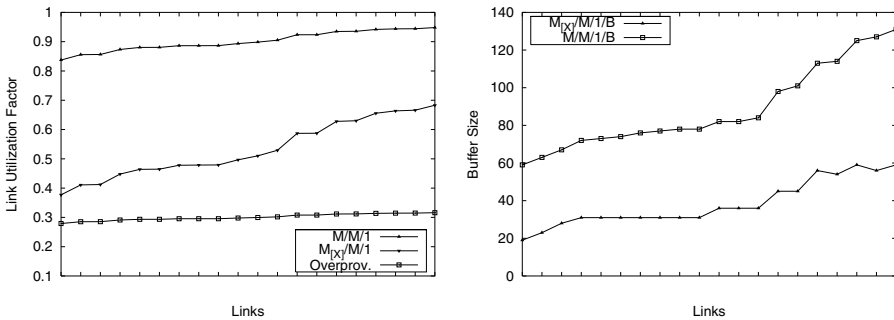


Fig. 2. Link Utilization Factors and Buffer Sizes for the 10–node network.

Avoiding to optimize the flow assignment subproblem results in more expensive solutions, as shown by the “Min Hop” routing associated with an optimized CA problem. This underlines the need to solve the CFA problem rather than a simpler CA problem.

To complete the evaluation of our methodology, we compare the link utilization factors and buffer sizes obtained when considering the classical  $M/M/1$  queueing model [14] instead of the  $M_{[X]}/M/1$  model. Fig. 2 shows the link utilizations (top plot) and buffer sizes (bottom plot) obtained with our method and with the classical model. It can be immediately noticed that considering the burstiness of IP traffic radically changes the network design. Indeed, the link utilizations obtained with our methodology are much lower than those produced by the classical approach, and buffers are much longer.

It is important to observe that the QoS perceived by end users in a network dimensioned using the classical approach cannot be tested by using simulations, because the loss probability experienced by TCP flows is so high that retransmissions cause the offered load to become larger than 1 for some links. This means that the network designed with the classical approach is not capable of supporting the offered load and therefore cannot satisfy the QoS constraints.

In addition in Fig. 2, we also compare our results to those of an overprovisioned network, in which the capacities obtained by using the traditional  $M/M/1$  model are multiplied a posteriori by the minimum factor which allows the QoS constraints to be met. The overprovisioning factor was estimated by a trial and error procedure based

on path simulations at the packet level that ends when the QoS constraints are satisfied. Since it is difficult to define an overprovisioning factor for the BA problem, we fixed a priori the buffer size to be equal to 150. The final overprovisioned network is capable of satisfying the QoS constraints, but a larger cost is incurred, which is directly proportional to the increase in link capacities. Note also that the heuristic used to find the minimum overprovisioning factor can not be applied for large/high-speed networks, due to scalability problem of packet level simulators.

### 3 Metaheuristic Search Algorithms

#### 3.1 Genetic Algorithms

Genetic Algorithms (GAs) are stochastic optimization heuristics in which explorations in solution space are carried out by imitating the population genetics stated in Darwin's theory of evolution. To use a genetic algorithm, is necessary represent a solution to a problem as a genome (or *chromosome*). In a problem those parameters which are subject to optimization (e.g., delay, throughput) constitute the phenotype space. On the other hand, the genetic operators work on abstract mathematical aspects like binary strings (e.g., chromosomes), the genotype space. *Selection*, *Genetic Operation* and *Replacement*, directly derived by from natural evolution mechanisms are applied to a population of solutions, thus favoring the birth and survival of the best solutions. GAs are not guaranteed to find the global optimal solution to a problem, but they are less susceptible to getting entrapped at local optima. They are generally good at finding "acceptably good" solutions to problems in "reasonable" computing times. Genetic Algorithms have been successfully applied to many NP-hard combinatorial optimization problems, in several application fields such as business, engineering, and science. But, during the last decade, the application of GAs to the telecommunication problem domain has started to receive significant attention.

The population comprises a group of  $N_I$  individuals (chromosomes) from which candidates can be selected for the solution of a problem. Initially, a population is generated randomly. The *fitness* values of the all chromosomes are evaluated by calculating the objective function in a decoded form (phenotype). A particular group of chromosomes (parents) is selected from the population to generate the offspring by the defined genetic operations (*mutation* and *crossover*). The fitness of the offsprings is evaluated in a similar way to their parents. The chromosome in the current population are then replaced by their offspring, based on a certain replacement strategy. Such a GA cycle is repeated until a desired termination criterion is reached (for example, a predefined number of generations  $N_G$  is produced). If all goes well throughout this process of simulated evolution, the best chromosome in the final population can become a highly evolved solution to the problem.

