Routing with End to End QoS Guarantees in Broadband Networks

Ariel Orda
Department of Electrical Engineering, Technion — Israel Institute of Technology

Abstract— **We consider routing schemes for connections with end-to-end delay requirements, and investigate several fundamental problems. First, we focus on networks which employ rate-based schedulers and hence map delay guarantees into nodal rate guarantees, as done with the Guaranteed Service class proposed for the Internet. We consider first the basic problem of identifying a feasible route for the connection, for which a straightforward, yet computationally costly solution exists. Accordingly, we establish several approximation schemes that offer substantially lower computational complexity. We then consider the more general problem of optimizing the route choice in terms of balancing loads and accommodating multiple connections, for which we formulate and validate several optimal algorithms. We discuss the implementation of such schemes in the context of link-state and distance-vector protocols.**

Next, we consider the fundamental problem of constrained path optimization. This problem, typical of QoS routing, is NP-hard. While standard approximation methods exist, their complexity may often be prohibitive in terms of scalability. Such approximations do not make use of the particular properties of large-scale networks, such as the fact that the path selection process is typically presented with a hierarchical, aggregated topology. By exploiting the structure of such topologies, we obtain an ϵ-optimal algorithm for the constrained shortest path problem, which offers a substantial improvement in terms of scalability.

Keywords— **Keywords: QoS Routing, Rate-Based Schedulers, Constrained Path Optimization, Topology Aggregation, Hierarchical Networks.**

I. Introduction

Broadband integrated services networks are expected to support multiple and diverse applications, with various quality of service (QoS) requirements. Accordingly, a key issue in the design of broadband architectures is how to provide the resources in order to meet the requirements of each connection, and, moreover, how to meet that goal in a networkwide efficient manner. The establishment of efficient QoS routing schemes is, undoubtedly, one of the major building blocks in such architectures.

Indeed, QoS routing has been the subject of several studies and proposals (see, e.g., [1], [4], [5], [8], [9], [14], [15], [18], [20] and references therein). It has been recognized that the establishment of an efficient QoS routing scheme poses several complex challenges.

One of the major problems in the establishment of a connection with QoS guarantees arises from the need to map some end-to-end requirements, mainly delay, into local (nodal) requirements, which would indicate how to reserve resources along the route. Obviously, the identification of a feasible, furthermore "optimal" route for such a connection, greatly depends on our ability to perform such a mapping. Conversely, the efficiency of such a mapping scheme depends on our ability to transform it into a path identification algorithm, which should be computationally manageable and would allow to establish connections that are both feasible and networkwide-efficient.

More precisely, the ability to provide end-to-end guarantees, such as delay, depends to a large extent on the scheduling policy and service discipline employed in the nodes. Such disciplines are characterized by bounds on the maximal delay that any node can incur, and hence a corresponding bound on the end-to-end delay can be derived. Such a bound provides a valuable tool for quantifying the quality of a path in terms of its ability to meet the QoS delay requirement. The corresponding routing problem is, therefore, to identify the path that has the best performance, according to that bound and with respect to the QoS requirement. Typical schedulers map delay guarantees into rate guarantees, and have nodes advertise the residual rate they have available (e.g., [7], [17], [21]). In particular, the Guaranteed Service class proposed for the Internet [19] is based on such "rate-based" principles. Consequently, several recent studies [8], [9], [15], [18] aimed at taking advantage of the properties of such schedulers when computing paths that satisfy end-to-end delay bounds. In particular, it was shown in [8] that for a given connection with end-to-end delay constraint, the existence and identity of a feasible path can be obtained through up to M executions of a standard shortest path algorithm, where M is the number of network links. In [15], [18] it was shown that, through this scheme, one can accommodate additional connection requirements such as jitter.

While the above findings are encouraging, they are still unsatisfactory and call for additional work. For example, [8] addressed the need to cope with the inherent imprecision of dynamic parameters, in particular link residual rates. Yet even the basic solution of [8], [9], [15], [18] for the standard (precise) setting may be computationally prohibitive, as it involves the execution of up to M shortest-path algorithms. Efficient solutions of lower complexity are thus called for. Another important issue is how to conduct the path search in a way that would guarantee not only its feasibility with respect to a particular connection, but also its efficiency, or optimality, with respect to the overall management of network resources. These two issues, namely reducing the complexity and optimizing the path choice, are the subject of the first part of this paper.

Another major problem in QoS routing is the need to conduct multi-objective, constrained path optimization. Indeed, a typical setting is to identify a path that minimizes some target function, e.g. administrative costs, while observing one or more constraints, such as end-to-end de-

email: ariel@ee.technion.ac.il

lay. It is well known that such problems are, in general, NP-hard [6]. Some general approximation schemes that are ϵ-optimal have been proposed (see [11] and references therein). However, while being of polynomial complexity, these schemes may be computationally prohibitive. For example, the "strongly polynomial" solution of [11] has a complexity of $O(M \cdot N^2 \cdot \frac{1}{\epsilon} \cdot \log(\frac{N}{\epsilon}))$, where N, M, and ϵ are, respectively, the number of nodes, number of links and accuracy of the solution; this is at least N^2 worse than a standard shortest-path algorithm, meaning that its scalability is considerably poorer. On the other hand, exploiting the particular structure of broadband network topologies should facilitate the establishment of more efficient solutions. Accordingly, in the second part of the paper we look at some recent proposals for hierarchical topology aggregation, and establish a new algorithm for constrained path optimization. This algorithm also has ϵ-accuracy yet it maintains the scalability of standard shortest path algorithms, *i.e.*, its complexity bears the same dependence on network size.

The rest of the paper is structured as follows. In Section II we consider routing schemes for rate-based schedulers. After formulating the model in Subsection II-A, we consider algorithms for finding feasible paths, in Subsection II-B. Here, we revise the standard approach, and then present several approximation methods, which considerably decrease the complexity of the solution. Next, in Subsection II-C, we present and validate several path optimization schemes, which identify feasible paths in a networkwide-efficient fashion. Based on these findings, in Subsection II-D we discuss the implementation of distributed protocols for rate-based QoS routing. Next, in Section III, we consider the problem of constrained path optimization in hierarchical networks. After formulating the model and problem in Subsection III-A, we present, in Subsection III-B, a new approximation algorithm for constrained path optimization, which exploits the hierarchical structure and provides better scalability than standard approaches. Finally, conclusions appear in Section IV. Due to space limits, proofs are omitted from this version, and can be found in [16].

II. QoS Routing with Rate-Based Schedulers

A. Model Formulation

Given is a network $G(V, E)$, where V is the set of nodes and E is the set of links, and let $N = |V|$ and $M = |E|$. We denote by H the maximal possible number of hops in a path.[1]

Each link $l \in E$ is characterized by the following values:

- A *maximal rate* (capacity) R_l, which the link can offer to a new connection. When a new connection with a rate $r < R_l$ is established through link l, the value of R_l becomes $R_l - r$.
- A *constant delay* value d_l, related to the link's speed, propagation delay and maximal transfer unit.

A connection i in the network is characterized by the following values:

- A *source node* s^i and a *destination node* t^i.
- A *bias* value σ^i, related to the connection's maximal burst.
- A *maximal packet size* c^i.
- A *maximal end-to-end delay constraint* D^i.
- A *bandwidth requirement* b^i.

A connection should be routed through some path $\mathbf{p}$ between the corresponding source and destination nodes. We shall denote by $n(\mathbf{p})$ the number of hops (*i.e.*, links) of a path $\mathbf{p}$, and by $r(\mathbf{p})$ its maximal available rate, *i.e.*, $r(\mathbf{p}) = \min_{l \in \mathbf{p}} R_l$. We assume that the scheduling policy in the network belongs to the "rate-based" class [7], [17], [21], as assumed in the specification of the Guaranteed Service for IP [19]. Accordingly, when a connection i is routed over a path $\mathbf{p}$ with a reserved rate r (where $r \le r(\mathbf{p})$), the following upper bound $D^i(\mathbf{p}, r)$ on the end-to-end delay applies[2]:

$$D^i(\mathbf{p}, r) = \frac{\sigma^i + n(\mathbf{p}) \cdot c^i}{r} + \sum_{l \in \mathbf{p}} d_l. \qquad (1)$$

We shall also denote by $D^i(\mathbf{p})$ the minimal possible value of $D^i(\mathbf{p}, r)$, *i.e.*, $D^i(\mathbf{p}) = D^i(\mathbf{p}, r(\mathbf{p}))$; $D^i(\mathbf{p})$ shall be referred to as *the guaranteed delay* of $\mathbf{p}$. Accordingly, a path $\mathbf{p}$ between s^i and t^i is said to be *feasible* for connection i if $D^i(\mathbf{p}) \le D^i$ and $r(\mathbf{p}) \ge b^i$; we denote by $\mathcal{P}^i$ the set of all feasible paths for that connection. A connection i is feasible if it has a feasible path, *i.e.*, $\mathcal{P}^i \ne \emptyset$. Finally, a path with minimal guaranteed delay shall be said to be *fastest.*[3]

B. Finding Feasible Paths

We begin with the basic problem of identifying feasible paths. Consider first the following multi-commodity problem, where multiple connections should be established.

Multi-commodity Feasible Path Problem (MFP): *Given are a network $G(V, E)$, with a maximal rate R_l and a delay d_l for each $l \in E$, and connections i, $1 \le i \le I$, each with source s^i, destination t^i, bias σ^i, packet size c^i, delay constraint D^i and bandwidth requirement b^i. For each connection, find a feasible path $\mathbf{p}^i$,* i.e. *a path between s^i and t^i and a rate r^i, such that:*

1. For all i: $\frac{\sigma^i + n(\mathbf{p}^i) \cdot c^i}{r^i} + \sum_{l \in \mathbf{p}^i} d_l \le D^i$.
2. For all l: $\sum_{\{i | l \in \mathbf{p}^i\}} r^i \le R_l$.
3. For all i: $r^i \ge b^i$.

Proposition 1: Problem MFP is NP-hard. □

Therefore, we first restrict our attention to the single-commodity problem. Maintaining multiple connections shall be addressed later on, from a dynamic perspective and in the more general context of path optimization.

[1] Obviously, $H \le N - 1$. Moreover, the actual value of H is usually much smaller, both due to the fact that, in typical network topologies, the diameter is considerably smaller than N, and also since the value of H is often restricted *a-priori* by the routing protocol.

[2] We note that our findings can easily accommodate a somewhat tighter bound, where $n(\mathbf{p})$ is replaced by $n(\mathbf{p}) - 1$.

[3] For convenience, the definition includes the case where the connection is infeasible.

Single-commodity feasible path problem (SFP): *An MFP instance with $I = 1$.*

The following algorithm is a straightforward solution to SFP, which is both exact and polynomial.[4]

Algorithm SFP (sketch):

Let $R^1 < R^2 < \ldots < R^K$ be the different values of R_l, where $K \leq M$. For each k, $1 \leq k \leq K$, perform the following: delete all links l for which $R_l < R^k$; on the remaining graph, execute a Bellman-Ford shortest path algorithm, with respect to the metric $\{d_l\}$, in order to identify a shortest path for each possible number of hops n, for all $1 \leq n \leq H$. This way, H paths are identified at each iteration, and $K \cdot H$ paths overall. The solution to SFP is found by computing the respective guaranteed delay of each path, and choosing the best.

The computational complexity of Algorithm SFP is $O(M \cdot H \cdot K)$.[5] While being polynomial, such complexity could still be prohibitive, since K would often be of the same order as M, due to the dynamic nature of link rates. Note that the high complexity stems from our need to simultaneously deal with three metrics, namely link delays, link rates and number of hops. Therefore, in order to reach an efficient yet computationally tractable solution, we elaborate on two approximation schemes, based on quantization. In the first scheme, quantization is applied to the maximal link rates, whereas in the second it applies to the considered number of hops.

B.1 Rate Quantization

For ease of presentation, we begin with an approximation of order 2, *i.e.*, the following algorithm shall derive a solution (path) whose guaranteed delay is at most twice greater than that of the fastest.

Let the maximal rate values in the network be $R^1, R^2, \ldots, R^K$, where $K \leq M$. Denote $\widehat{R} = \frac{R^K}{R^1}$. Without loss of generality, we assume that links for which $R_l < b^i$ have been deleted, thus $R^1 \geq b^i$, *i.e.* the bandwidth requirement can be neglected; this assumption shall be made throughout the rest of the paper. We group these values into $O(\log \widehat{R})$ *rate-classes*, such that, for $0 \leq j \leq \lfloor \log_2 \widehat{R} \rfloor$, rate-class j contains all maximal rates in the range $R^1 \cdot 2^j \ldots \min(R^1 \cdot 2^{j+1} - 1, R^K)$. We say that a link l is in rate-class j if $R^1 \cdot 2^j \leq R_l < R^1 \cdot 2^{j+1}$. The Rate-Quantized 2-Approximation algorithm is as follows.

Algorithm SFP-RQ-2:

1. For all j, $0 \leq j \leq \lfloor \log_2 \widehat{R} \rfloor$:

(a) If rate-class j is empty, skip to the next value of j.

(b) Delete from the network all links whose rate-class is less than j.

(c) Using a Bellman-Ford shortest path algorithm, find, for all n, $0 \leq n \leq H$, a path $\mathbf{p}_n(j)$, which is shortest with respect to the metric $\{d_l\}$ among n-hop paths.

2. Among the $O(H \cdot \log_2 \widehat{R})$ paths $\mathbf{p}_n(j)$, choose a path $\tilde{\mathbf{p}}$ with the smallest guaranteed delay.

The following result is immediate:

Lemma 1: The complexity of algorithm SFP-RQ-2 is $O(H \cdot M \cdot \min(\log \widehat{R}, K))$. □

We observe that, typically, the value of $\widehat{R}$ can be expected to be much smaller than 2^K, thus algorithm SFP-RQ-2 offers a substantially smaller complexity than algorithm SFP.