In the following paragraphs, we describe various techniques that are employed in the GA process for encoding, fitness evaluation, parent selection, genetic operation, and replacement.

**Encoding Scheme:** Its should be noted that each chromosome represents a trial solution to the problem setting. To enhance the performance of the algorithm, a chromosome

representation that stores problem-specific information is desired. The chromosome is usually expressed in a string of variables, each element of which is called a gene. The variable can be represented by binary, real number, or other forms and its range is usually defined by the problem.

Manipulation of topologies with the genetic algorithm requires that they are represented in some suitable format. Although more compact alternatives are possible, the characteristic matrix of a graph is quite an adequate representation. A graph is represented by an  $n \times n$  binary matrix, where  $n$  is the number of nodes of the graph. A “1” in row  $i$  and column  $j$  of the matrix stands for an arc from node  $i$  to node  $j$  and a “0” represents that node  $i$  and node  $j$  are not connected.

**Fitness Evaluation:** The objective function is a main source to providing the mechanism for evaluating the status of each chromosome. It takes a chromosome (or phenotype) as input and produces a number or list of numbers (objective value) as a measure to the chromosome’s performance. The evaluation of the objective function is usually the most demanding part of a GA program because it has to be done for every individual in every generation. In this paper, a fitness function, which is an estimation of the goodness of the solution for the topological design problem, is inversely proportional to the objective function value (cost). The lower the cost the better the solution is.

**Parent Selection:** Parent selection emulates the survival-of-the-fittest mechanism in nature. The choice of parents to produce offspring is somewhat more challenging than it might appear. Simply choosing the very best individual and breeding from that will not generally work successfully, because that best individual may have a fitness which is at a local, rather than a global optimal. Instead, the GA ensures that there is some diversity among the population by breeding from a selection of the fitter individuals, rather than just from the fittest.

There are many ways to achieve effective selection, including *proportionate schemes*, *ranking*, and *tournament* [10]. In this paper tournament selection is used. In this approach pairs of individuals are picked at random and the one with the higher fitness (the “winner of the tournament”) is used as one parent. The tournament selection is then repeated on a second pair of individuals to find the other parent from which to breed.

**Genetic Operations:** Crossover is a recombination operator that combines subparts of two parent chromosomes to produce offspring that contain some parts of both parents’ genetic material. The probability,  $p_c$ , that these chromosomes are recombined (mated) is a user-controlled option and is usually set to a high value (e.g., 0.95). If they are not allowed to mate, the parents are placed into the next generation unchanged. A number of variations on crossover operations are proposed, including *single-point*, *multipoint*, and *uniform* crossover [10]. In this paper single-point crossover is used. In this case, two parents are selected based on the above mentioned selection scheme. A crossover point is randomly selected and the portions of the two chromosomes beyond this point are exchanged to form the offspring.

Mutation is used to change, with probability  $p_m$ , the value of a gene (a bit) in order to avoid the convergence of the solutions to “bad” local optima. As a population

evolves, there is a tendency for genes to become predominant until they have spread to all members. Without mutation, these genes will then be fixed for ever, since crossover alone cannot introduce new gene values. If the fixed value of the gene is not the value required at the global optimal, the GA will fail to optimize properly. Mutation is therefore important to “loosen up” genes which would otherwise become fixed. In our experiments good results were obtained using a mutation operator that simply changes one bit, picket at random, for each produced offspring.

**Repair Function:** Simple GAs assume that the crossover and mutation operators produce a high proportion of feasible solutions. However, in many problems, simply concatenating two substrings of feasible solutions, or modifying single genes do not produce feasible solutions. In such cases, there are two alternatives. If the operators produce sufficient number of feasible solutions, it is possible to let the GA destroy the unfeasible ones by assigning them low fitness values. Otherwise, it becomes necessary to modify the simple operators so that only feasible individuals result from their application. In our crossover operation we employ a simple repair function which aims to produce 2-connected (feasible) topologies. If the repair operation is unsuccessful the parents are considered as crossover outputs.

**Replacement Strategies:** After generating the subpopulation (offspring), two representative strategies can be proposed for old generation replacement: *Generational-Replacement* and *Steady-State reproduction* [10]. We use Generational-Replacement. In this strategy each population size  $N_I$  generates an equal number of new chromosomes to replace the entire population. It may make the best member of the population fail to reproduce offspring in the next generation. So the method is usually combined with an *elitist strategy* where once the sons’ population has been generated, it is merged with the parents’ population according to the following rule : only the best individuals present in both sons’ population and parents’ population enter the new population. The elitist strategy may increase the speed of domination of a population by a super chromosome, but on balance it appears to improve the performance.

**Population Size:** Although biological evolution typically takes place with millions of individuals in the population, GAs work surprisingly well with quite small populations. Nevertheless, if the population is too small, there is an increased risk of convergence to a local optimal; they cannot maintain the variety that drives a genetic algorithm’s progress. As population sizes increase, the GA discovers better solutions more slowly; it becomes more difficult for the GA to propagate good combinations of genes through the population and join them together.

A simple estimate of appropriate population size for topological optimization is given by  $N_I \geq \frac{\log(1-p^{\frac{1}{q}})}{\log(1-\frac{1}{n(n-1)})}$  where  $N_I$  is the population size,  $n$  is the number of nodes,  $q$  is the number of links, and  $p$  is the probability that the optimal links occurs at least once in the population.

The choice of an appropriate value for  $q$  is based on the behavior of the topological optimization process, which can change the network number of links in order to reduce the network cost. We, therefore, set  $q$  as the minimum number of links that potentially can maintain the network 2-connectivity.

### 3.2 Tabu Search Algorithm

The second heuristic we propose relies on the application of the Tabu Search (TS) methodology. The TS algorithm can be seen as an evolution of the classical local optimal solution search algorithm called Steepest Descent. However, thanks to the interior mechanism that provides to accept also worse solutions than the best solution found so far, it is not subject to local minima entrapments. TS is based on a partial exploration of the space of admissible solutions, finalized to the discovery of a good solution. The exploration starts from an initial solution that is generally obtained with a greedy algorithm and when a stop criterion is satisfied (e.g., a number maximum of iterations  $N_T$ ), the algorithm returns the best visited solution. During the search in the space of the admissible solutions, it is possible the utilization of some exploration criterion called *intensification* and *diversification* criterion. The first one it used when a very careful exploration of a small part of the space solution is required, while the second one is used to jump into another place of the space solution to better cover that.

For each admissible solution, a class of neighbor solutions is defined. A neighbor solution, is defined as a solution that can be obtained from the current solution by applying an appropriate transformation of that called *move*. The set of all the admissible moves uniquely defines the *neighborhood* of each solution. The dimension of the neighborhood is  $N_N$ . In this paper a simple move is considered, it either removes an existing link or adds a new link between network nodes.

At each iteration, all solutions in the neighborhood of the current one are evaluated, and the best is selected as the new current solution. A special rule, the *tabu list*, is introduced in order to prevent the algorithm to deterministically cycle among already visited solutions. Tabu list tracks the last accepted moves so that a stored move in it cannot be used to generate a new move for a duration of a certain number of iterations. Therefore it may happen that TS continuing the search will select an inferior solution because better solutions are tabu. The choice of the tabu list size is important: a small size could cause the cyclic visitation of the same solutions while a big one could block for many iterations the optimization process, avoiding a good visit of the space solution.

### 3.3 Time Complexity

We now discuss the complexity of the heuristics that were described before. The GA time complexity, considering a simple GA, is  $O(N_G \cdot [N_I \cdot (\psi_{ps} + \psi_{go} + \psi_{of}) + \psi_{rp}])$ . Where  $\psi_{ps}$ ,  $\psi_{go}$ ,  $\psi_{of}$  are time complexities related to the following GA tasks: parent selection, genetic operators, and individual (objective function) evaluation, respectively. These tasks are done for each of the  $N_I$  individuals. Afterwards the replacement, with complexity  $\psi_{rp}$ , is required. Finally, all these operations is made for each of the  $N_G$  generations.

The complexity of the tabu search is evaluated as follows. At each iteration the neighborhood generation and evaluation of  $N_N$  solutions are necessary; this requires  $O(\psi_{ng} + N_N \cdot \psi_{of})$  operations, since the generation of the neighborhood has complexity  $\psi_{ng}$ , and the evaluation of each solution has complexity  $\psi_{of}$ . If the number of iterations is  $N_T$ , the resulting complexity is  $O(N_T \cdot [\psi_{ng} + N_N \cdot \psi_{of}])$ .