Proposition 2: The guaranteed delay of the output $\tilde{\mathbf{p}}$ of algorithm SFP-RQ-2 is at most twice larger than the minimal value, *i.e.*, if $\mathbf{p}^*$ is a fastest path then

$$D(\tilde{\mathbf{p}}) \leq 2 \cdot D(\mathbf{p}^*). \tag{2}$$

□

We proceed to generalize the above algorithm for obtaining an ϵ-approximation. Given a value $\epsilon > 0$, the maximal rate values are grouped into $O(\log_{1+\epsilon} \widehat{R})$ rate-classes, such that, for $0 \leq j \leq \log_{1+\epsilon} \widehat{R} - 1$, rate-class j contains all maximal rates in the range $R^1 \cdot (1+\epsilon)^j \ldots R^1 \cdot (1+\epsilon)^{j+1} - 1$. We say that a link l is in rate-class j if the value of R_l is in that rate-class. The corresponding algorithm SFP-RQ-ϵ is identical to algorithm SFP-RQ-2, only that it consists of $\min(K, \log_{1+\epsilon} \widehat{R})$ iterations and it considers the above $O(\log_{1+\epsilon} \widehat{R})$ rate-classes.

Lemma 2: The complexity of algorithm SFP-RQ-ϵ is $O(H \cdot M \cdot \min(\frac{1}{\epsilon} \cdot \log \widehat{R}, K))$. □

Proposition 3: The guaranteed delay of the output $\tilde{\mathbf{p}}$ of algorithm SFP-RQ-ϵ is at most $1+\epsilon$ times larger than the minimal value, *i.e.*, if $\mathbf{p}^*$ is a fastest path then

$$D(\tilde{\mathbf{p}}) \leq (1+\epsilon) \cdot D(\mathbf{p}^*). \tag{3}$$

□

We observe that algorithm SFP-RQ-ϵ enables to find a suitable tradeoff between precision and complexity.

B.2 Hop Quantization

In this subsection we focus our attention on approximation schemes that reduce the complexity by narrowing (*i.e.*, quantizing) the number of considered hops. As already mentioned, Problem SFP can be solved through $O(K)$ invocations of Dijkstra's shortest path algorithm, hence avoiding the need to iterate on hop values. However, as shall become evident in the next subsection, in many cases one needs to consider a solution *for each possible* hop value. Therefore we shall present the hop quantization scheme through the following generalization of Problem SFP.

Hop-Constrained SFP (HSFP): *For* each *hop value h, $1 \leq h \leq H$, find a path* of at most h hops *that solves Problem SFP.*

An exact solution to Problem HSFP is provided by Algorithm SFP, with a complexity of $O(K \cdot M \cdot H)$. We seek an algorithm that provides an approximate solution, within a lower complexity.

Before proceeding, it is important to understand what a "hop" stands for, in terms of the delay bound (1).

[4] As previously mentioned, a similar method has already been presented in [8] ("deterministic case") and subsequently in [15], [18].

[5] We note that a lower complexity bound, of $O(K \cdot (N \log N + M))$, could be achieved with Dijkstra's algorithm. We chose Bellman-Ford's as our building block here, since it is essential for several extensions presented in the following. Relatedly, the Bellman-Ford algorithm has the additional advantage of identifying a *minimum-hop* feasible path. Moreover, as observed in [9], in many typical networking scenarios, it is the case that $O(MH) < O(N \log N + M)$.

Throughout the paper, we assume that a "hop" is a single network link. However, when the network represents an aggregated topology[6], a "hop" accounts for the actual number of (real) links that are represented by a single element in the aggregated topology. In such a ("generalized-hop") case, the maximal hop value H can be significantly larger than the number of "logical" nodes N. By considering the hop value as a *constraint* and performing path optimization with respect to the metric $\{d_l\}$, one can apply approximation schemes for the Constrained Shortest Path problem (e.g., [11]), and obtain an efficient ϵ-optimal algorithm for Problem HSFP.

We turn back to the standard case, in which a "hop" represents a single link, hence $H < N$. For this case, we present the following scheme, which provides a constant-factor approximation while decreasing the complexity. The basic idea is the following. Given the metric $\{d_l\}$, compose a new metric $\{\widehat{d_l}\}$ by adding a value κ to each link. By increasing the value of κ, we decrease the number of hops on a path that is shortest with respect to $\{\widehat{d_l}\}$. Therefore, for any hop value h, we can obtain a path that is shortest with respect to $\{\widehat{d_l}\}$, whose number of hops is below h. On the other hand, by making κ increase "just as needed", one obtains a solution that is efficient also with respect to $\{d_l\}$. The above discussion is formally specified and validated in the following.

Let η be a positive number, and let $\widehat{H} = \lceil \log_{1+\eta} H \rceil$. We group the hop values $1, 2, \ldots, H$ into $O(\log_{1+\eta} H)$ *hop-classes*, such that, for $0 \leq j \leq \widehat{H}$, hop-class j contains all hops in the range $1, \ldots, \min((1+\eta)^j, H)$ (note that each hop class contains all previous hop classes). Accordingly, we say that a path $\mathbf{p}$ belongs to hop-class j if $n(\mathbf{p}) \leq (1+\eta)^j$.

Algorithm HSFP:
1. Delete from G all links l for which $d_l \geq D^i$.
2. For all k, $1 \leq k \leq K$:
 (a) Delete all links l for which $R_l < R^k$, obtaining the residual network G^k.
 (b) Execute Algorithm HQ(G^k, s^i,t^i), below, thus obtaining, for each $0 \leq j \leq \widehat{H}$, a path $\mathbf{p}_j^k$ between the source s^i and the destination t^i that belongs to hop-class j.
3. For each hop class j, $0 \leq j \leq \widehat{H}$, among the $O(K)$ paths $\mathbf{p}_j^k$, choose a path $\tilde{\mathbf{p}}_j$ with the smallest guaranteed delay.

Algorithm HQ(G^k, s, t):
1. Initialization:
 (a) Compute $H_{\min}$, the minimum number of hops between s and t on G^k.
 (b) Compute $j_{\min}$, the smallest hop class to which $H_{\min}$ belongs.
 (c) $j \leftarrow \widehat{H}$, $\kappa \leftarrow -1$.
2. While $j \geq j_{\min}$ do:
 (a) If $\kappa \leq 0$ then $\kappa \leftarrow \kappa + 1$ else $\kappa \leftarrow (1+\eta) \cdot \kappa$. [7]
 (b) For all links l in G^k: $\widehat{d_l} \leftarrow \kappa + d_l$.
 (c) Find a path $\mathbf{p}_j^k(\kappa)$ in G^k between s and t that is shortest with respect to $\{\widehat{d_l}\}$.
 (d) If $n(\mathbf{p}_j^k(\kappa)) > (1+\eta)^j$ then go to step 2(a).
 (e) Compute j', the smallest hop class to which $\mathbf{p}_j^k(\kappa)$ belongs.[8]
 (f) For all $j' \leq m \leq j$: $\mathbf{p}_m^k \leftarrow \mathbf{p}_j^k(\kappa)$.
 (g) $j \leftarrow j' - 1$.

Proposition 4:
1. The complexity of Algorithm HQ is

$$O((N \log N + M) \cdot \log_{1+\eta}(H_{\min} D^i)). \tag{4}$$

2. For $1 \leq k \leq K$ and $1 \leq h \leq H$, let $\mathbf{p}^{*k}(h)$ be a shortest path in G^k, with respect to the metric $\{d_l\}$, among paths with at most h hops. Denote by j_h the smallest hop class which includes the value of h hops. Then:

$$n(\mathbf{p}_{j_h}^k) \leq (1+\eta) \cdot h, \tag{5}$$

and

$$\sum_{l \in \mathbf{p}_{j_h}^k} d_l \leq (2 + \frac{1}{\eta}) \sum_{l \in \mathbf{p}^{*k}(h)} d_l. \tag{6}$$

In other words, for any number of hops h, Algorithm HQ identifies a path that approximates the shortest h-hop constrained solution within a factor of $1+\eta$ with respect to the number of hops and a factor of $2 + \frac{1}{\eta}$ with respect to the distance metric. □

Proposition 5:
1. The complexity of Algorithm HSFP is $O(K \cdot (N \log N + M) \cdot \log_{1+\eta}(H_{\min} D^i))$.
2. For each hop value h, $1 \leq h \leq H$, let j_h be the smallest hop class that includes h. Then, the guaranteed delay of the output $\tilde{\mathbf{p}}_j$ of Algorithm HSFP is at most $\max(1+\eta, 2+\frac{1}{\eta})$ times larger than the optimal value for h hops, and its number of hops is at most $(1+\eta) \cdot h$. □

For example, for $\eta = 1$, Algorithm HSFP provides a solution with approximation factors of 2 with respect to the number of hops and 3 with respect to the delay; for $\eta = \frac{1+\sqrt{5}}{2}$, both approximation factors are equal to ≈ 2.6.

Remark: The method employed in Algorithm HQ somewhat resembles the Lagrangian relaxation method, which is employed in some approximation schemes for constrained network optimization problems (see, e.g., [2]). However, a straightforward adaptation of that method to our framework results in an approximation that is less efficient in terms of either the computation complexity or the accuracy with respect to the number of hops.

Having demonstrated how to apply approximations through either rate or hop quantization, in the rest of the paper we shall focus only on exact solutions.

C. Optimizing the Path Selection

The ability to identify a feasible path for a connection does not yield yet a satisfactory QoS routing solution. In order to supervise multiple connections across the network, the routing algorithm must carefully select an "optimal" path among the feasible ones. There does not seem to be a precise definition for the "optimality" of a path in this

[6] E.g., as with the PNNI aggregation scheme[1], discussed in the next section.

[7] This way, κ is assigned the initial values of 0 and 1.

[8] Note that, due to step 2(d), $j' \leq j$.

context, and various path selection criteria and techniques have been proposed and investigated, .e.g., load balancing [3], [9], VC packing [10] and load profiling [4]. We focus, in Subsection II-C.1, on criteria that optimize the consumption of rates; this choice was driven by the observation that, with rate-based schedulers, the amount of rate allocated to a connection is not *fixed a priori* but rather *depends on the path selection*. However, we believe that our results serve to indicate how additional criteria, such as those mentioned above, can be accommodated within the context of rate-based schedulers. In Subsection II-C.2 we illustrate this for load balancing.

C.1 Rate Consumption Criteria

Consider first the simple criterion of choosing, for a connection request i, a path $\mathbf{p}$ for which the consumed rate is minimal. For a feasible path $\mathbf{p}$, the consumed rate r should be such that $D^i(\mathbf{p}, r) \leq D^i$, therefore its minimal possible value $r^i_{\mathbf{p}}$ is:

$$r^i_{\mathbf{p}} = \frac{\sigma^i + n(\mathbf{p}) \cdot c^i}{D^i - \sum_{j \in \mathbf{p}} d_j}.$$

Alternatively, one may prefer to minimize the *overall* reduction of rates imposed by the new connection. In that case, an optimal path is one that minimizes the sum of the consumed rates over all links, *i.e.*, $\sum_{j \in \mathbf{p}} r^i_{\mathbf{p}}$.

In fact, the above criteria are special cases of the following, general optimization scheme. Suppose we seek a path and a consumed rate value with minimal "cost" among the path-rate pairs that satisfy the delay constraint, and assume that the cost of a path-rate pair is some nondecreasing function of (i) the number of hops, (ii) the rate and (iii) the corresponding end-to-end delay bound. Formally, to a connection i, path $\mathbf{p}$ and rate r, there corresponds a cost $C(r, \mathbf{p}) = C(r, n(\mathbf{p}), D^i(\mathbf{p}, r))$, which is non-decreasing with respect to each of its three arguments. For example, $C(r, n(\mathbf{p}), D^i(\mathbf{p}, r)) = r$ corresponds to the rate minimization criterion considered above, whereas $C(r, n(\mathbf{p}), D^i(\mathbf{p}, r)) = r \cdot n(\mathbf{p})$ corresponds to the overall rate minimization criterion. As another example, choosing $C(r, n(\mathbf{p}), D^i(\mathbf{p}, r)) = \alpha \cdot r \cdot n(\mathbf{p}) + D^i(\mathbf{p}, r)$, for a large enough α, results in selecting a path of minimal delay bound, *among those that have minimal overall rate*; such a criterion may suit an environment where there are (small) uncertainties with respect to the network parameters, hence one prefers, if possible, to be a bit away from the delay bound.

Considering the above general cost structure, the following algorithm identifies an optimal path.

Algorithm Min. Cost (MC):

1. For $k = 1$ to K:
 (a) Delete links l with $R_l < R^k$.
 (b) For $1 \leq n \leq H$:
 i. Find a path $\mathbf{p}(n, k)$ that is shortest with respect to the metric $\{d_l\}$, among paths of at most n-hops.
 ii. If $D^i(\mathbf{p}(n, k), R^k) \leq D^i$ then:
 A. calculate the value r_0 for which $D^i(\mathbf{p}(n, k), r_0) = D^i$;
 B. $r(n, k) \leftarrow argmin_{r_0 \leq r \leq R^k} C(r, \mathbf{p}(n, k))$;
 C. $MC(n, k) \leftarrow C(r(n, k), \mathbf{p}(n, k))$.
 iii. Else $MC(n, k) \leftarrow \infty$.
2. Let $\tilde{n}$ and $\tilde{k}$ be such that $MC(\tilde{n}, \tilde{k})$ is minimal over all $1 \leq n \leq H$ and $1 \leq k \leq K$. If $MC(\tilde{n}, \tilde{k}) = \infty$ then the connection is not feasible, otherwise $\mathbf{p}(\tilde{n}, \tilde{k})$ and $r(\tilde{n}, \tilde{k})$ are, respectively, the required path and rate.

Proposition 6: Algorithm MC correctly identifies the feasibility of the connection. Whenever the connection is feasible, the path $\mathbf{p}(\tilde{n}, \tilde{k})$ and rate $r(\tilde{n}, \tilde{k})$, identified by the algorithm, achieve the minimal cost among all feasible solutions. The algorithm's complexity is $O(K \cdot H \cdot M)$. □

C.2 Load Balancing Criteria

The way that the path rate affects the available rate of its links depends on the current rate value at each link. Therefore, a better measure for balancing the loads over the network may be one that accounts for the *relative*, rather than absolute, impact of the new connection on each link. For example, we could seek a feasible path $\mathbf{p}$ that minimizes the value of $\max_{l \in \mathbf{p}} \frac{r^i_{\mathbf{p}}}{R_l}$. Since

$$\max_{l \in \mathbf{p}} \frac{r^i_{\mathbf{p}}}{R_l} = \max_{l \in \mathbf{p}} \frac{1}{R_l} \cdot \frac{\sigma^i + n(\mathbf{p}) \cdot c^i}{D^i - \sum_j d_j},$$

the problem can be stated as finding a feasible path $\mathbf{p}$ that *maximizes*

$$\min_{l \in \mathbf{p}} R_l \cdot \frac{D^i - \sum_{j \in \mathbf{p}} d_j}{\sigma^i + n(\mathbf{p}) \cdot c^i}. \tag{7}$$

Note that the optimization criterion (7) cannot be accommodated as a special case of the cost function discussed in the previous subsection. The following algorithm solves the problem, *i.e.*, identifies a path that maximizes (7). As before, we denote by $R^1 < R^2 < \ldots R^K$ the different values of link maximal rates.