## 4 Selected Results

In this section we present some selected numerical results considering network designs obtained with the metaheuristic approaches. We consider the same mixed traffic scenario where the file size follows the distribution shown in [2].

The first set of experiments is aimed at investigating the performance of GA and TS algorithms. As a first example, we applied the proposed methodology to set up a 10-node VPN network over a given 25-node, 140-link physical topology. The target QoS constraints for all traffic pairs are: i) file latency  $L_t \leq 1$ s for TCP flows shorter than 20 segments, ii) throughput  $T_h \geq 512$  Kbps for TCP flows longer than 20 segments. Selecting  $P_{loss} = 0.01$ , we obtain a network-level design constraint equal to  $RTT \leq 0.15$ s for all source-destination pairs. Each traffic relation offers an average aggregate traffic equal to  $\gamma = 2$  Mbps. Link lengths vary between 25 Km and 760 Km.

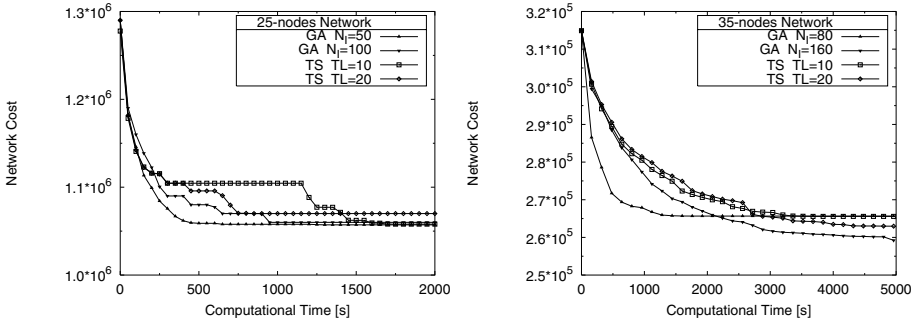
In left plot of Fig. 3 we show the network cost versus computational time, in seconds, considering both the GA and TS algorithms (for different values of population and tabu list sizes). In order to improve further the performance of the TS algorithm we used a diversification criterion which corresponds to restart the procedure, with a new random initial solution. As we can see, in this case, the final solutions differ by at most 2%; but the GA can reach better solutions quicker than the TS. At each iteration the TS evaluates  $N_N = 140$  solutions (recall that a move corresponds to add or remove a link) and the GA evaluates “only”  $N_I = 50$  (or 100) solutions. Obviously, the complexity of each evaluation task (that is considered dominant regarding the other algorithm tasks) is the same for both GA and TS. Thus, in the studied cases, the GA will be faster than TS. As expected, for  $N_I = 50$  individuals the GA converges faster than runs with  $N_I = 100$  individuals, being the solution (after 30 minutes) almost the same.

In the second example, we set up a 15-node VPN network over a given 35-node, 140-link physical topology. The target QoS constraints for all traffic pairs are the same for the first case; and the average aggregate traffic is equal to  $\gamma = 3$  Mbps. Link lengths vary between 15 Km and 300 Km (average = 225 Km). In right plot of Fig. 3 we report the network cost versus computational time. We notice that, after a period of 1.38 hours, the solution values differ by at most 2.5%. The best solution is given by the GA with  $N_I = 160$  individuals (in contrast, the same solution value was reached by the TS, with  $TL = 20$  moves, after a period of 12 hours).

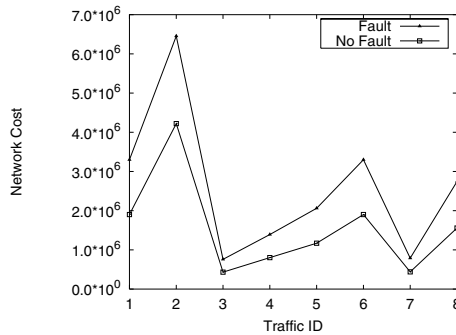
The GA with a small population quickly stagnates, and its solution value is “relatively poor”. Using a population size near the value suggested by the estimate ( $N_I = 160$ ) the GA make slower progress but consistently discover better solutions. If the population size increases further, the GA becomes more slowly, without however, being able to improve the solutions (not show in the figure).

Coming back to the 10-node VPN network case, we also compare the network cost to those of a dimensioning where the 2-connectivity constraint is relaxed, i.e., in this case, the failure of a node can interrupt the information exchange for some origin/destination pairs of the network. Fig. 4 presents the network costs, from the GA solutions, for different values of the average traffic. The plots show that 2-connectivity networks have an extra cost of about 75%.