Algorithm Min. Relative Rate (MRR):

1. For $k = 1$ to K:
 (a) Delete links l with $R_l < R^k$.
 (b) For $1 \leq n \leq H$, find a path $\mathbf{p}(n, k)$ that is shortest with respect to the metric $\{d_l\}$, among n-hop paths.
 (c) $MRR(n, k) \leftarrow R^k \cdot \frac{D^i - \sum_{j \in \mathbf{p}(n,k)} d_j}{\sigma^i + n \cdot c^i}$.
2. Let $\tilde{n}$ and $\tilde{k}$ be such that $MRR(\tilde{n}, \tilde{k})$ is maximal over all $1 \leq n \leq H$ and $1 \leq k \leq K$. If $MRR(\tilde{n}, \tilde{k}) < 1$ then the connection is not feasible, otherwise $\mathbf{p}(\tilde{n}, \tilde{k})$ is the required path.

Proposition 7: Algorithm MRR correctly identifies the feasibility of the connection. Whenever the connection is feasible, the path $\tilde{\mathbf{p}}$, identified by Algorithm MRR, maximizes expression (7). The algorithm's complexity is $O(K \cdot H \cdot M)$. □

Alternatively, we could aim at balancing loads by seeking a path for which the *residual* maximal rate (*i.e.*, after establishing the new connection) of its *bottleneck link* is maximal. Indeed, this is yet another way to drive further loads away from congested areas.

More precisely, given a connection i and a feasible path $\mathbf{p}$ between its source and destination nodes, denote by $B^i_{\mathbf{p}}$ the (largest possible value of the) residual maximal rate of its bottleneck link, *i.e.*,

$$B^i_{\mathbf{p}} = \min_{l\in\mathbf{p}}(R_l - r^i_{\mathbf{p}}) = \min_{l\in\mathbf{p}}(R_l - \frac{\sigma^i + n(\mathbf{p})\cdot c^i}{D^i - \sum_{j\in\mathbf{p}} d_j}). \quad (8)$$

The problem, then, is to find a feasible path $\mathbf{p}$ that maximizes (8).

As previously, let the maximal rate values in the network be $R^1 < R^2 < \ldots R^K$, $K \leq M$. The following algorithm solves this problem.

Algorithm Max. Residual Bottleneck (MRB):
1. For $k = 1$ to K:
 (a) Delete all links l for which $R_l < R^k$.
 (b) For all n, $1 \leq n \leq H$:
 i. For $1 \leq n \leq H$, find a path $\mathbf{p}(n,k)$ that is shortest with respect to the metric $\{d_l\}$, among n-hop paths.
 ii. If $\sum_{j\in\mathbf{p}(n,k)} d_j < D^i$ then $B(n,k) \leftarrow R^k - r^i_{\mathbf{p}(n,k)}$;
 iii. else $B(n,k) \leftarrow -\infty$.
2. Let $\tilde{n}$ and $\tilde{k}$ be such that $B(\tilde{n},\tilde{k})$ is maximal over all $1 \leq n \leq H$ and $1 \leq k \leq K$. If $B(\tilde{n},\tilde{k}) < 0$ then the connection is not feasible, otherwise $\tilde{\mathbf{p}} = \mathbf{p}(\tilde{n},\tilde{k})$ is the required path.

Proposition 8: Algorithm MRB correctly identifies the feasibility of the connection. Whenever the connection is feasible, the path $\tilde{\mathbf{p}}$, identified by Algorithm MRB, maximizes expression (8). The algorithm's complexity is $O(K \cdot H \cdot M)$. □

D. Distributed Protocols

In practice, the path selection process relies on some distributed protocol. Focusing on the basic problem of finding a path with the minimal guaranteed delay, we proceed to discuss the two typical schemes of *link-state* and *distance-vector* protocols [12].

Consider first a link-state protocol, where each node should be continuously updated with the current parameters of the various links. While the d_l values can be assumed to be constant over large portions of time, the R_l values change with each new connection, thus it is impractical to inform the source upon each and every change. The approximation method[9] presented in Section II-B.1, which is based on rate quantization, can be conveniently employed in the present context: a node informs the rest of the network, through a flooding protocol, on a change in the maximal rate of an outgoing link whenever the new value is at least twice larger/smaller than the previously reported value. Following the same reasoning used in the proof of Proposition 2, it is easy to see that a source node would select a path that is at most twice worse than a fastest one. Moreover, the values of maximal rates can be quantized (*a priori*, and networkwide) into some predetermined set of "thresholds" $R_{min}, R_{min}\cdot 2, R_{min}\cdot 4, \ldots, R_{min}\cdot\widehat{R}$, such that nodes would report on changes only when crossing a new threshold. This way, we also guarantee that a source node would have to consider at most $\log\widehat{R}$ maximal rate values, at any given time, thus obtaining the complexity of Algorithm SFP-2.

[9]For simplicity, we consider here a 2-approximation.

Consider now a distance-vector protocol. A straightforward approach would be to run an independent execution for each maximal rate value R_l, each time considering the metric $\{d_l\}$ on the respective sub-graph. The implementation, per destination node, would consist of the exchange of $O(K)$ distance vectors between adjacent nodes, and the concurrent execution of $O(K)$ shortest path algorithms. The approach is straightforward, hence the details are omitted.[10].

A simpler distance-vector approach is possible, again by applying the approximation method. Instead of considering all $O(K)$ maximal rate values, one can consider the $O(\log\widehat{R})$ static values $R_{min}, R_{min}\cdot 2, R_{min}\cdot 4, \ldots, R_{min}\cdot\widehat{R}$, where R_{min} is a known lower bound on the maximal rate values. This way one achieves an approximation of 2 by exchanging just $O(\log\widehat{R})$ vectors and concurrently running $O(\log\widehat{R})$ shortest path algorithms.

III. Constrained Path Optimization in Hierarchical Networks

A. Motivation and Formulation

We consider now double objective optimization, where one metric, say "delay", imposes a constraint on the path, and the other, say "weight" (e.g., administrative cost), should be minimized, while observing the constraint. Each link l has a delay d_l and a weight w_l. The delay and weight of a path $\mathbf{p}$ are, respectively, the sums of the delays and weights of its links, and are denoted as $D(\mathbf{p})$ and $W(\mathbf{p})$. We thus seek a path $\mathbf{p}^*$ such that (i) $D(\mathbf{p}^*) \leq D^i$, and (ii) for any other path $\mathbf{p}$, either $D(\mathbf{p}) > D^i$ or $W(\mathbf{p}) \geq W(\mathbf{p}^*)$.