In addition, by considering physical topologies with different number of links, we can analyze the behavior of network costs and computational times. In this case we



**Fig. 3.** Network Cost versus GA and TS Computational Time: 25–node network (left plot) and 35–node network (right plot).



**Fig. 4.** Network Costs for Different Traffics Scenarios (10–node VPN network).

considered initially the case of 150 links, and further increased the number of links randomly (until an all-connected case). Initial solutions to the GA are obtained by considering 100 random topologies for each case (number of links).

## 5 Conclusion

In this paper, we have considered the QoS and reliability design of packet networks, and in particular the joint Topology, Capacity and Flow Assignment problem where the link placements, routing assignments and capacities are considered to be decisions variables. By explicitly considering TCP traffic, we also need to consider the impact of finite buffers, therefore facing the Buffer Assignment problem.

Our solution approach is based on the exploration of the solution space using metaheuristic algorithms (GA and TS), where candidate solutions are evaluated by solving CFA problems. In order to obtain a practical, useful TCFA solver, it is fundamental a fast CFA problem solver. For this scope, we have proposed a new CFA algorithm, called GWFD, that is capable of assign flow and capacities under e2e QoS constraints.

Examples of application of the proposed design methodology to different networking configurations have been discussed; also experiments were performed in order to

evaluate the effectiveness of the metaheuristic approaches for solving the TCFA problem. Computational results suggest a better efficiency of the GA approach in providing good solutions for medium-sized computer networks, in comparison with well tried conventional TS methods.

The network target performances are validated against detailed simulation experiments, proving that the proposed design approach is both accurate and flexible.

## References

1. K. Claffy, Greg Miller, and Kevin Thompson, "The nature of the beast: Recent traffic measurements from an Internet backbone". In *Proceedings of INET '98*, July 1998.
2. M. Mellia, A. Carpani, R. Lo Cigno, "Measuring IP and TCP behavior on Edge Nodes", *IEEE Globecom 2002*, Taipei, TW, Nov. 2002.
3. E. Wille, *Design and Planning of IP Networks Under End-to-End QoS Constraints*, PhD dissertation, Politecnico di Torino, Available at [http://www.tlc-networks.polito.it/mellia/papers/wille\\_phd.pdf](http://www.tlc-networks.polito.it/mellia/papers/wille_phd.pdf)
4. E.Wille, M.Garetto, M.Mellia, E.Leonardi, M.Ajmone Marsan, "Considering End-to-End QoS in IP Network Design", *NETWORKS 2004*, Vienna, June 13-16
5. V.Paxson, S.Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, Vol.3, N.3, pp.226-244, Jun. 1995.
6. B. Gavish and I. Neuman. "A system for routing and capacity assignment in computer communication networks". *IEEE Transactions on Communications*, 37(4):360-366, April 1989.
7. K. Kamimura and H. Nishino. "An efficient method for determining economical configurations of elementary packet-switched networks". *IEEE Transactions on Communications*, 39(2), Feb. 1991, pp. 278-288.
8. K. T. Cheng and F. Y. S. Lin, "Minimax End-to-End Delay Routing and Capacity Assignment for Virtual Circuit Networks", *Proc. IEEE Globecom*, pp. 2134-2138, 1995.
9. M.Garetto, D.Towsley, "Modeling, Simulation and Measurements of Queuing Delay under Long-tail Internet Traffic", *ACM SIGMETRICS 2003*, San Diego, CA, June 2003.
10. D. E. Goldberg, *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison Wesley Publishing Company, 1989.
11. Fred Glover and Manuel Laguna, *Tabu Search*, Kluwer Academic Publishers, 1997.
12. B. Gavish. "Topological design of computer communication networks – the overall design problem". *European Journal of Operational Research*, 58, (1992) 149-172.
13. T. T. Mai Hoang and W. Zorn, "Genetic Algorithms for Capacity Planning of IP-Based Networks", *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, 1309-1315, 2001.
14. Gerla, M., L. Kleinrock, "On the Topological Design of Distributed Computer Networks", *IEEE Transactions on Communications*, Vol.25, pp.48-60, Jan. 1977.
15. Wright, M.; "Interior methods for constrained optimization", *Acta Numerica*, Vol.1, pp. 341-407, 1992.
16. A.Medina, A.Lakhina, I.Matta, J.Byers, "BRITE: Boston university representative internet topology generator", Boston University, <http://cswww.bu.edu/brite>, April 2001.