As already mentioned, the above problem is known as the *Constrained Shortest Path* problem. It is an NP-complete problem[6], for which several approximations have been proposed. Typically, the complexity of the approximations depends on the (maximal) values of delays and weights. For example, in [11], an ϵ-optimal solution for acyclic graphs is presented, with complexity $O(\log\log(B)(NM\cdot\frac{1}{\epsilon} + \log\log(B)))$, where N and M are, respectively, the number of nodes and number of edges, and B is an upper bound on the quality (in terms of either weight or delay) of the solution. Avoiding such dependencies on maximal weight and delay values is clearly beneficial. Accordingly, [11] proposes an ϵ-optimal solution (again, for acyclic graphs) which is *strongly polynomial*, *i.e.*, it does not depend on the link metrics. The complexity of that approximation is $O(M\cdot N^2\cdot\frac{1}{\epsilon}\cdot\log(\frac{N}{\epsilon}))$. While being strongly polynomial, the complexity of this approach may still be prohibitive. As shall be shown, better approximations, in particular in terms of scalability, can be found, for general graphs, by exploiting the typical hierarchical structure of the topology, with which the path selection process is presented.

Here, it is important to note that, as networks grow in size and, moreover, as they interconnect, it becomes impossible to maintain at the various nodes accurate knowledge

[10]As with any distance-vector protocol, routing loops can be formed during transient epochs; they can be avoided using standard techniques, e.g., the path tracing scheme employed in BGP[12].

about all components. There have been numerous proposals on how to provide scalable solutions, in particular for routing purposes, to the distribution of state information such as link metrics and network topology. A standard approach is to introduce a hierarchical process that progressively "aggregates" state information as networks get more and more remote. This is, for example, the approach embodied in the ATM Forum PNNI standard [1], which we chose as a generic model to focus our discussion.

We proceed to outline the hierarchical aggregation process, according to the PNNI recommendation (the description is taken mainly from [8]). The network or set of networks across which connections need to be routed is represented by a graph $G(V,E)$, where V is the set of *layer 1* nodes, and E is the set of *layer 1* links interconnecting them. $G(V,E)$ will be referred to as *the actual network*. Layer 1 nodes are clustered to form layer 2 nodes, which are clustered into layer 3 nodes etc., up to the last, say L-th, layer. Nodes of the same layer i, that are clustered into the same $(i+1)$-layer node, are said to be *peers* that belong to the same *peer group* [1]. Layer-i nodes that form a layer-$(i+1)$ node are called its *children*, and the upper layer node is called their *parent*. An *ancestor* of a node is either its parent or an ancestor's parent; similarly, if node u is an ancestor of node v, then v is said to be a *descendant* of u. Nodes at a layer i, say $u(i)$ and $v(i)$, are connected by a (single) layer-i link *iff* there is at least one layer-$(i-1)$ link between a child of $u(i)$ and a child of $v(i)$. We note that, in general, state aggregation can be employed in a more detailed manner, e.g., having a logical fully-connected topology between border nodes. While our findings could be extended to incorporate such generalizations, for simplicity of exposition we chose to focus on the above basic setting.

We assume that each peer group is composed of at most m nodes of the lower layer, where m is a *fixed* (reasonably small) number. Since $|V| \approx m^L$, we conclude that the number of layers L is $O(\log|V|)$. For each layer 1 node i, the above hierarchical procedure derives a corresponding aggregated representation of the network, which will be referred to as *the image* (of the network $G(V,E)$) *at node* i; in other words, the image of the network at a node is the aggregated topology comprising of its layer 1 peers, the layer 2 peers of its layer 2 ancestor, etc. A layer-i node which is a neighbor of a node of an upper layer $j > i$, is said to be a *layer-j border node*. The above terminology is illustrated in Figure 1. The "squares" correspond to layer 1 nodes, whereas the "clouds" represent layer 2 and layer 3 nodes. The nodes and links drawn with full lines correspond to the network image at any of the layer 1 nodes located at the upper-left part of the figure. Accordingly, the dashed nodes and links correspond to the components that are invisible at those nodes. In this example we have $m = 4$, $L = 3$ and $|V| = 32$.

Associated with each layer-i link $(u(i), v(i))$, are two numbers, $d_{(u(i),v(i))}$ and $w_{(u(i),v(i))}$, which aggregate the respective delays and weights of the actual (layer-1) paths that traverse between layer-1 descendants of $u(i)$ and $v(i)$.

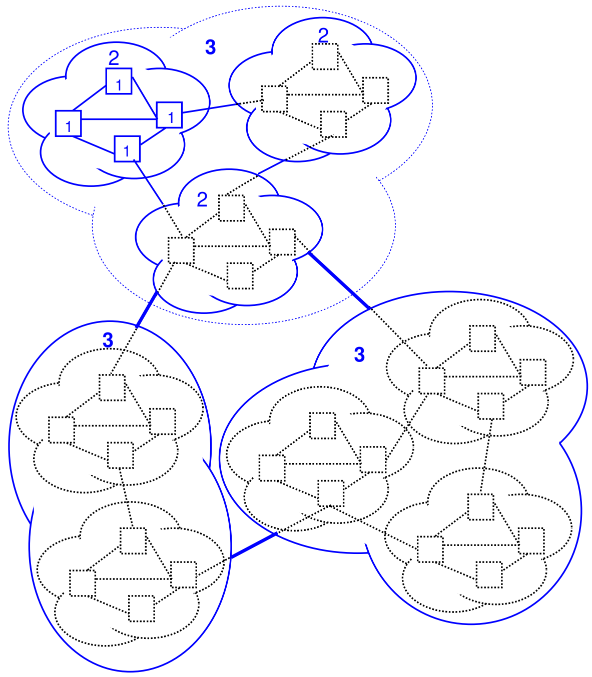

Fig. 1. Example of Hierarchical Network

The way by which this metric aggregation is performed is out of the scope of this paper (see, e.g., [13] and references therein for typical examples). Nevertheless, the following general observations can be made, independently of the precise method of metric aggregation.

H1 The aggregated weight values grow with the layer index. Indeed, the weight associated with a layer i component aggregates the weights over entire paths at the lower, $i-1$'st layer; moreover, there may be an (increasing) administrative cost imposed for using higher layers. Accordingly, we denote by q, where $q > 1$ is some value related to m, the minimal ratio between the weight value associated with any layer i link and the weight value associated with any layer $i-1$ link. For example, a typical value might be $q = \theta(\sqrt{m})$, for peer groups with diameter $\approx \sqrt{m}$.

H2 Each layer is *homogeneous* with respect to the weight metric, *i.e.*, in the image of the network seen at the source node, all layer i links have the same weight $w(i)$. Observing that the weights, e.g. administrative costs, depend to a large extent on (i) geographical distances, (ii) number of traversed (physical) components and (iii) technology, the homogeneity assumption (H2) is supported by the following observations:

- Geographical distances between components of the same layer are typically of the same order.
- Often, the infrastructure at a certain layer is similar for all the components.
- Similar number of (real, "physical") hops at each layer.
- Usually, a policy routing scheme would be employed for excluding *a priori* those links that have excessive costs (weights) for the routing policy that should be employed for the new connection. This means that the routing algorithm is presented with a different aggregated topology for each routing policy (see [9] for a related discussion).

Such pruning results in homogenizing the weights of the remaining components.
The above observations notwithstanding, the homogeneity assumption (H2) may not hold strictly, therefore we shall later relax it.

We now summarize the above discussion. Connection requests are placed to path selection processes at source (layer 1) nodes. Each connection is characterized by an end-to-end delay constraint. The aim of the path selection process is to identify whether the connection is feasible, *i.e.*, there is at least one path that obeys the constraint, and, if so, to select a (feasible) path with minimal weight. The information available to the path selection process at a node i corresponds to the image of the network at that node. Considering a specific connection request at a source s, we denote by N the number of nodes and by M the number of links, in the image of the network at s.

As an example, consider again Figure 1. Suppose that a connection request is placed at one of the layer 1 nodes drawn with full lines at the upper left part of the figure. It is easy to verify that any path to the destination, on the image of the network at the source node, is composed of at most four nodes of layer 1, two nodes of layer 2 and two nodes of layer 3, *i.e.*, $N = 8$. The network image consists of five links between layer 1 nodes, three links between layer 2 nodes and three links between layer 3 nodes, *i.e.*, $M = 11$.

B. A New Approximation

Our aim is to obtain an algorithm that finds a path which obeys the delay constraint and whose weight is at most $1 + \epsilon$ times greater than that of the optimal one (among the paths corresponding to the image seen at the source node).

Before proceeding, it is worthwhile to make the following (straightforward) observations. The main difficulty with constrained path optimization problems arises from the need to simultaneously consider two criteria, such as weight and delay; with just a single criterion, the problem is a standard shortest path one. Moreover, a special case of the constrained path problem, for which an efficient solution exists, is when one of the two criteria is hop-count.

Our algorithm is based on the above observations and on the following two ideas. The first idea is that, due to H1, the main contribution to the path weight comes from the higher layers; thus, one does not lose "much" by considering the weight criterion only at a few, uppermost, layers; in other words, in most layers, only the delay criterion needs to be considered. The second idea is that, due to the homogeneity property H2, weight optimization *at each layer* is equivalent to hop minimization at that layer.[11] Such hop minimization should observe the restrictions imposed by the delay bound. Therefore, by identifying a minimal delay path *for each possible number of hops* at each layer, and concatenating all these paths, one can identify a path of minimum weight among those that observe the delay bound; this proposition shall be formally established in the sequel. Such concatenation produces a set of paths whose size grows exponentially with the number of layers; however, since the weight (hop) criterion is considered only at a few layers, that size can be kept within a reasonable bound.

[11] As mentioned, H2 shall be later relaxed, however the weight criterion would still be translated into a hop criterion, albeit requiring a higher computational complexity.

To sum up, the algorithm partitions the network layers into two sets. In the first set, which consists of all layers except for the few highest ones, the algorithm performs a simple shortest-path computation, in order to identify a path of minimum delay between the source and each other (aggregated) node at those layers. In the second set, which consists of the highest layers, the algorithm runs a Bellman-Ford procedure, in order to identify, *for each layer*, a minimum-delay path for any possible number of hops, between the corresponding source node and each other node at that layer; here, the "corresponding source" refers to the ancestor of the original source node at that layer. It shall be shown that, by concatenating all such paths across all layers, and then adding the corresponding minimum-delay path found for the lower layers, one can identify an ϵ-optimal solution to the constrained path optimization problem.

We proceed with a formal specification of the algorithm. Let m, q, N and M be defined as in the previous subsection. Let $\widehat{L}$ be the smallest integer such that $q^{\widehat{L}} \geq \frac{m-1}{q-1} \cdot \frac{1}{\epsilon}$, *i.e.*, $\widehat{L} = O(\log_q(\frac{m-1}{q-1}) + \log_q \frac{1}{\epsilon})$.

Algorithm HIE-ϵ:

1. Considering all layers k, except for the $\widehat{L}$ upper ones (*i.e.*, for $1 \leq k \leq L - \widehat{L}$), find a minimum-delay path from the source to each layer-k node.
2. Within *each* layer k of the $\widehat{L}$ upper layers (*i.e.*, for $L - \widehat{L} + 1 \leq k \leq L$), run a Bellman-Ford minimum-delay path algorithm, in order to find, for all $1 \leq n \leq m - 1$, an n-hop minimum-delay path from the layer-k ancestor of the source node to each border node at layer k.
3. Produce $O(m^{2\widehat{L}})$ paths, by applying the following process: for $L-\widehat{L}+1 \leq k \leq L$, concatenate each of the layer-k paths identified at the previous step with each of the layer-$(k+1)$ paths found at that step, by concatenating each layer-k border node with each corresponding layer-$(k+1)$ neighbor. (Note that each of these $O(m^{2\widehat{L}})$ paths starts at the layer-$(L - \widehat{L} + 1)$ ancestor of the source node, and runs along the upper $\widehat{L}$ layers.)
4. Concatenate each of the $O(m^{2\widehat{L}})$ paths, found in the previous step, with the corresponding shortest path found, at step (1), for the lower $(L - \widehat{L})$ layers.
5. Check the end-to-end delay of each of the $O(m^{2\widehat{L}})$ paths constructed in the previous step, and delete those that do not comply with the delay constraint. If no path is left then stop: the connection is not feasible.
6. Compute the end-to-end weight of each of the remaining paths, and choose a path $\widehat{\mathbf{p}}$ with minimal weight.

Proposition 9: Algorithm HIE-ϵ correctly identifies the feasibility of the connection. When the connection is feasible, it identifies a path $\widehat{\mathbf{p}}$, whose weight is at most $1 + \epsilon$ times larger than the optimal value, *i.e.*, if $\mathbf{p}^*$ is a path of minimal weight among those that obey the delay constraint, then

$$W(\hat{p}) \leq (1+\epsilon) \cdot W(\mathbf{p}^*). \quad (9)$$

□

Proposition 10: The number of steps of Algorithm HIE-ϵ is $O(N \log N + M + \log(\frac{1}{\epsilon}) \cdot (\frac{1}{\epsilon})^{2 \log_q m})$. □

Thus, Algorithm HIE-ϵ is an ϵ-optimal, strongly polynomial approximation, with an advantageous scalability property: for large networks, its complexity converges to that of a standard shortest-path algorithm. We also note that, for a fixed accuracy ϵ, the ratio between the complexity bounds of Algorithm HIE-ϵ and the strongly polynomial algorithm of [11] grows faster than N^2.

We now relax the homogeneity assumption (H2). Suppose that the ratio between the weights of any two links belonging to the same layer is bounded by some constant $w \geq 1$. A simple approach (and probably not the most efficient) for adapting Algorithm HIE-ϵ to handle this case is to expand, in the network image, each link l at a layer k into a sequence of $\lceil \frac{w_l}{w_{min}(k)} \rceil$ links, where $w_{min}(k)$ is the minimal link weight at layer k; the delay d_l of the original link is equally split among the new links. It is easy to see that running Algorithm HIE-ϵ on the new topology, with $\widehat{L} = O(\log_q(\frac{w \cdot (m-1)}{q-1}) + \log_q \frac{1}{\epsilon})$,[12] still renders an ϵ-optimal solution. The complexity bound now becomes

$$O(Nw \log(Nw) + wM + \log(\frac{1}{\epsilon}) \cdot (\frac{1}{\epsilon})^{2 \log_q (mw)}),$$

and, as $w = O(1)$, we get a complexity of

$$O(N \log N + M + \log(\frac{1}{\epsilon}) \cdot (\frac{1}{\epsilon})^{2 \log_q (mw)}),$$

which still offers the same scalability.

IV. Conclusions

In this paper we addressed two major problems related to the design of QoS routing schemes. The first is the need to efficiently combine the routing decision with the service disciplines employed in the network, while the second is the need to perform, at times, multi-objective path optimization.

First, we considered routing schemes that are designed to operate in conjunction with rate-based service disciplines, following recent proposals. After overviewing a basic solution scheme, which has been proposed in the literature, we focused on two important objectives, namely reducing the computational complexity and optimizing the networkwide consumption of resources. Considering the first objective, we established two schemes, based on either rate or hop quantization, which significantly reduce the computational complexity, while still offering efficient solutions. Considering the second objective, we formulated several optimization rules, and established their solutions. These schemes provide the network controllers with the ability to reduce the consumption of bandwidth and balance the loads across the network. It should be noted that the establishment of a QoS routing mechanism with such properties, namely reasonable computational burden and efficient resource management, is mandatory for the deployment of the related, rate-based, service disciplines.

We then proceeded to consider a generic double-objective path optimization problem, where one wants to minimize some "cost" metric while observing an end-to-end bound such as delay. This problem is known to be NP-complete and approximations have been proposed. However, those solutions addressed the general problem, on arbitrary topologies, hence did not consider the characteristic structure of large-scale communication networks. In this study, we exploited the typical hierarchical structure of such networks, in order to obtain a new approximation algorithm, which has superior performance in terms of scalability. In particular, as the network grows in size, the complexity bound of our approximation converges to that of a standard shortest-path algorithm. Such a property is particularly important, as the ability to deploy networks of giant scales is one of the major current challenges of communication networking.

Our study focused on *worst-case* analysis, and an important open issue is to test the *average* performance of the various algorithms in typical network scenarios. More generally, the establishment of an efficient QoS routing solution, which fully addresses the various problems and challenges posed by large-scale, broadband networks, remains to be a complex problem, with many facets still to be addressed. Yet, our study provides further evidence that a proper QoS routing mechanism should be designed as an integral component of the overall network architecture; specifically, it indicates that, by exploiting the relation between routing, service provision and topology representation, new roads to the solution are being opened.

[12] here, q is the value that corresponds to the expanded network, which is larger than the value in the original network, hence reducing the complexity.

References

[1] Private Network-Network Interface Specification v1.0 (PNNI). ATM Forum Technical Committee, March 1996.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Networks Flows.* Prentice-Hall, New Jersey, 1993.

[3] G. Apostolopoulos, R. Guérin, S. Kamat, and S. Tripathi. Quality of service based routing: A performance perspective. In *Proceedings of SIGCOMM*, Vancouver, Ontario, CANADA, August 1998.

[4] A. Bestavros and I. Matta. Load Profiling for Efficient Route Selection in Multi-Class Networks. In *Proceedings of IEEE ICNP'97*, Atlanta, GA, October 1997.

[5] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick. A Framework for QoS-based Routing in the Internet – RFC No. 2386. Internet RFC, August 1998.

[6] M.R. Garey and D.S. Johnson. *Computers and Intractability.* Freeman, San Francisco, 1979.

[7] L. Georgiadis, R. Guérin, V. Peris, and K. N. Sivarajan. Efficient Network QoS Provisioning Based on per Node Traffic Shaping. *IEEE/ACM Transactions on Networking*, 4(4):482–501, August 1996.

[8] R. Guérin and A. Orda. QoS-based routing in networks with inaccurate state and metrics information. *IEEE/ACM Transactions on Networking*, 1999. To appear.

[9] R. Guérin, A. Orda, and D. Williams. QoS routing mechanisms and OSPF extensions. Internet Draft, December 1996. Also in Proceedings of the 2nd IEEE Global Internet Mini-Conference, Phoenix, AZ, November 1997.

[10] S. Gupta. Performance Modelling and Management of High-Speed Networks. Ph.D. Thesis, University of Pennsylvania, Department of Systems, 1993.

[11] R. Hassin. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations Research*, 17(1):36–42, February 1992.

[12] C. Huitema. *Routing in the Internet.* Prentice Hall, Englewood Cliffs, NJ, 1995.

[13] W. C. Lee. Topology aggregation for hierarchical routing in ATM networks. In *Proceedings of SIGCOMM*, pages 82–92, April 1995.

[14] D. H. Lorenz and A. Orda. QoS Routing in Networks with Uncertain Parameters. *IEEE/ACM Transactions on Networking*, 6(6):768–778, December 1999.

[15] Q. Ma and P. Steenkiste. Quality of Service Routing for Traffic with Performance Guarantees. In *Proceedings of IWQoS'97*, Columbia University, New York, NY, May 1997.

[16] A. Orda. Routing with End to End QoS Guarantees in Broadband Networks. EE Pub. No. 1089, Department of Electrical Engineering, Technion, Haifa, Israel, May 1997. Revised: February 1999. ftp://ftp.technion.ac.il/pub/supported/ee/Network/ao.qosr97.ps.

[17] A. K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Multiple Node Case. *IEEE/ACM Transactions on Networking*, 2:137–150, 1994.

[18] C. Pornavalai, G. Chakraborty, and N. Shiratori. QoS Based Routing Algorithm in Integrated Services Packet Networks. In *Proceedings of IEEE ICNP'97*, Atlanta, GA, October 1997.

[19] S. Shenker, C. Partridge, and R. Guerin. Specification of Guaranteed Quality of Service – RFC No. 2212. Internet RFC, November 1997.

[20] Z. Wang and J. Crowcroft. Quality-of-Service Routing for Supporting Multimedia Applications. *IEEE JSAC*, 14(7):1288–1234, September 1996.

[21] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10):1374–1399, October 1995.

Ariel Orda (S'94-M'92-SM'97) received the B.Sc. (summa cum laude), M.Sc., and D.Sc. degrees in Electrical Engineering from the Technion -Israel Institute of Technology, Haifa, Israel, in 1983, 1985, and 1991, respectively.
Since 1994, he has been with the Department of Electrical Engineering at the Technion, where he is currently a Senior Lecturer and the Academic Head of the Computer Networking Laboratory. In 1993–1994, he was with the Center for Telecommunication Research, Columbia University, New York, NY, as a Visiting Scientist, and with AT&T Bell Laboratories, Murray Hill, NJ, as a Scientific Consultant. During the summers of 1992 and 1995–1998, he held visiting positions at AT&T Bell Laboratories, Murray Hill, NJ, IBM Watson, Hawthorne, NY, and Bell Laboratories - Lucent Technologies, Holmdel, NJ. Since 1991 he has held several consulting positions with the Israeli industry.

His current research interests include QoS routing, the application of game theory to computer networking, network pricing, and distributed network algorithms.

Dr. Orda received the Award of the Chief Scientist in the Ministry of Communication in Israel, a Gutwirth Award for outstanding distinction, and the Research Award of the Association of Computer and Electronic Industries in Israel.